Autonomous Reverse Parking System Based on Robust Path Generation and Improved Sliding Mode Control

Xinxin Du and Kok Kiong Tan, *Member, IEEE*

***Abstract*—Some commercial vehicle models have been equipped with semiautonomous parking systems to a certain extent. However, gaps to fully automated solutions still exist, and cost considerations further constrain their acceptance among consumers. This paper proposes a low-cost vision-based approach to a fully self-reverse parking system. It consists of four key modules: a novel path-planning module ensures that a feasible path is available under any initial poses, which frees human intervention completely; a modified sliding mode controller on the steering wheel is designed for path following; image processing with Kalman state prediction provides consistent and real-time estimation on the vehicle pose; and a robust overall control scheme ensures that the vehicle can accurately park along the slot center line without intrusion into adjacent slots. Experimental results based on 216 on-field tests under different illumination conditions showed that the proposed system was able to accurately and consistently park the vehicle in all cases with a 4.71-cm RMS offset distance from the center line and a 1.24° RMS orientation deviation. With its easy setup and excellent performance, this system can be practically and robustly implemented to existing vehicles with minimal additional cost.**

***Index Terms*—Autonomous reverse parking, extended Kalman filter, path planning, ridge detector, sliding mode control (SMC).**

I. Introduction

RECENT advances in vehicles with autonomous parking systems have drawn much attention and interest from researchers, the media, and the general public. So far, several commercial car models have adopted, to a certain extent, various types of self-parking systems. However, most of them still require some degree of human intervention. For example, in the Toyota Prius [1] and BMW 7 series [2], the driver needs to regulate the speed of the vehicle by pressing and releasing the brake pedal. The system only takes control of the steering wheel. Thus, gaps still exist in the attempt to render a fully self-parking system.

Parking can be categorized into two major groups based on the parking slot orientation with respect to the road, i.e., reverse parking and parallel parking, and the focus of this paper is on reverse parking. All concepts except path generation can be still directly applied to parallel parking. To realize autonomous parking, typically, three basic steps are needed, i.e., target position designation, path planning, and path following/tracking.

The target position designation is to identify the available parking slot and track the desired slot during the parking process. Based on the sensors utilized, it can be roughly divided into three categories, including active ultrasonic or laser-sensor-based, infrastructure-based, and vision-based methods.

1) The active-sensor-based method is the most common method for parallel parking. The system collects range data as the vehicle passes by a free parking space and registers the range data using odometry to create a depth map. Some typical examples can be found in [3]–[5]. However, as pointed out in [6], this approach usually fails in reverse parking because the incident angle between the sensor and the side facets of nearby vehicles is too large. Moreover, the final parking accuracy depends on the nearby vehicles' parking orientation.
2) The infrastructure-based-method requires plenty of modifications on existing car park systems. In [7], a local Global Positioning System, a car park digital map, and communication with the parking administration systems are required. Yan *et al.* [8] proposed a smart parking system consisting of wireless transceivers, parking belts, and infrared devices. However, these approaches may not be applicable in a short time due to the requirement of additional hardware installation on current car parks.
3) Vision-based methods provide the simplest and cheapest solution among these three categories. In [9] and [10], the vision system was used to recognize adjacent vehicles as the boundary of the available car park slot, whereas Wang *et al.* [11] used fish-eye cameras and a Canny edge detector to identify the car park slot markings. However, conventional vision-based methods may not be robust enough to tackle various illumination situations in different types of car parks [5]. To deal with a poorly lit environment, which is common for indoor/underground car parks, a new system was proposed in [12] to recognize 3-D information by analyzing the light stripe from a light plane projector. However, it requires external references to work properly, such as vehicles in the adjacent slots, pillars near the slot boundary, etc.

Taking all these into consideration, we select to use one webcam for the target position designation. A ridge detector

Manuscript received February 26, 2014; revised May 19, 2014 and July 27, 2014; accepted August 28, 2014. This work was supported by Toyota Tsusho Asia Pacific Private Ltd. The Associate Editor for this paper was H. Huang.

The authors are with the Department of Electrical and Computer Engineering, Faculty of Engineering, National University of Singapore, Singapore 117583 (e-mail: a0107257@nus.edu.sg; eletankk@nus.edu.sg).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2014.2354423

is implemented to detect slot markings, which is proven to be more robust than the conventional edge detector [13]. Furthermore, it works extremely well under poor lighting conditions.

In the domain of path planning for reverse parking, many continuous-curvature path generators based on geometrical information have been proposed in literature. Pioneering works by Dubins [14] and Reeds and Shepp [15] proved that the shortest paths for a car to move from one pose to another consist of arcs of circles (the minimum turning radius) and straight line segments. Based on this concept, Wada *et al.* [7] and Yan *et al.* [16] proposed several algorithms to address this problem in their respective works. Some other researchers [7], [17] adopted the concepts of way points. Wang *et al.* [11] proposed the double circular trajectory in which the first arc follows the minimum turning arc and the second arc depends on the available space. However, the key idea is still based on the arc-line type of path. A common disadvantage of their works is that a feasible path is not always guaranteed under the constraint of the minimum turning radius. The success in finding a path strongly depends on the vehicle initial pose relative to the car park slot, e.g., the double circular trajectory [11] requires the initial pose to be parallel to the road. Oetiker *et al.* proposed a totally different algorithm from the navigation field point of view [18]. However, the critical problem is to determine the way point, and the available paths depend on a precalculated vector field.

Leveraging on these considerations, we propose an arc-line-based path-planning algorithm that guarantees to find a feasible path under any initial pose.

For the path following controller design, there are a lot of designs available, varying from conventional PID to more advanced adaptive controllers, which include sliding mode control (SMC) [19], input–output linearization [20], a fuzzy neural network [21], model predictive control [22], and backstepping-based control [23]. Among them, SMC has been claimed to be one of the ideal solutions considering stability, computation, complexity, and practical maneuverability [24]. It yields a fast response, good transient tracking, and robustness with respect to system uncertainties and disturbances.

However, it has its own drawbacks as well. As shown later, the control signal is not smooth using conventional approaches when it is applied to nonholonomic vehicles. Several unexpected spikes arise that induce strong vibrations to the vehicle. To mitigate this, we propose a revised SMC design that suppresses these spikes to a large extent.

In this paper, a complete set of solutions to the self-reverse parking system is proposed. The system utilizes a monocular camera with a Kalman filter [25] to generate real-time feedback on the vehicle pose. A supervision control scheme prevents the vehicle from entering adjacent parking slots and ensures that the vehicle parks along the slot center line accurately. The system is proven, through experiment data, to be robust, reliable, and practical. The main contributions of this paper include the following.

1) A robust path-planning algorithm. It is proven that a feasible path can be generated regardless of the vehicle initial pose with respect to the car park slot.
2) A smooth SMC path following controller. It is able to eliminate the control signal spikes induced by the conventional SMC and to provide a higher level of human comfort without jerking or shaking.

This paper is organized as follows. Sections II and III provide detailed explanation on the main contributions of the paper, including path generation and SMC. Section IV briefly explains the image processing process, and Section V elaborates the interactions between individual modules. The experiment setup and results are illustrated in Section VI, and conclusions are drawn in Section VII.

II. Path Generation

It was proven in [14] and [15] that the shortest path between two vehicle poses consists of straight line segments connected with circular arcs of minimum turning radius. Based on this result, we propose a novel path-planning algorithm that guarantees the existence of a feasible path at any vehicle pose. Although the curvature profile at the transition points between arcs and lines is discontinuous [21], it is not critical as far as this application is concerned since the vehicle moves at a slow speed (1–2 km/h).

To better illustrate the algorithm, the coordinate system in Fig. 6 is defined with its origin at the center of the bottom boundary. The last segment of all the paths must be a straight line along the Z-axis; otherwise, part of the vehicle body will have to pass through the forbidden areas before reaching the destination point.

For a path consisting of two segments, the first segment must be an arc. Depending on the vehicle moving direction and steering direction, there are four cases. Similarly, for a path consisting of three segments, the first segment has four cases as well; thus, the total number of paths in this case is $4 \times 4 = 16$. Analogously, for a path consisting of n segments, each segment except the last has four cases, and the possible path number is 4^{n-1}.

Given a vehicle pose, since the number of path segments is unknown, the program has to check from the path with one segment until n segments. Then, the total number of paths to be examined equals $1 + 4 + 4^2 + \cdots + 4^{n-1} = (4^n - 1)/3$. Based on our implementation, when the check is up to $n = 4$, there will be at least one feasible path for 90.5% of the possible poses. Furthermore, increasing n, the total number of paths to check will increase by 256, and no explicit solution is available for a five-segment path, although the arc-line combination and sequence are fixed. The computation requirement increases dramatically. Therefore, it is not worth trying for the remaining 9.5% of poses. A compensation solution will be proposed later to cover these remaining poses.

