applied sciences

MDPI

Article

Around-View-Monitoring-Based Automatic Parking System Using Parking Line Detection

Yunhee Lee [1] and Manbok Park [2,*]

[1] Litbig, Seongnam-Si 13487, Korea; leeyunis@gmail.com
[2] Department of Electronic Engineering, College of Convergence Technology, Korea National University of Transportation, Chungju-Si 27469, Korea
* Correspondence: ohnnuri@ut.ac.kr; Tel.: +82-43-841-5369

Abstract: This paper introduces an automatic parking method using an around view monitoring system. In this method, parking lines are extracted from the camera images, and a route to a targeted parking slot is created. The vehicle then tracks this route to park. The proposed method extracts lines from images using a line filter and a Hough transform, and it uses a convolutional neural network to robustly extract parking lines from the environment. In addition, a parking path consisting of curved and straight sections is created and used to control the vehicle. Perpendicular, angle, and parallel parking paths can be created; however, parking control is applied according to the shape of each parking slot. The results of our experiments confirm that the proposed method has an average offset of 10.3 cm and an average heading angle error of 0.94°.

Keywords: around view monitoring (AVM) system; parking line detection; automatic parking system; vehicle control; deep learning; convolutional neural network

Citation: Lee, Y.; Park, M. Around-View-Monitoring-Based Automatic Parking System Using Parking Line Detection. *Appl. Sci.* **2021**, *11*, 11905. https://doi.org/10.3390/app112411905

Academic Editor: Juan-Carlos Cano

Received: 8 November 2021
Accepted: 9 December 2021
Published: 14 December 2021

1. Introduction

Recently, as the number of vehicles has increased, the automobile industry has expanded its research and development; for instance, driver assistance systems are being studied intensively. Driver assistance systems consist of lane departure warning, forward collision warning, pedestrian protection, and parking assist systems [1,2]. The parking assist system assists the driver in parking situations so that they can park easily. These systems have continuously increased in sophistication from collision warning systems using an ultrasonic sensor [3,4] to systems that show a rear image to the driver using a rear camera, systems that provide a 360°-view using an around view monitoring (AVM) camera [5–7], and fully automatic parking systems [5].

This paper presents an automatic parking system that uses AVM images in a driver assistance system. An AVM system consists of cameras on the front, rear, left, and right sides of the vehicle and shows a 360°-camera view of the vehicle's environment. In particular, it shows the driver a synthesized image of the top view of the vehicle. Because the AVM system provides a top view, it can be used for various applications involving image recognition such as lane detection, moving object detection, and parking line detection. These systems can use the original images from the cameras, but such an approach requires extensive computation time because four images must be processed. By contrast, if a synthesized top view image is used, the processing time of only one image is required. Therefore, in this paper, we propose an automatic parking method that detects parking lines and performs parking control using a top view image.

Parking line recognition is an important process in an automatic parking system. Using the detected parking lines, a system can be configured to automatically park the car in the area desired by the driver. When parking lines are detected in the top view AVM image, the position of each pixel can be converted into a distance, which is advantageous for automatic parking. In addition, all parking lines in each direction can be detected.

The detected parking lines provide relative coordinates because it is easy to convert pixels into distances from the vehicle. Hence, we use these relative coordinates to set a goal to the desired parking location and perform automatic parking using path planning and tracking methods.

Figure 1 presents an overview of the proposed method.