When $n = 4$, the total number of paths is 85, out of which some are redundant particularly for those with four segments. Therefore, instead of checking all 85 paths, we used 21 nonredundant paths as the basic set. The details are tabulated in Table I. The first letter refers to the steering direction (**S**traight, **L**eft, and **R**ight), and the second letter refers to the moving direction (**F**orward and **B**ackward).

TABLE I
BASIC PATH MODELS

Segment	Path	Number
n=1	SB	1
n=2	RB/LB/RF/LF-SB	4
n=3	SB/SF-(All cases with n=2) RB/RF-LF-SB LB/LF-RF-SB	12
n=4	SB/SF-LB-RB-SB SB/SF-RB-LB-SB	4

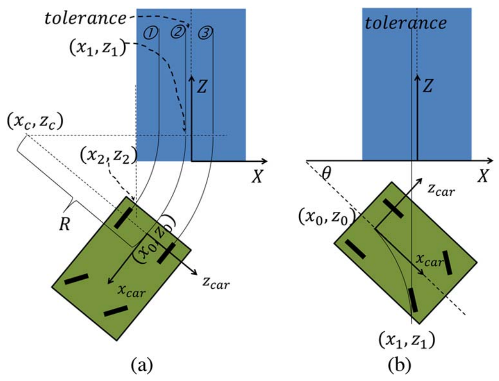


Fig. 1. Paths with two segments.

To facilitate the illustration on how different paths are checked, we define the following parameters: vehicle position (x_0, z_0) is represented by the center point of its rear axle; its orientation θ is the angle from the $+X$-direction to its body's center axis (see Fig. 1); the car width is W_{car}; and the parking slot width is W_{slot}. For a given car pose (x_0, z_0, θ), its turning center (x_c, z_c) with radius R is calculated according to its steering direction (top signs are for right steering) as

$$x_c = x_0 \pm R\sin\theta \quad (1)$$
$$z_c = z_0 \mp R\cos\theta. \quad (2)$$

A. Paths With Two Segments

To check a path with two segments for RB–SB, as shown in Fig. 1(a), the path must satisfy the following conditions.

1) $|x_1| \leqslant$ tolerance, where (x_1, z_1) is the transition point, and tolerance defines the allowable offset from the parking slot center line. From a geometrical relationship, $x_1 = x_c + R$.
2) Arc① intersects with the left boundary, or $x_1 - W_{\text{car}}/2 \geqslant -W_{\text{slot}}/2$.
3) The lower intersection point (x_2, z_2) is below the bottom boundary. Arc① is given in (3). By substituting $x_2 = -W_{\text{slot}}/2$ and picking the smaller value, z_2 is shown in

$$(x - x_c)^2 + (z - z_c)^2 = (R - W_{\text{car}}/2)^2 \quad (3)$$
$$z_2 = z_c - \sqrt{(R - W_{\text{car}}/2)^2 - (x_c + W_{\text{slot}}/2)^2}. \quad (4)$$

Once these validations are passed, the path can be uniquely determined by points (x_0, z_0) and (x_1, z_1).

For the path in Fig. 1(b) (RF–SB), the only thing to ensure is the first criterion as none of the vehicle body will enter the forbidden areas along the path. The remaining two paths with two segments (LB–SB and LF–SB) can be similarly worked out since they are symmetrical with the aforementioned two examples.

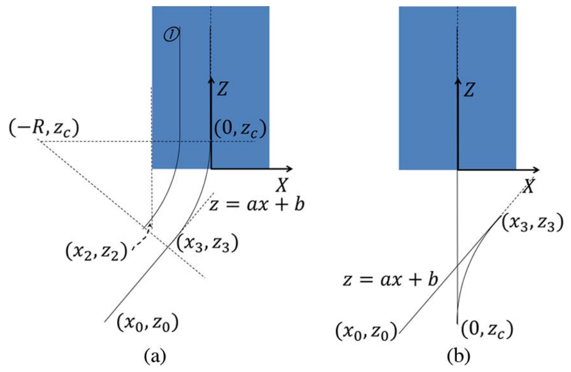


Fig. 2. Paths with three segments, i.e., *Line–Arc–Line*.

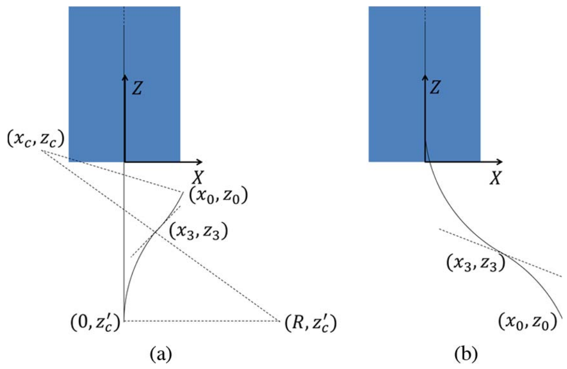


Fig. 3. Paths with three segments, i.e., *Arc–Arc–Line*.

B. Paths With Three Segments

For paths consisting of three segments, they can be divided into two groups according to the first segment, i.e., *Line–Arc–Line* (see Fig. 2) and *Arc–Arc–Line* (see Fig. 3).

For the first group, the straight line $z = ax + b$ can be determined based on the vehicle initial pose, with $a = \tan\theta$, and $b = z_0 - x_0 \tan\theta$. The first transition point (x_3, z_3) satisfies

$$z_3 = ax_3 + b \quad (5)$$
$$(x_3 - (-R))^2 + (z_3 - z_c)^2 = R^2 \quad (6)$$
$$a(z_3 - z_c)/(x_3 - (-R)) = -1. \quad (7)$$

From the three aforementioned equations, all transition points (x_3, z_3) and $(0, z_c)$ can be derived as

$$x_3 = -R + aR/\sqrt{1 + a^2} \quad (8)$$
$$z_3 = -aR + b + a^2R/\sqrt{1 + a^2} \quad (9)$$
$$z_c = aR + b + R/\sqrt{1 + a^2}. \quad (10)$$

By comparing the relative position between (x_0, z_0) and (x_3, z_3), the moving direction for the first segment is determined.

Similarly, a collision check on the left boundary is necessary to ensure that the vehicle does not enter forbidden areas. The condition is $z_2 < 0$, where z_2 is defined in (4), with $x_c = -R$.

For the path in Fig. 2(b), the transition points can be similarly derived with the arc center at (R, z_c) instead of $(-R, z_c)$. For the collision check, as long as the vehicle is outside the forbidden areas at (x_3, z_3), it will be safe along the whole path.

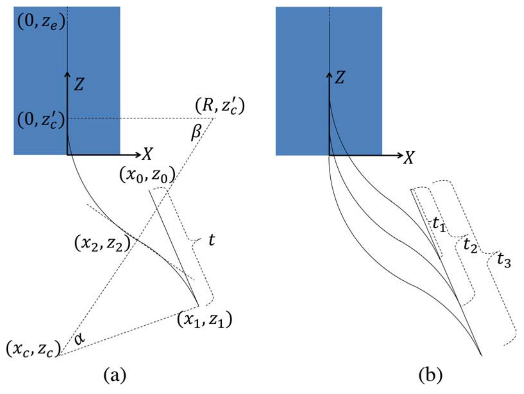


Fig. 4. Paths with four segments and its nonuniqueness.

However, there is no short-form solution for this case. Instead of using the analytical approach, we implement a more general collision checking method, which will be illustrated at the end of this section.

All the remaining paths in the *Line–Arc–Line* group can be similarly worked out based on the vehicle initial pose and the corresponding turning centers.

If the path is *Arc–Arc–Line*, as shown in Fig. 3, the transition points for Fig. 3(a) can be derived as follows. The center for the first arc (x_c, z_c) is given by (1) and (2). Since the two arcs are tangent with each other, the distance between the two centers satisfies

$$(x_c - R)^2 + (z_c - z_c')^2 = (2R)^2. \tag{11}$$

Then, z_c' can be calculated by selecting the smaller value. Transition point (x_3, z_3) can be then derived as

$$x_3 = 0.5(x_c + R) \tag{12}$$
$$z_3 = 0.5(z_c + z_c'). \tag{13}$$

The path shown in Fig. 3(b) is not covered as it is redundant to the path with four segments.

C. Paths With Four Segments

For the path with four segments, only the type *Line–Arc–Arc–Line* is checked. Even for this single type, the number of solutions given an initial vehicle pose is infinity. For example, in Fig. 4(a), the path is related to the length of the first straight line t. As shown in Fig. 4(b), three different t result in three different paths.