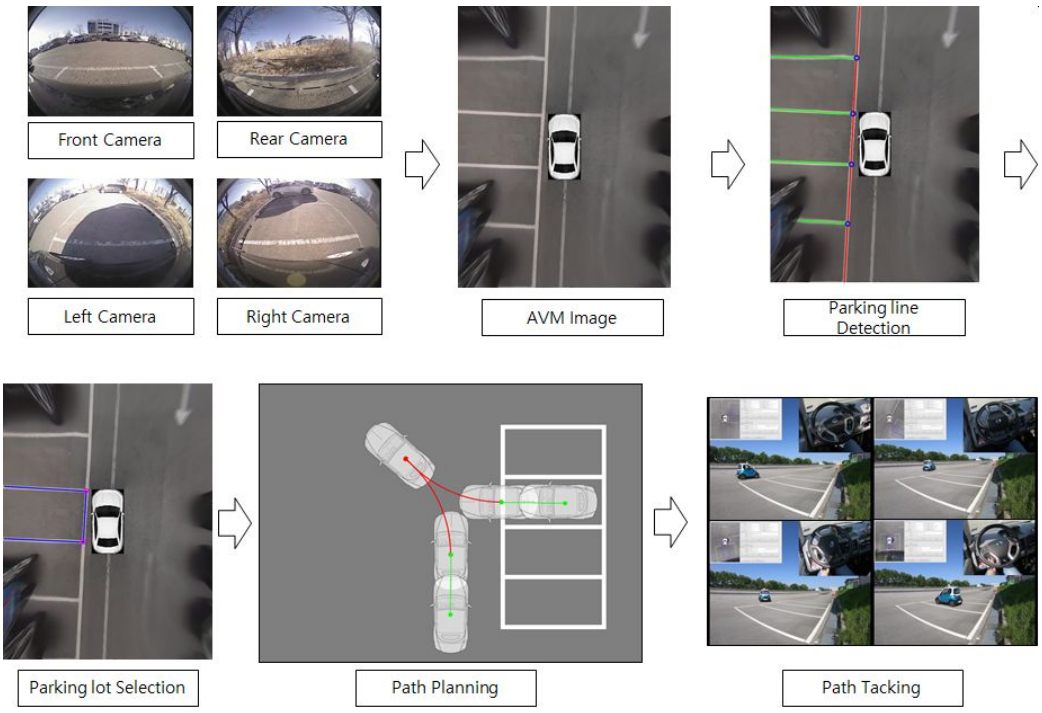


Figure 1. Proposed automatic parking system using AVM images.

As shown in the figure, when an AVM image is input to the system, the parking lines are detected. The proposed system determines the parking area from the detected parking lines. It also determines the relative location of the car to the parking area and the shape of the parking area. Path planning is performed using this information, and automatic parking is performed through path tracking.

The rest of this paper is organized as follows. Related studies on the detection of parking areas and automatic parking control are summarized in Section 2. Section 3 presents the proposed method. Section 4 presents the experimental results, and the paper is concluded in Section 5.

2. Related Works

Automatic parking systems using AVM images can be divided into a parking slot detection part and a path planning and tracking part.

Various studies have been conducted on the detection of parking slots using AVM-based systems. To recognize parking lines, a commonly used method involves locating a corner point through line recognition and determining whether or not that corner is part of a parking dividing line. For instance, parking lines have been detected in AVM images using the Hough transform [8]; a top hat filter and directional DBSCAN [9]; Canny edge detection and the Radon transform [10]; local symmetry of brightness and a Dirac comb [11]; the Sobel filter, RANSAC, and edge pairs [12]; FAST corner detection and RANSAC [13]; and a probabilistic occupancy filter [14]. Moreover, Zhang et al. detected the corners of parking lines and parking slots using gradient values, AdaBoost, and decision trees [15]. Kim et al. recognized parking slots by detecting lines and junctions [16], and parking slots have also been detected using a semantic segmentation filter and a convolutional neural network (CNN) [17]. Zhang et al. recognized parking areas using a deep CNN [18]. Partially

occluded parking lines were recognized by Allodi et al. using segment labeling [19] and by Lee et al. using line features and feature clustering in the top view image [20].

Furthermore, various studies have been conducted on path planning and tracking. Zips et al. defined perpendicular, angle, and parallel parking modes, and created parking paths using a fast motion planning algorithm [21]. Subsequently, they created optimized paths that can be used for narrow spaces [22]. Lin et al. defined perpendicular parking and parallel parking modes and created parking paths using a four-phase algorithm [23]. Sedighi et al. and Kim et al. generated paths for perpendicular parking using a clothoid-based method [24,25]. Vorobieva et al. created a parallel parking path using clothoids and performed parking control using an actual vehicle [26]. Wang created circular trajectories using the parking lines recognized by the AVM system and performed automatic parking with a test vehicle [10]. Li et al. created paths for perpendicular parking and parallel parking using fuzzy control and performed parking control using a mobile robot [27]. Parking paths have been created using collision-free and nonholonomic paths generated by the slice projection technique [28] and a method that considers the surrounding environment and generates an optimal path by dividing the motion of a car into translation and rotation motions [29].