Define the length of the first straight line segment as t, then the remaining transition points (x_1, z_1), (x_2, z_2), and $(0, z_c')$ and corresponding turning centers (x_c, z_c) and (R, z_c') in Fig. 4(a) can be specified as

$$x_1 = x_0 + t\cos\theta \tag{14}$$
$$z_1 = z_0 + t\sin\theta \tag{15}$$
$$x_c = x_1 + R\sin\theta \tag{16}$$
$$z_c = z_1 - R\cos\theta \tag{17}$$
$$z_c' = z_c + \sqrt{4R^2 - (x_c - R)^2} \tag{18}$$

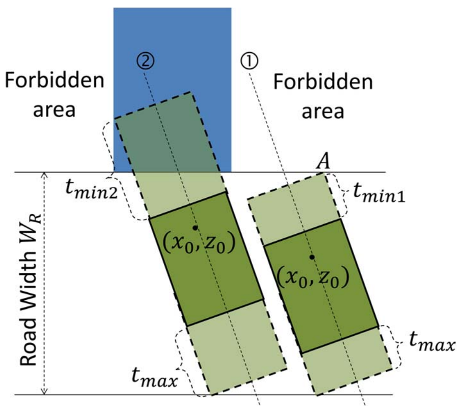


Fig. 5. Range of t for paths with four segments.

$$x_2 = 0.5(x_c + R) \tag{19}$$
$$z_2 = 0.5(z_c + z_c'). \tag{20}$$

Theoretically, t should be selected according to the shortest travelling distance d, as shown in

$$d = t + R\alpha + R\beta + (z_e - z_e') \tag{21}$$

where

$$\alpha = 2\arcsin\left(\frac{\sqrt{(x_1 - x_2)^2 + (z_1 - z_2)^2}}{(2R)}\right) \tag{22}$$

$$\beta = 2\arcsin\left(\frac{\sqrt{x_2^2 + (z_2 - z_c')^2}}{(2R)}\right). \tag{23}$$

However, the explicit form of d is too complicated to analytically get its minimum as it consists of terms with squares under square root, square root inside arcsine, etc. It also depends on the vehicle initial pose (x_0, z_0, θ). A numerical method is preferred to solve this minimization problem since the range of t can be estimated roughly.

The range of t is defined in Fig. 5, where the solid dark areas mark the vehicle initial location and the dotted light areas mark the extreme locations it might end in when moving straight forward or backward. The upper limit of t is determined by the road boundary in front of the parking slot, as shown in Fig. 5. The maximum distance that it is able to move forward to from location (x_0, z_0) is $t_{\max}$; otherwise, it will move beyond the road boundary. The width for a typical two-lane road W_R is about 8 m, which can be used as an estimation to determine the upper limit of t as

$$t_{\max} = (z_0 + W_R)/(\cos\theta) - (l + l_1) + W_{car}/(2\tan\theta) \tag{24}$$

where l and l_1 are defined in Fig. 7(a).

For its lower limit, two cases may occur, as indicated by path ① and ②, depending on the x location of point A, or x_A. On path ①, the maximum distance the vehicle is able to move backward from location (x_0, z_0) is $t_{\min 1}$; otherwise, it will enter the forbidden area. While on path ②, it is able to move back further into the slot before it enters the forbidden area. From the geometrical relationship, lower limit $t_{\min}$ can be shown as

$$t_{\min} = \begin{cases} \frac{z_0}{\sin\theta} - l_1 + \frac{W_{\text{car}}}{2\tan\theta}, & |x_A| > \frac{W_{\text{slot}}}{2} \\ \frac{x_0 + 0.5W_{\text{slot}}}{\cos\theta} - l_1 + \frac{W_{\text{car}}\tan\theta}{2}, & |x_A| \leqslant \frac{W_{\text{slot}}}{2} \end{cases} \quad (25)$$

where $x_A = x_0 - (W_{\text{car}}/2\sin\theta) - (z_0/\tan\theta)$.

Since the scale of the parking range is several meters, a discrete series of t with a step size of 0.01 m from $t_{\min}$ to $t_{\max}$ will provide an accurate enough estimation in locating the minimum value of d. However, each element in the t series needs to meet the following criteria before being passed to d.

1) Equation (18) must be real.
2) Similar to (4), the intersection point between the vehicle's inner wheel path and the parking slot's right boundary has to be below the X-axis.

The two criteria lead to the following inequalities:

$$(x_0 + R\sin\theta - R + \cos\theta t)^2 \leqslant 4R^2 \quad (26)$$

$$\sin\theta t + \sqrt{4R^2 - (x_0 + R\sin\theta - R + \cos\theta t)^2} \leqslant R\cos\theta + \sqrt{(R - 0.5W_{\text{car}})^2 - (R - 0.5W_{\text{slot}})^2} - z_0. \quad (27)$$

In summary, for a path with four segments, first, get the range of t based on (24) and (25). Discretize t with a step size of 0.01 m from $t_{\min}$ to $t_{\max}$. Select the valid elements in t that satisfy (26) and (27). Pass all the valid elements to (14)–(23) to get the minimum of d and its corresponding transition points.

D. Refined Paths

As aforementioned, the feasible path is not available at 9.5% of the poses just based on these 21 basic paths. To compensate for this shortfall, we propose the following path generation algorithm incorporated with these basic paths.

1) For a given initial vehicle pose (x_0, z_0, θ), get the shortest feasible path out of the basic paths if such a path exists.
2) Assume that the vehicle moves LF (or LB, RF, or RB) by a small angle $\delta\theta$.
3) Based on the new vehicle pose, get the shortest feasible path out of the basic paths if such a path exists.
4) Assume that the vehicle further moves in the same direction and steering angle by another $\delta\theta$.
5) Repeat 3) and 4) until a feasible path is found or the vehicle hits the forbidden area.
6) Change to another combination of moving and steering directions (LB, RF, or RB), and repeat 2)–5).
7) The optimal path is selected as the path with the shortest traveling distance.

If the path is still not available, move the vehicle forward or backward along a straight line by a small distance, and repeat the aforementioned steps.

Fig. 6 illustrates this process briefly. Path 1 is from the basic path, whereas Path 2 and Path 3 are generated by assuming that the vehicle steers to the right and moves backward to Point A and Point B first. As will be shown in the simulation in Section IV, the proposed path generation algorithm can provide

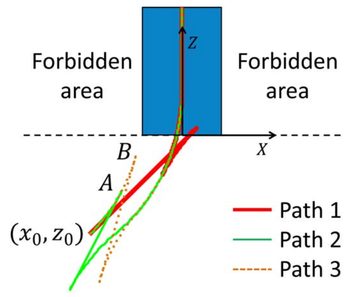


Fig. 6. Illustration on the path refining process.

a feasible path at each single pose, even when the vehicle is inside the parking slot.

E. Collision Check

As aforementioned, due to the complexity in analytically checking collision under some cases, a general method is proposed, which not only can validate a generated path but also check the vehicle collision status during the real run as long as the vehicle pose is known.

First, define a new coordinate system $x_{\text{car}}-z_{\text{car}}$ attached to the vehicle, as shown in Fig. 1(a). Every point on the vehicle boundaries can be easily expressed in this coordinate system. By taking points on the boundaries with a step size of 0.01 m, we can get a collection of points representing the vehicle boundaries as

$$C_{\text{car}} = \begin{bmatrix} x_1 & x_2 & \ldots & x_n \\ z_1 & z_2 & \ldots & x_n \end{bmatrix}. \quad (28)$$

The translation matrix T from the $x_{\text{car}}-z_{\text{car}}$ coordinate to the X–Z coordinate is

$$T = \begin{bmatrix} \cos\theta & -\sin\theta & x_0 \\ \sin\theta & \cos\theta & z_0 \end{bmatrix} \quad (29)$$

where (x_0, z_0, θ) is a given vehicle pose.

The vehicle boundary points expressed in the X–Z coordinate are

$$C = T\begin{bmatrix} C_{\text{car}} \\ 1_{1\times n} \end{bmatrix}. \quad (30)$$

If any of the boundary points in C fall in the forbidden area, collision occurs.

III. SMC Design

To control the vehicle to follow the generated path, we apply SMC on the vehicle steering wheel. The controller will calculate a target steering angle based on the vehicle position and orientation errors with respect to the target point on the path. The vehicle velocity is not under the control of SMC. To implement SMC, the nonholonomic vehicle model is adopted as follows [see Fig. 7(a)]:

$$\dot{x}(t) = \cos\theta(t) \cdot v(t) \quad (31)$$
$$\dot{z}(t) = \sin\theta(t) \cdot v(t) \quad (32)$$
$$\dot{\theta}(t) = \tan\varphi(t) \cdot v(t)/l \quad (33)$$

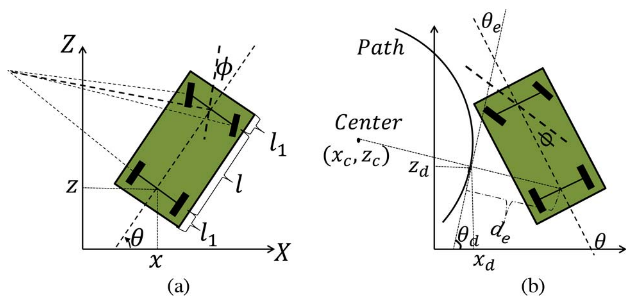


Fig. 7. Nonholonomic vehicle model and pose error definition.

where (x, z, θ) defines the vehicle pose with respect to the global coordinate system, v is the car moving velocity (forward is $+$), φ is the steering angle (left is $+$), and l is the interwheel axle distance.

The target point on the path is defined as the closest point to the vehicle control point. For the particular relative position shown in Fig. 7(b), the target point and the target orientation are shown as

$$\begin{cases} x_d = x_c + R|x - x_c|/\sqrt{(x-x_c)^2 + (z-z_c)^2} \\ z_d = z_c - R|z - z_c|/\sqrt{(x-x_c)^2 + (z-z_c)^2} \\ \theta_d = -\arctan\left(\frac{(z-z_c)}{(x-x_c)}\right) \end{cases}. \tag{34}$$

For other relative positions, similar equations can be derived accordingly.

Then, the position and orientation error with respect to the target point can be derived from their geometrical relationship as

$$d_e = -\sin\theta_d(x - x_d) + \cos\theta_d(z - z_d) \tag{35}$$

$$\theta_e = \theta - \theta_d. \tag{36}$$

The position error dynamics can be worked out based on (31), (32), and (35) as

$$\begin{aligned} \dot{d}_e &= v\sin\theta_e - \dot{\theta}_d\left[\cos\theta_d(x - x_d) + \sin\theta_d(z - z_d)\right] \\ &= v\sin\theta_e \qquad \text{(from geometry)}. \end{aligned} \tag{37}$$

The following sliding surface is proposed such that both position and orientation errors converge to zero:

$$s = \dot{d}_e + kd_e = v\sin\theta_e + kd_e. \tag{38}$$

k has to be nonnegative for a stable sliding surface [26].

The sliding surface convergence rate can be shown as

$$\dot{s} = v^2\cos\theta(\tan\varphi - \tan\varphi_d)/l + kv\sin\theta_e. \tag{39}$$

Define $\dot{s} = -f(s)$, where $f(s)$ could be any nondecreasing odd function or a combination of them, such as sign, saturation, etc. In order to suppress the chattering effect and prevent a sudden change in the control signal due to $f(s)$, we choose the arctan function that is nondecreasing odd and, more importantly, continuous as

$$\dot{s} = -Q\arctan\left(\frac{s}{P}\right). \tag{40}$$

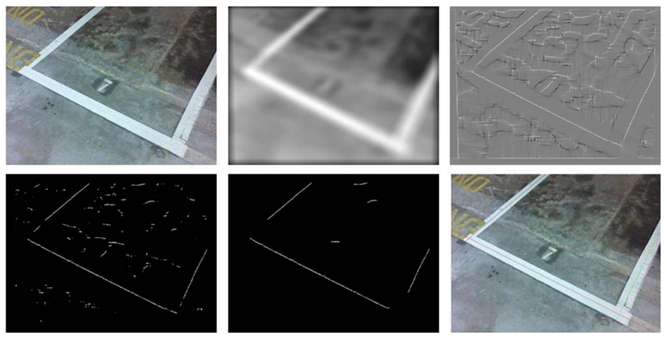


Fig. 8. Illustration on image processing (rowwise: original, smoothed, ridgeness, threshold, filtering, and line assignment).

Equations (39) and (40) provide the control law on φ as

$$\varphi = \arctan\left(-\frac{lQ\arctan\left(\frac{s}{P}\right)}{v^2\cos\theta_e} - \frac{lk\tan\theta_e}{v} + \tan\varphi_d\right). \tag{41}$$

A simple Lyapunov stability check using $V = s^2/2$ implies that P and Q must be positive for a stable system.

Conventionally, we should go on to calibrate k, P, and Q based on their constraints and the corresponding control performance. However, for this particular application, a fixed value of k and Q is not sufficient to provide a smooth control signal, although some precautions have been taken in designing $f(s)$.

The main reason is due to the v terms in the denominator in (41). $|v|$ may drop close to zero occasionally, which unnecessarily magnifies the control signal in φ even when both s and θ_e are small. Such fast switching in the steering directions induces strong jerking experience to the driver.

To compensate for this effect, we propose the following designs for Q and k, where Q' and k' are the new parameters to be tuned. Note that Q is ensured to be nonnegative as $-\pi/2 < \theta_e \leqslant \pi/2$. The resultant control law is shown in

$$Q = \left(\frac{v^2\cos\theta_e}{l}\right)Q' \qquad k = \left|\frac{v}{l}\right|k' \tag{42}$$

$$\varphi = \arctan\left(-Q'\arctan\frac{s}{P} - sg(v)k'\tan\theta_e + \tan\varphi_d\right). \tag{43}$$

To summarize, (34)–(36), (38), and (43) form the full set of the SMC design on the steering wheel, which can generate a smooth and fast response for path following. Note that the vehicle pose information involved is based on the Kalman filter estimation instead of image processing. The Kalman filter enhances the continuity of the pose estimation in each SMC control cycle, which in turn increases the smoothness of the control action.

IV. Image Processing

As aforementioned, we implement a ridge detector to extract slot markings. The top row in Fig. 8 illustrates the results from the ridge calculation. Unlike other detectors, it directly extracts the medial axis of the slot boundary markings. Their ridge values are larger than the rest of the image.

Therefore, a simple fixed value threshold can be applied on the ridge image. Then, a noise filtering step, consisting of connected components labeling the operation and removal of components with a small number of pixels (< 25 pixels), is applied to remove the noise pixels. A sequential random sample consensus (RANSAC) line fitting algorithm is implemented to detect all the lines representing the visible parking slot boundaries, followed by a line assignment process to determine which boundary the lines belong to. Other robust line fitting algorithms are also applicable to replace RANSAC, e.g., the Radon transform [11]. Fig. 8 illustrates the whole process.

This line assignment process is based on the Kalman filter prediction. From the predicted vehicle pose, a simulated parking slot image can be realized; thus, all its boundary medial axes can be expressed in the image coordinate system. The fitted lines from the real image will be then compared with each of the simulated medial axis in terms of gradient and average pixel distance d_{ave} as

$$d_{\text{ave}} = \frac{\left(\sum_{i=1}^{N} |Cx_i + Ey_i + F/2|\right)}{(N\sqrt{C^2 + E^2})} \tag{44}$$

where (x_i, y_i) represents one pixel on the fitted line, N is the total number of pixels on the fitted line, and $Cx + Ey + F/2 = 0$ refers to the simulated slot boundary medial axis.

The vehicle pose with respect to each parking slot boundary can be calculated by making use of the fitted lines and the camera intrinsic and extrinsic transformation matrix.

The implementation of the Kalman filter here imposes tempo continuity on the slot boundary detection and makes the system less sensitive to measurement noise. Suhr and Jung [27] also used the predicted image to fuse with actual images to improve the tracking accuracy. However, the predicted image was directly estimated from vehicle odometry sensors without any filtering mechanisms; thus, the results may be sensitive to noise.

V. System Integration

In this section, an overall picture of the algorithm is illustrated to demonstrate how each individual module interacts with each other.

When the driver stops the car at any pose near the intended car park slot, the fully automated reverse parking process can be initiated if part of the parking slot is within the camera view. The steps of the process are listed as follows.

1) Take a picture of the parking slot, and start the image processing to calculate the vehicle pose.
2) Initialize the path generator with the aforementioned information, and start to work out a feasible path. The associated moving velocity and steering angle at each point of the path are generated simultaneously.
3) Initialize the Kalman filter and SMC.
4) Start the iteration by taking a new picture at the new vehicle pose.
5) Derive the new vehicle pose based on image processing and the Kalman filter state prediction.
6) Read the in-vehicle moving speed and steering angle.

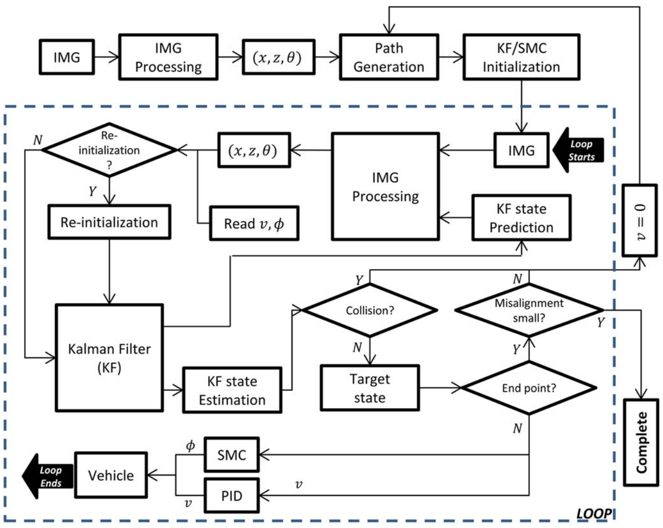


Fig. 9. Flowchart of the whole algorithm.