The proposed method uses the Hough transform to detect lines and cross points. Using the Hough transform has the advantage of detecting straight lines and cross points with simple operations. On the other hand, there is a disadvantage that it is difficult to distinguish the parking line from other lines. Therefore, the proposed method uses CNN to distinguish cross points with high accuracy. Furthermore, the proposed method uses circles and lines for path planning. Although this method has the disadvantage of taking a long time to control the vehicle, it has the advantage of being suitable for application to an actual vehicle and controlling the vehicle with high accuracy. Therefore, the proposed method uses a method using a circle and a line that is suitable for high accuracy and real automobile applications.

3. Proposed Method

3.1. Parking Line Detection

We propose a method for recognizing parking lines in AVM images. To detect parking lines, straight lines are extracted using the Hough transform. To check whether a corner on an extracted straight line is a parking line, a CNN, which is a deep learning method, is used to distinguish the corner points of the parking lines and other line components.

3.1.1. Parking Line Candidate Extraction Using the Hough Transform

To find parking lines in an AVM image, it is better to extract the parking lines from the image converted to the top view [7]. Because the converted image provides the viewpoint looking down from above, it is more suitable than the original images from the cameras. In particular, in the top view image, the parking lines are straight lines, and a position in the top view image can be displayed as a relative distance from the vehicle. Therefore, in the proposed method, parking lines are detected from the top view.

In top view images, the parking lines form rectangular shapes. Therefore, in a top view image, if there is a section where a vehicle can be parked, straight lines appear around its edges. To use this feature, parking lines are detected by locating the positions where straight lines intersect the rectangle [16].

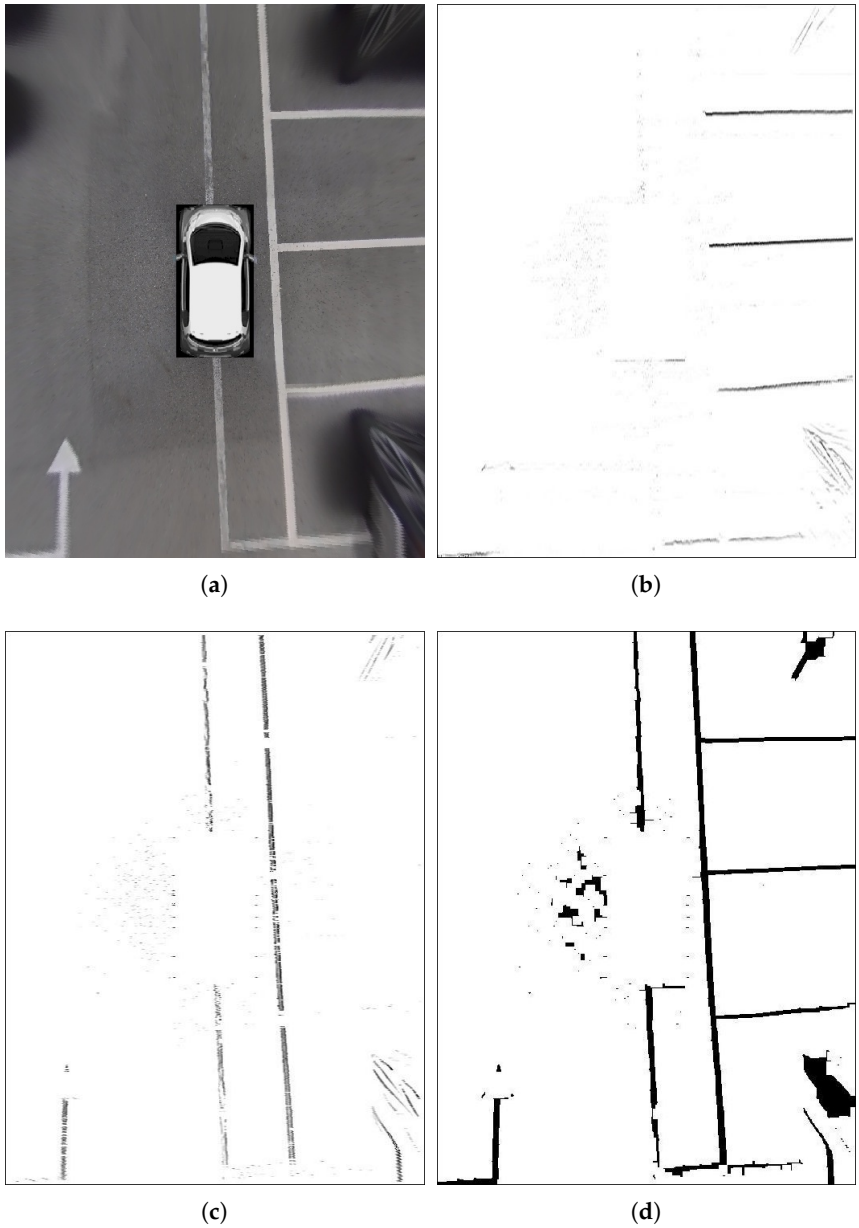
To detect a straight line, the linear components are first extracted from the AVM image using a line filter. This enables robust performance in various environments; for instance, weather and lighting conditions can vary in underground parking slots or outdoors [9].