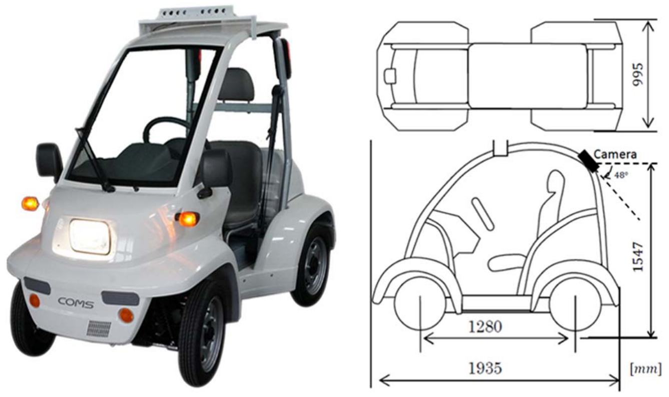


Fig. 10. Test bed: Toyota COMS EV and schematic.

7) Pass the new vehicle pose, speed, and steering angle to the Kalman filter for state estimation and prediction. A check on the reinitialization of the Kalman filter is carried out at the same time.
8) Check collision based on the estimated states. If a collision occurs, stop the vehicle, regenerate a feasible path, and resume from step 3 onwards.
   If not, work out the target position and orientation on the path based on the estimated states.
9) Check whether the vehicle passes beyond the end point. If not, proceed to the next step. If yes, stop the vehicle and further check its alignment.
   If its misalignment is small (within $\pm 2^\circ$ and ± 7 cm from the Z-axis), the parking is completed. If not, regenerate a feasible path, and resume from step 3.
10) Generate the error signals, and pass them to SMC if there is no collision and the vehicle is not beyond the end point.
11) Work out a steering angle command through SMC, and send it to the vehicle steering motor.
12) Send the target speed to an independent PID controller.
13) Repeat steps 4–13 until the parking is completed.

The flowchart of the whole algorithm is illustrated in Fig. 9. The Kalman filter reinitialization prevents the initial error,

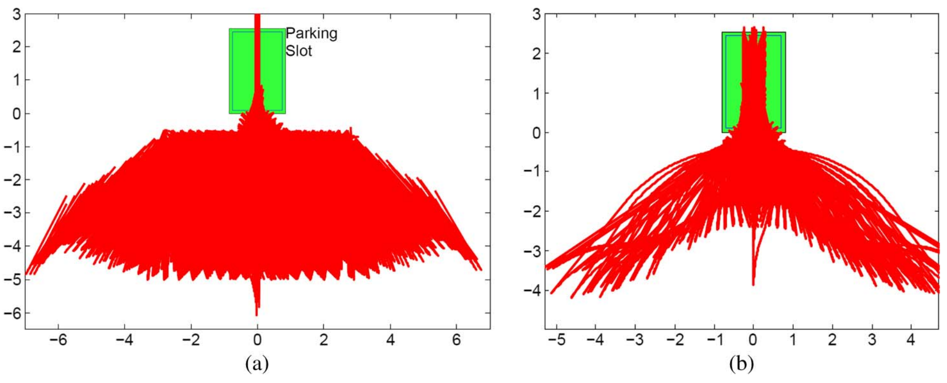


Fig. 11. Path distribution for the testing vehicle with $R = 3.6$ m.

which is induced from the difference between the preassumed and actual parking slot sizes, from inflating as the iteration continues. It also makes the system adaptive to different parking slot sizes.

The entire scheme guarantees a collision-free parking process since it checks collision at every iteration. The final parking accuracy is imposed as well by misalignment checking. The path-planning step is only carried out at the beginning and when a collision or end-point misalignment occurs. The robust path planning frees the whole process from human intervention as there are no dead spots that require the human driver to adjust the car pose in order to generate a feasible path.

VI. Experimental Results

To validate the proposed algorithm, we implemented it on a Toyota COMS electrical vehicle (EV) (see Fig. 10). The camera is mounted at the center of the vehicle rear roof, pointing downwards at an angle of 48° from the horizon and with a height of 1.55 m from the ground. Other related dimensions are shown in the schematic [see Fig. 10(b)]. Note that the minimum turning radius measured from the control point is about 3.5 m, which is large compared with the vehicle length. This means that, under certain poses, a larger free space and a longer path are needed as it is not able to make sharp turns.

All the function modules aforementioned are realized by MATLAB running in the onboard PC with an Intel Core i5 CPU. The communication between the PC and the EV driving and steering modules is conducted by a low-level control PC running Linux with an i586 CPU. The average time taken for one iteration is about 0.18 s, which is satisfactory for this low-speed application. Eighty-five percent of the time is spent on image processing.

A. Results on Path Generation

To validate the robustness of the path generation, the following simulation is designed: For the space in front of the parking slot, assume that vehicle position X varies from -2.8 to 2.8 m with an interval of 0.2 m and that Z varies from -0.5 to -2.5 m with an interval of 0.2 m as well. For each position (X, Z), assume that the vehicle orientation varies from 0 to $-\pi$ rad with an interval of 0.1 rad. Based on this meshgrid, we generate 10 208 vehicle poses to test.

Fig. 11(a) shows the distribution of the path at each pose generated by the proposed algorithm with turning radius $R = 3.6$ m. The green box refers to the parking slot. At each of the 10 208 poses, at least one feasible path can be generated. The free space required varies from -7 to 7 m for X and up to -6 m for Z. The widespread distribution is mainly caused by the poses that are close to the horizontal direction and far away from the parking center line. If the turning radius can be reduced, the free space required can be shrunk accordingly.

When the vehicle is inside the parking slot, similar grids are generated. The path distribution is shown in Fig. 11(b). For the poses where the vehicle is at a collision status, the paths are not shown. An example is when the vehicle is near the left or right boundary; however, it seldom ends up at those poses as the collision check is carried out in every control cycle. The widespread wings are due to the poses around the parking slot's bottom boundary where orientations near the horizontal are allowed.

Since the test bed is smaller than normal sedans, we carry out further simulations to verify the path generation algorithms under a normal sedan size and a normal parking slot size. The sedan size is based on Toyota Corolla [28], which is 1.8 m wide and 4.6 m long with a minimum turning radius of 5.4 m. The size for a standard parking slot is 2.4 m $\times$ 4.8 m [29]. The testing area is enlarged accordingly. The range in the X-direction is from -5 to 5 m, and Z is from -1 to -5 m.

For all the simulated poses, we obtained the path distribution, as shown in Fig. 12. Again, we confirmed that, for a normal sedan, at least one feasible path is available at any given vehicle pose, even when it is inside the parking slot [see Fig. 12(b)].

However, as compared with the test EV, the free space required is larger for a normal sedan since both its size and turning radius R are larger. For the same reason, under certain poses, a normal sedan requires paths with more segments. These paths are harder to follow since more maneuvers are required, and they also increase the computation time for the path generation if they have more than four segments.

To further evaluate the proposed algorithm, we also compared it with other two state-of-the-art algorithms proposed in [30] and [31] that also try to generate a feasible path regardless of vehicle initial poses.

In [30], the path only consists of tangent arcs except for the last segment that is a straight line (we refer to this as

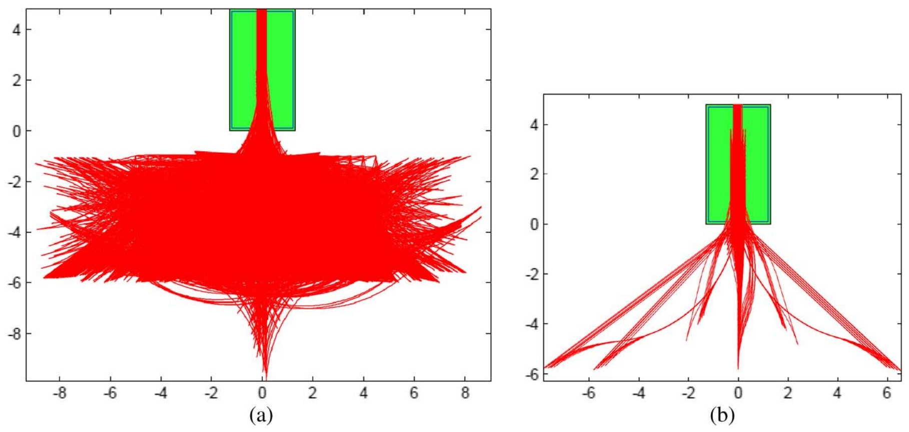


Fig. 12. Path distribution for a normal sedan with $R = 5.4$ m.