The mask of the line filter is depicted in Figure 2. This mask shape is used to filter the image in the x and y directions. In addition, because a parking dividing line has a rectangular shape, this filter is able to detect a parking line from any direction.

-1	-1	-1	1	...	1	-1	-1	-1

Figure 2. Line filter.

Lines with a certain width are distinguished by the filter. In the top view image, the width of a parking line is the same regardless of its distance from the vehicle. Therefore, the image can be filtered without changing the width of the line filter according to the distance. Figure 3 shows the result of finding features using a line filter. Figure 3a shows the AVM image; Figure 3b shows the vertical line filter result; Figure 3c shows the horizontal line filter result; and Figure 3d shows the result of combining the vertical and horizontal line filter results.



(**a**) (**b**)

(**c**) (**d**)

Figure 3. Line filter results: (**a**) AVM image, (**b**) vertical line filter results, (**c**) horizontal line filter results, and (**d**) combined results of (**b**,**c**).

Linear components can be distinguished using the features found using the line filter. The proposed method obtains straight lines using the Hough transform, which represents a

straight line using parameters rho and theta through a coordinate system transformation [8] as follows:

$$r = x\cos\theta + y\sin\theta. \quad (1)$$

As shown in Equation (1), (x, y) represents the position of a point in the image, (r, θ) represents the angle formed by a straight line with the distance from the origin. In Equation (1), a line is expressed as a straight-line component having a distance and an angle. Therefore, it is possible to detect the linear components formed by the most points.

Parking lines are straight lines because it helps make parking easier for drivers. Accordingly, there are corners in the parking lines on the left and right, and the parking line can be determined using these corners. They play an important role in finding parking lines. Figure 4 shows an example of the corner detection of an image. Figure 4a shows the Hough transform results and Figure 4b shows the corner detection results.

(**a**) (**b**)

Figure 4. Corner detection example: (**a**) Hough transform result and (**b**) corner detection result.

3.1.2. Parking Line Detection Using a CNN

For each detected parking corner, a CNN is used to determine whether or not it was a corner point of a parking line [18,30]. There are various models of CNNs, and there are many models that perform well, such as AlexNet [30], ZFNet [31], VGGnet [32], and GoogLeNet [33]. Because the proposed method must perform automatic parking in real time, a simple real-time model is best. Therefore, because simple operation is more important than a complex model, a modified form of LeNet [34] was used.

Figure 5 shows the CNN model used in the proposed method. It consists of six convolutional layers and two fully connected layers.

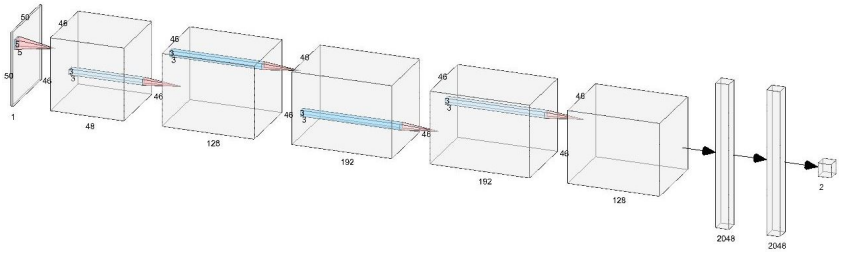


Figure 5. Layers of the CNN.

To train the CNN model, each corner point and non-corner point in the training images was classified and used for training. Figure 6 shows the examples of data used for learning.

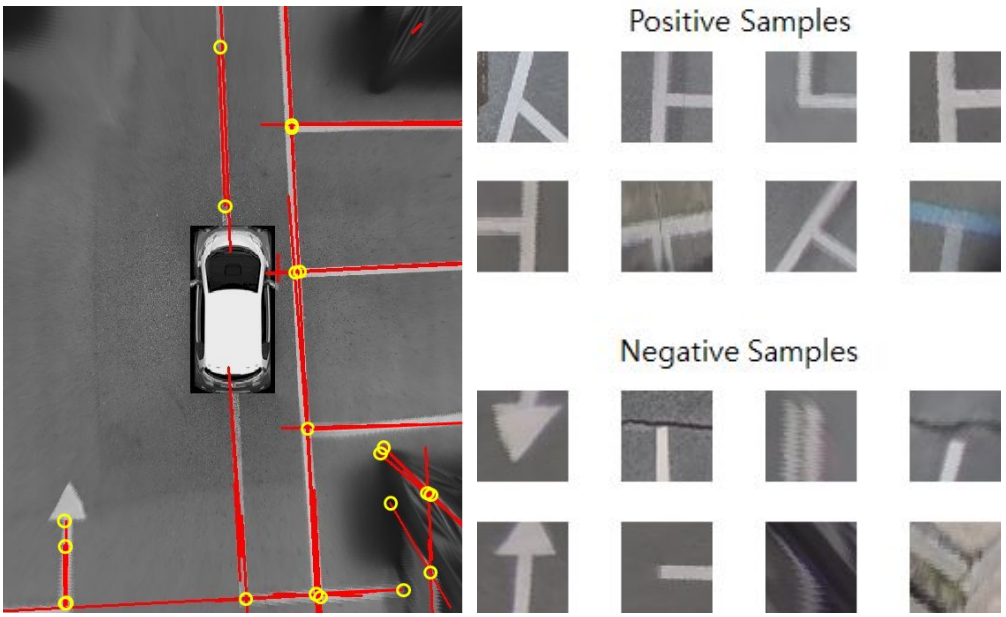


Figure 6. Parking slot corner true and false data examples: (**a**) extraction of corner point candidates, and (**b**) true and false corner point examples.

3.1.3. Parking Slot Tracking

After recognizing a parking slot, the parking location is determined and parking is performed. The parking slot is selected by the driver. Automatic parking is performed through path planning and tracking. During automatic parking, the vehicle moves and inevitably occludes the parking lines. In particular, there are cases where the corners and parking lines, which are important in this study, are not visible. As Figure 7 shows, during parking, the parking line may not be recognized in various circumstances, such as when a parking line or corner point is occluded. This highlights the importance of tracking the parking slot while parking to ensure proper parking control. Therefore, the approach proposed in this paper uses a method to track the parking slot so that the parking location information cannot be lost even while parking is in progress [35].

Figure 7. Examples of parking lines occluded by the vehicle.

Therefore, the proposed method tracks the parking slot and is configured to provide information about the parking slot even when it is not covered or recognized during parking. For this, an algorithm to track the parking slot was employed.

To track the parking area, both corners of the parking area and the distance between them are used. The corners are tracked using a Kalman filter. The state matrix of the Kalman filter is as follows.

$$x = [L_x \; L_y \; R_x \; R_y \; \Delta L_x \; \Delta L_y \; \Delta R_x \; \Delta R_y]. \tag{2}$$

In Equation (2), L_x and L_y are the coordinates of the left corner of the parking area, ΔL_x, ΔL_y represent the derivative at the left corner coordinates, R_x and R_y are the coordinates of the right corner, and ΔR_x and ΔR_y represent the derivative at the right corner. In addition, because the distance between both points is fixed, this condition is used for tracking. When one point is occluded, the location and distance to the other point is used for tracking.