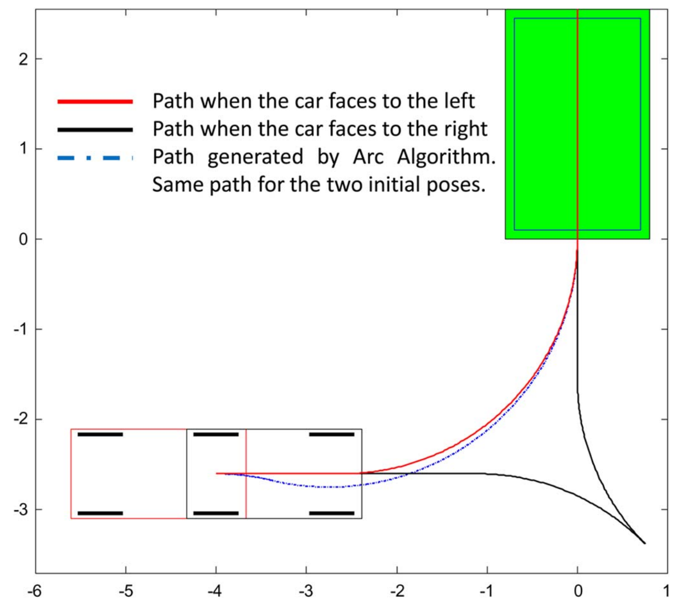


Fig. 13. Nonuniqueness of the final orientation by the arc algorithm.

the arc algorithm). One of its disadvantages is that the final parking orientation is not unique. For example, in Fig. 13, if the initial pose faces to the left (the red vehicle), the final vehicle orientation faces outwards. If the initial pose faces to the right, it generates the same path (blue dot-dash line) and ends up with an inwards-faced orientation. If uniqueness is to be preserved, there is no feasible path under some poses. However, our algorithm guarantees an outwards-faced final orientation.

Moreover, no path is available when the vehicle initial pose is inside the slot. This situation happens when the vehicle deviates from the planned path due to some disturbances and ends up in collision with the adjacent slot. The path needs to be regenerated, and the arc algorithm will fail.

For the algorithm proposed in [31] [the Korea University Path Planner (KPP)], it generates a multifolded path, of which each section consists of translation, followed by rotation (the minimum radius) and another translation. It is also able to generate a feasible path under any vehicle initial poses. Fig. 14 illustrates the comparison between the paths generated by the KPP algorithm and our algorithm for three initial poses. The three poses were used to compare the KPP with another algorithm, i.e., the rapidly exploring random tree (RRT) algorithm, in [32]. For a fair comparison, we adopted all the dimensions in that paper.

Table II shows the comparison of the path length, the number of maneuvers, and the computing time to generate the path. Both algorithms generate similar paths in terms of the path length and no clear advantages of one algorithm over the other. However, due to the multifolded property of the KPP, it tends to generate paths with more maneuvers, which is more difficult to follow. Moreover, our algorithm is able to generate a feasible path much faster than the KPP. The average time of generating one path shown in Fig. 11 is 0.098 s.

Therefore, the proposed algorithm outperforms the KPP in the domain of reverse parking. However, under a cluttered environment, as shown in [31], a multifolded approach is preferred at the expense of the computing time. Note that our algorithm can be extended to a multifolded version by alternatively adding straight lines and minimum turning arcs at the beginning of the proposed algorithm.

In summary, we have validated the robustness of the path generation algorithm through simulation and verified that a feasible path is guaranteed under any initial poses.

B. Results on SMC

Fig. 15 illustrates how the vehicle position error d_e and orientation error θ_e vary from frame to frame during one typical trial. The errors are calculated based on (35) and (36). The left y-axis and the blue solid line are for θ_e, whereas the right y-axis and the red dotted line are for d_e. Both values vary within a small range, around 0. Frames 65–75 show the fast response from the controller to bring large deviations back to 0. The revised SMC is capable to control the vehicle to follow the planned path with very small position and orientation errors.

To validate the improvement in the human driver's comfort level, the proposed path following controller was compared with the conventional SMC in [24] and the fuzzy logic control. Three sets of on-field tests using these three controllers were carried out. Each set consists of 50 trials with random initial poses.

Similar to SMC, the fuzzy logic takes d_e and θ_e as the input and φ as the output. The membership functions are illustrated

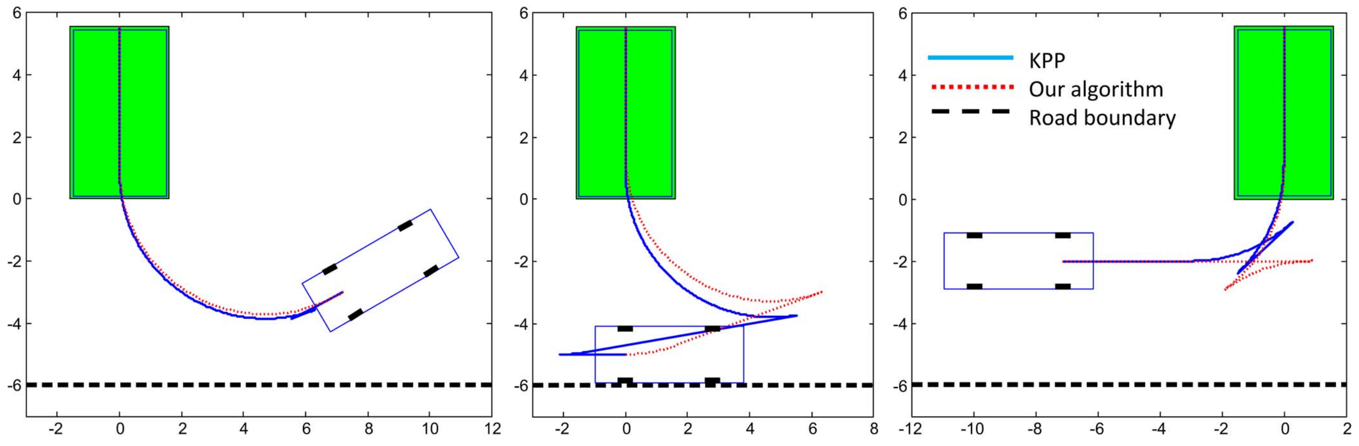


Fig. 14. Path generation comparison between the KPP algorithm and our proposed algorithm.

TABLE II
COMPARISON BETWEEN THE KPP AND THE PROPOSED ALGORITHM

Initial Pose	(1)		(2)		(3)	
Algorithm	Ours	KPP	Ours	KPP	Ours	KPP
Length(m)	14.6	15.5	19.8	22.64	20.11	18.56
Maneuvers	2	4	3	4	3	4
Com. time(s)	0.132	0.875	0.138	0.812	0.111	0.813

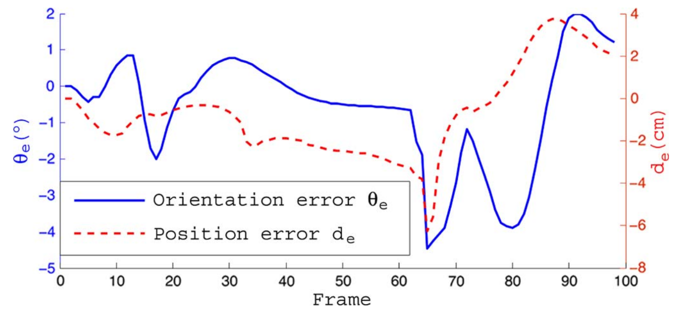


Fig. 15. Position and orientation error trend during one trial.

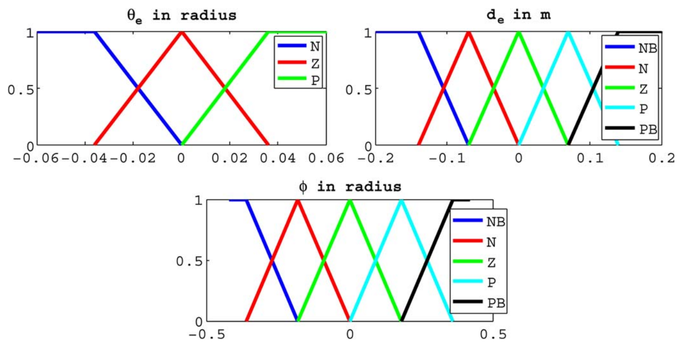


Fig. 16. Fuzzy logic membership function design.

in Fig. 16. Based on the vehicle moving direction, 30 rules are designed in total. (NB stands for negative big, N for negative, Z for zero, P for positive, and PB for positive big.)

To measure the comfort level, the following indexes are used.

1) Unexpected steering swings (USSs). One steering swing (SS) is defined as turning the steering wheel from the center or zero position to its left or right limit. The number of expected SSs is counted from the generated path. The difference between the expected SS and the SS counted from the SMC output is the USS. Ideally, the USS equals to 0. The smaller the USS is, the smoother the controller is.