It is important to maintain the information about the parking area for vehicle control during actual automatic parking. Figure 8 shows how this is achieved. As shown, as parking proceeds, the parking line is almost completely occluded, but the parking line is detected because of the tracking algorithm.

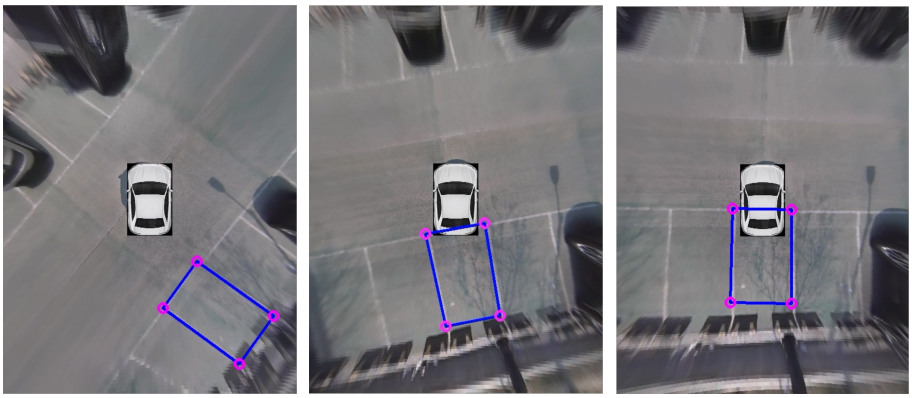


Figure 8. Parking slot tracking example.

3.2. Motion Planning and Tracking

Automatic parking is divided into three types: perpendicular, angle, and parallel parking. In this paper, we consider three types of parking and describe how automatic parking is performed using other methods and the results of parking slot detection.

3.2.1. Motion Planning

Parking motion is divided according to the parking control method, regardless of the type of parking line [21,23].

In this study, the vehicle motion for automatic parking is divided into curved motions and straight motions [10]. Therefore, in the method proposed in this paper, when a parking path is generated, it is divided into curved sections and straight sections. If the path of the car is divided into straight sections and curved sections, the steering angle needs to be maintained at 0° in the straight section and a specific angle is adopted in the curved section to easily control the car. To find the steering angle on a curved path, δ_f is calculated, as shown in Figure 9; here, δ_f is the steering value and L is the wheelbase of the vehicle.

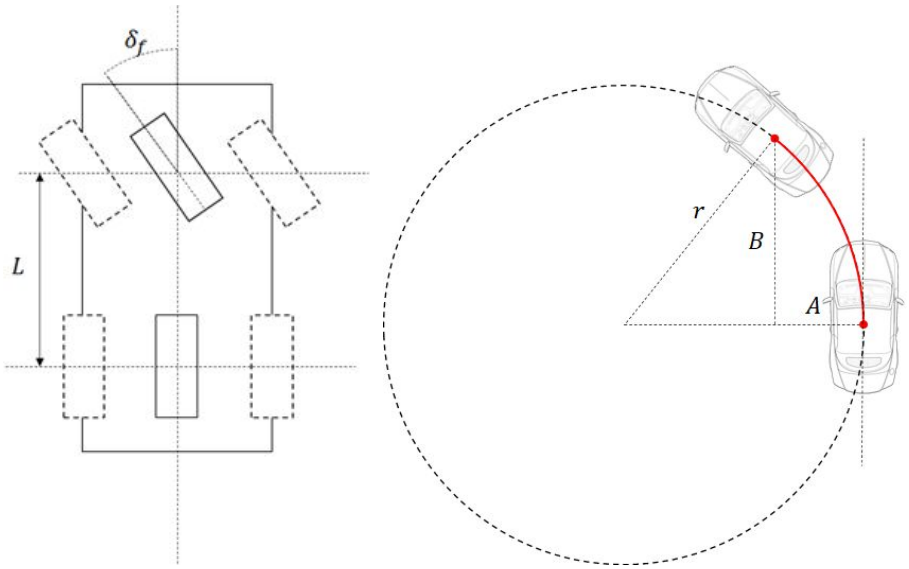


Figure 9. Vehicle parameters for calculating the angle in curved motion.