2) Jerk [33]. It is the time derivative of the acceleration. Both vehicle motion J_v and steering wheel turning J_φ induce jerk to the system. J_v can be decomposed into two components, i.e., longitudinal J_L and lateral J_T, as shown in (45) and (46). Both the J_v and J_φ average absolute values based on all the measurements in one trial are used as indexes to indicate the jerking effect of that trial, as shown in

$$J_L = \dot{a_L} = \frac{d(\dot{\theta}v)}{dt} = \frac{d\left(\frac{\tan(\varphi)v^2}{l}\right)}{dt} = \frac{\left(v^2\sec^2(\varphi)\dot{\varphi} + 2\tan(\varphi)v\dot{v}\right)}{l} \quad (45)$$

$$J_T = \dot{a_T} = \frac{d(\dot{v})}{dt} = \ddot{v} \quad (46)$$

$$J_v = \sqrt{J_L^2 + J_T^2} \quad (47)$$

$$J_\varphi = \frac{d^3(\varphi)}{dt^3}. \quad (48)$$

TABLE III
COMPARISON BETWEEN THE PROPOSED SMC, THE CONVENTIONAL SMC, AND FUZZY LOGIC

Controller	USS	$J_v(ms^{-3})$	$J_\phi(s^{-3})$
Proposed SMC	1.02	0.985	8.218
Conventional SMC	3.06	1.669	20.812
Fuzzy Logic	0.03	1.212	10.850

Table III shows the comparison between the proposed SMC, the conventional SMC, and the fuzzy logic performance. The data presented are mean values over the 50 trials for each controller. As compared with the conventional SMC, since there are less unexpected SSs, the proposed controller output is more capable of following the required steering profile while making necessary minor adjustments. The induced jerking effects can be suppressed significantly, particularly the jerks from the steering wheel.

As compared with the fuzzy logic control, although the proposed SMC yields a higher USS due to the fast transient response, it is still able to reduce the jerking effects in both J_v and J_φ and provide smoother control actions. Moreover, the

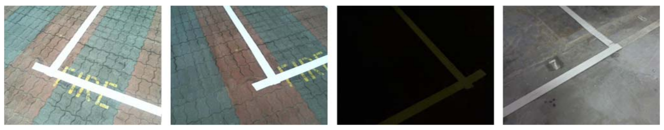


Fig. 17. Illumination during daytime, sunset, night, and indoor.

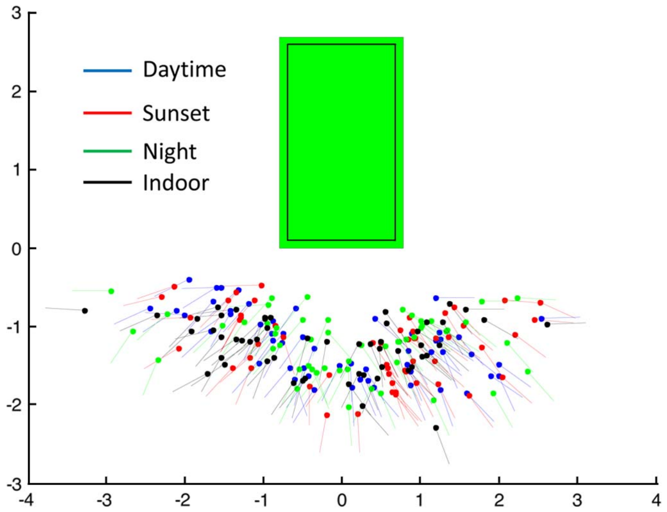


Fig. 18. Initial pose distribution for all the trials.

tuning of the proposed SMC is much simpler than the logic control since SMC only involves two parameters whereas the fuzzy logic control involves more than ten.

From all the experiments aforementioned, we can conclude that the revised SMC is able to generate a smooth control output and eliminate jerking to a large extent. It results in a higher level of the human driver's comfort.

To apply the revised SMC to a normal sedan, which is larger than the test EV, the related parameters Q' and k' in (42) need to be retuned accordingly. As shown in (33), a longer vehicle is slower in adjusting its orientation θ given the same speed and steering angle. Therefore, to maintain a fast response, Q' and k' need to be adjusted so that the target steering angle φ from the SMC can be larger.

C. Results on Final Parking Accuracy

Extensive experiments were conducted under different illumination situations to evaluate the robustness of the proposed algorithm. The situations include an indoor car park and an open-air car park during daytime, sunset, and nighttime. Sample images are shown in Fig. 17 to demonstrate illumination intensities. During daytime, the sunshine is strongest, and the visibility is the best. However, some pixels may be saturated. During sunset, the sunshine and the visibility are moderate. During night, the visibility is very poor, and the light source is the vehicle taillights. In the indoor car park, the visibility is also moderate, and a fluorescent lamp is the light source.

Fifty-four trials were carried out under each situation with a variety of initial poses (see Fig. 18) to better reflect real parking exercises. The parking accuracy is measured based on two indexes, i.e., the vehicle offset distance from the control point to the slot center line and its orientation deviation. Fig. 19 plots the absolute offset and the orientation error for each trial under different illumination conditions.

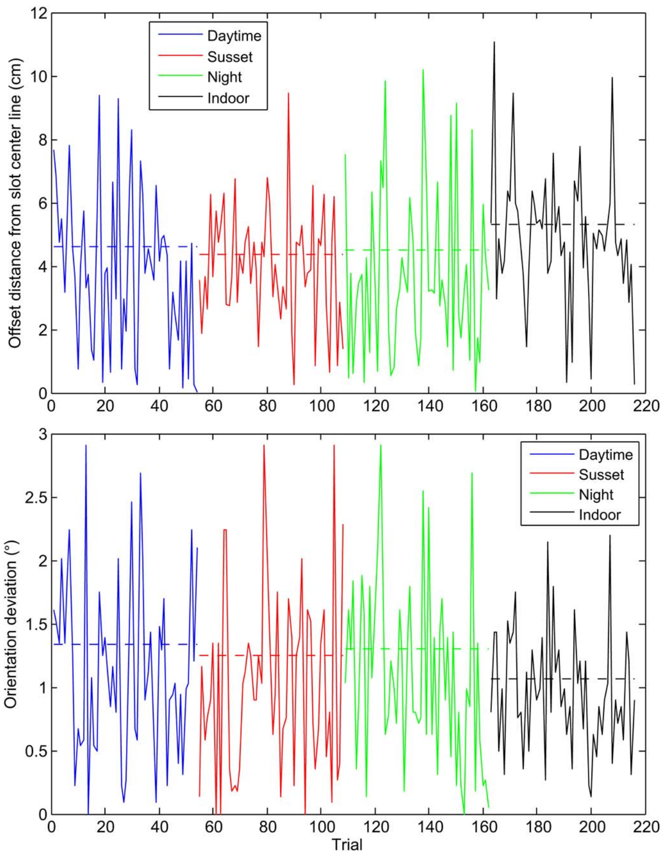


Fig. 19. Absolute offset distance and orientation deviation for each trial (the dashed line represents the RMS).

TABLE IV
OFFSET AND ORIENTATION RMS AND STD

		Daytime	Sunset	Night	Indoor	Overall
Offset (*cm*)	RMS	4.60	4.36	4.50	5.31	4.71
	STD	2.50	1.81	2.64	2.09	2.31
Orientation Deviation (°)	RMS	1.34	1.25	1.30	1.07	1.24
	STD	0.70	0.73	0.71	0.47	0.66

From both the offset and orientation error plots, the accuracy is similar under different illumination conditions. There is no clear advantage of one condition over the others. The same observation is made on the RMS values in Table IV. This infers that the proposed vision approach is robust to illumination variation, even when the image quality taken during the night is poor. The overall RMS distance offset is 4.71 cm, and the orientation deviation is 1.24°.

The overall standard deviation (STD) for the offset is 2.31 cm and 0.66° for the orientation deviation. This shows that the proposed approach is capable of consistently providing satisfactory parking accuracy under environment changes. The approach is therefore reliable.

It is important to note that the results obtained are subject to other disturbances, including inaccurate conversion from the steering motor turning angle to the vehicle turning radius, unmodeled slippage between ground and wheels and between wheels and driving motors, and deficits in camera calibration. All these factors can lead to inaccuracy in the Kalman filter. However, the result shows that the algorithm handles these disturbances well.

The parking accuracy was also compared with manual parking results. Except for the autonomous COMS aforementioned, there were eight more manual COMS shared among the National University of Singapore staff. The sharing scheme has been carried out for 1.5 years, and the COMS users are considered experienced. The parking accuracy was measured at the end of the day after the users returned all the COMS to the indoor center hub. Fifty sets of data were collected. The RMS distance offset is 8.12 cm, and the orientation deviation is 1.73°. This shows that the proposed autonomous parking approach can achieve slightly better accuracy than experienced human drivers.

Under the case of a normal sedan, both the offset and orientation errors could be potentially enlarged as compared with the data illustrated in Fig. 19. However, the misalignment checking process in Section V is expected to be able to readjust the vehicle to park within acceptable offset and orientation errors. The only problem is that it may take a longer time to park a normal sedan than the test EV as more rounds of readjustment might be required.

D. Discussions and Future Work

The current system relies on human drivers to identify an empty parking slot first before initiating the automatic parking process. This requirement may be removed using ultrasonic sensors fused with the vision system to detect empty parking slots automatically.

In addition, the current vision-only system is not able to identify the exact forbidden areas (see Fig. 6), and this may result in unnecessarily longer paths and false collision alarms. Currently, these areas are defined by the parking slot's left and right boundaries. A more efficient alternate approach is to define them by the side facets of adjacent vehicles, which also requires ultrasonic sensors to be in place.

VII. Conclusion

In this paper, a full set of solutions for autonomous reverse parking based on camera vision has been proposed. The low-cost system is proven to be practical and reliable through extensive on-field experiments under different illumination conditions.

The implementation of a ridge detector and a Kalman filter ensures the accuracy and consistency of vehicle pose estimation, whereas the revised SMC takes the human driver's comfort into account. The robust overall control scheme makes sure that there is no collision with adjacent parking slots and that the vehicle accurately parks along the slot center line.

The novel path planing algorithm guarantees a feasible path at any given pose. This is the key function to enable the fully autonomous feature because it does not require the human driver's intervention to readjust the vehicle to a certain pose to generate a feasible path.

The proposed approach tackles the autonomous reverse parking problem from practical and general perspectives, which ensures its feasibility for other vehicles besides the vehicle utilized in this paper. It can be also incorporated with other sensing systems to further expand its functionality and capability.

References

[1] The Hybrid That Started it All, Mar. 2014. [Online]. Available: http://www.toyota.com/prius/#!/Welcome

[2] BMW 7 Series—Park Assist, Mar. 2013. [Online]. Available: http://www.bmw.com/com/en/newvehicles/7series/sedan/2012/showroom/driver_assistance/park-assistant.html#t=l

[3] C. Heilenkötter *et al.*, "The consistent use of engineering methods and tools," *AutoTechnology*, vol. 6, no. 6, pp. 52–55, Nov. 2006.

[4] J. Pohl, M. Sethsson, P. Degerman, and J. Larsson, "A semi-automated parallel parking system for passenger cars," *Proc. Inst. Mech. Eng. D, J. Autom. Eng.*, vol. 220, no. 1, pp. 53–65, Jan. 2006.

[5] H. G. Jung, Y. H. Cho, P. J. Yoon, and J. Kim, "Scanning laser radar-based target position designation for parking aid system," *IEEE Trans. Intell. Transp. Syst.*, vol. 9, no. 3, pp. 406–424, Sep. 2008.

[6] P. Degerman, J. Pohl, and M. Sethson, "Ultrasonic sensor modeling for automatic parallel parking systems in passenger cars," presented at the SAE World Congress, Detroit, MI, USA, 2007, Paper 2007-01-1103.

[7] M. Wada, K. S. Yoon, and H. Hashimoto, "Development of advanced parking assistance system," *IEEE Trans. Ind. Electron.*, vol. 50, no. 1, pp. 4–17, Feb. 2003.

[8] G. Yan, W. Yang, D. B. Rawat, and S. Olariu, "Smartparking: A secure and intelligent parking system," *IEEE Intell. Transp. Syst. Mag.*, vol. 3, no. 1, pp. 18–30, 2011.

[9] K. Fintzel, R. Bendahan, C. Vestri, S. Bougnoux, and T. Kakinami, "3D parking assistant system," in *Proc. IEEE Intell. Veh. Symp.*, 2004, pp. 881–886.

[10] N. Kaempchen, U. Franke, and R. Ott, "Stereo vision based pose estimation of parking lots using 3d vehicle models," in *Proc. IEEE Intell. Veh. Symp.*, 2002, vol. 2, pp. 459–464.

[11] C. Wang *et al.*, "Automatic parking based on a bird's eye view vision system," *Adv. Mech. Eng.*, vol. 2014, pp. 847 406-1–847 406-13, 2014.

[12] H. G. Jung, D. S. Kim, and J. Kim, "Light-stripe-projection-based target position designation for intelligent parking-assist system," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 4, pp. 942–953, Dec. 2010.

[13] A. Löpez, J. Serrat, C. Canero, F. Lumbreras, and T. Graf, "Robust lane markings detection and road geometry computation," *Int. J. Autom. Technol.*, vol. 11, no. 3, pp. 395–407, Jun. 2010.

[14] L. E. Dubins, "On curves of minimal length with a constraint on average curvature, with prescribed initial and terminal positions and tangents," *Amer. J. Math.*, vol. 79, no. 3, pp. 497–516, Jul. 1957.

[15] J. Reeds and L. Shepp, "Optimal paths for a car that goes both forwards and backwards," *Pac. J. Math.*, vol. 145, no. 2, pp. 367–393, 1990.

[16] D. Wang, H. Liang, T. Mei, and H. Zhu, "Research on self-parking path planning algorithms," in *Proc. IEEE ICVES*, 2011, pp. 258–262.

[17] T.-H. Li and S.-J. Chang, "Autonomous fuzzy parking control of a car-like mobile robot," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 33, no. 4, pp. 451–465, Jul. 2003.

[18] M. B. Oetiker, G. P. Baker, and L. Guzzella, "A navigation-field-based semi-autonomous nonholonomic vehicle-parking assistant," *IEEE Trans. Veh. Technol.*, vol. 58, no. 3, pp. 1106–1118, Mar. 2009.

[19] K. Macek, R. Philippsen, and R. Siegwart, "Path following for autonomous vehicle navigation with inherent safety and dynamics margin," in *Proc. IEEE Intell. Veh. Symp.*, 2008, pp. 108–113.

[20] B. d'Andréa Novel, G. Campion, and G. Bastin, "Control of nonholonomic wheeled mobile robots by state feedback linearization," *Int. J. Robot. Res.*, vol. 14, no. 6, pp. 543–559, Dec. 1995.

[21] M. Khoshnejad and K. Demirli, "Autonomous parallel parking of a car-like mobile robot by a neuro-fuzzy behavior-based controller," in *Proc. Annu. Meet. NAFIPS*, 2005, pp. 814–819.

[22] K. Oyama and K. Nonaka, "Model predictive parking control for nonholonomic vehicles using time-state control form," in *Proc. ECC*, 2013, pp. 458–465.

[23] W. Dong and K.-D. Kuhnert, "Robust adaptive control of nonholonomic mobile robot with parameter and nonparameter uncertainties," *IEEE Trans. Robot.*, vol. 21, no. 2, pp. 261–266, Apr. 2005.

[24] R. Solea and U. Nunes, "Trajectory planning with velocity planner for fully-automated passenger vehicles," in *Proc. IEEE ITSC*, 2006, pp. 474–480.

[25] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Trans. ASME, J. Basic Eng.*, vol. 82, no. 1, pp. 35–45, Mar. 1960.

[26] H. K. Khalil, *Nonlinear Systems*, vol. 3. Upper Saddle River, NJ, USA: Prentice-Hall, 2002.

[27] J. K. Suhr and H. G. Jung, "Sensor fusion-based vacant parking slot detection and tracking," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 1, pp. 21–36, Feb. 2014.

[28] Elevate Your Drive, Mar. 2014. [Online]. Available: http://www.toyota.com/corolla/#!/Welcome

[29] *Code of Practice on Vehicle Parking Provision in Development Proposals*, Land Transport Authority (LTA) Singapore, Singapore, Mar. 2011.

[30] M. F. Hsieh and U. Ozguner, "A parking algorithm for an autonomous vehicle," in *Proc. IEEE Intell. Veh. Symp.*, 2008, pp. 1155–1160.

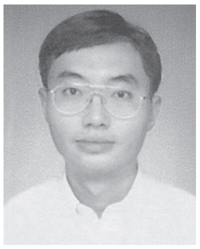
[31] D. Kim, W. Chung, and S. Park, "Practical motion planning for car-parking control in narrow environment," *IET Control Theory Appl.*, vol. 4, no. 1, pp. 129–139, Jan. 2010.

[32] H. Kwon and W. Chung, "Comparative analysis of path planners for a car-like mobile robot in a cluttered environment," in *Proc. IEEE Int. Conf. ROBIO*, 2011, pp. 602–607.

[33] J. C. Sprott, *Chaos and Time-Series Analysis*, vol. 69. Oxford, U.K.: Oxford Univ. Press, 2003.

Xinxin Du received the B.Eng. degree from Nanyang Technological University, Singapore, in 2011. He is currently working toward the Ph.D. degree in the Department of Electrical and Computer Engineering, Faculty of Engineering, National University of Singapore, Singapore.

From August 2011 to January 2013 he was an Industrial Engineer with Micron Semiconductor Asia Private Ltd., Singapore, working on production forecasting and system optimization. His research interests include autonomous electrical vehicles, image processing, and motion control.



Kok Kiong Tan (M'04) received the Ph.D. degree from National University of Singapore (NUS), Singapore, in 1995.

Prior to joining NUS, he was a Research Fellow with the Singapore Institute of Manufacturing Technology (SIMTech), which is a national research and development (R&D) institute spearheading the promotion of R&D in local manufacturing industries, where he was involved in managing industrial projects in information technology and automation. He is currently an Associate Professor with the Department of Electrical and Computer Engineering, Faculty of Engineering, NUS. His research interests include Internet applications, remote control and monitoring, precision motion control and instrumentation, advanced process control and autotuning, and general industrial automation.