The steering control value δ_f can be obtained from

$$\delta_f = \tan^{-1}\left(\frac{L}{r}\right), \tag{3}$$

where r is the radius of the circle drawn by the curved path and can be obtained using

$$r = \frac{A^2 + B^2}{2A}. \tag{4}$$

In the proposed method, the paths for perpendicular, angle, and parallel parking are created by separating them into straight and curved sections in different ways [10,21].

First, for perpendicular parking, a parking path is created as shown in Figure 10. The vehicle follows a straight path and then arrives at the target point using a curved path. Then, after changing into reverse, the reverse curve is driven, and finally the parking finishes with reverse straight driving.

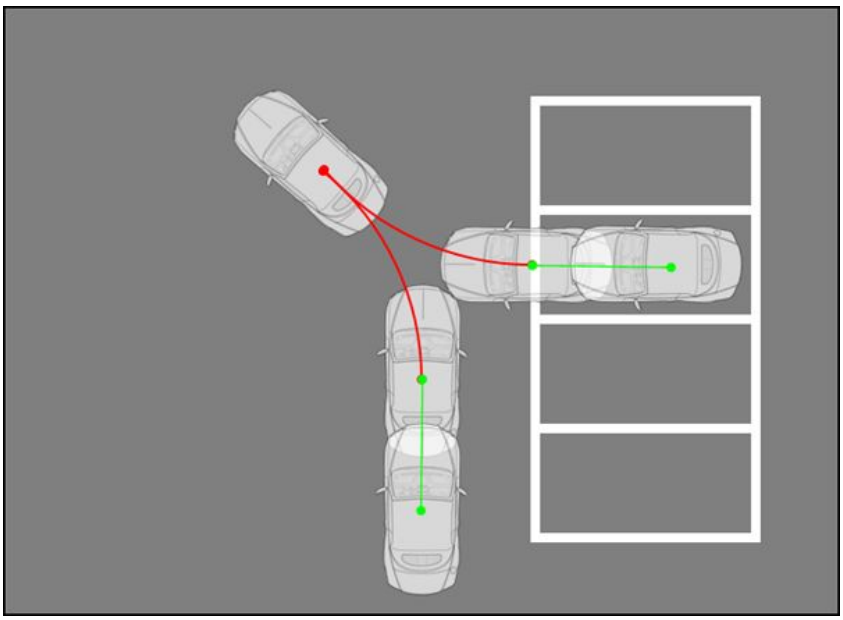
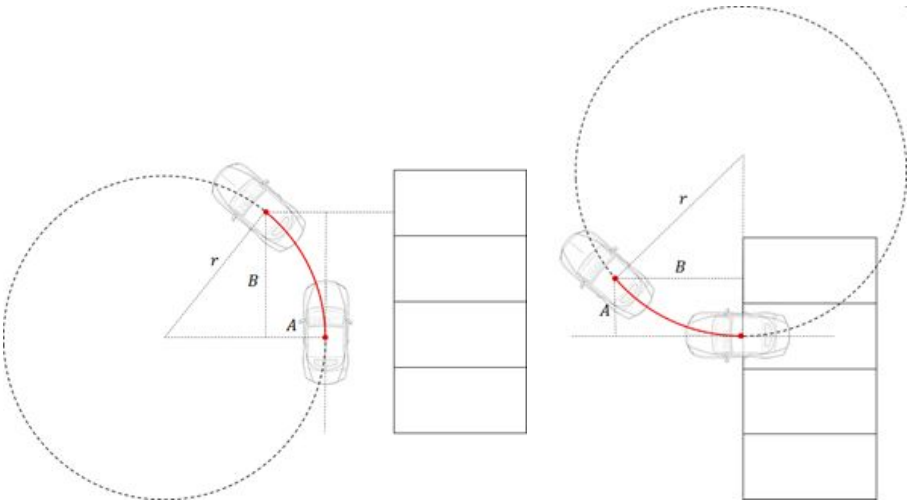


Figure 10. Perpendicular parking trajectory.

The curved sections for this path appear twice. The calculation of the radius r of the two paths is shown in Figure 11 where A, B are used and the steering control value is calculated from r.



(**a**) (**b**)

Figure 11. Calculation of r in the perpendicular parking trajectory: (**a**) calculation of the first circular trajectory and (**b**) calculation of the second circular trajectory.

Parallel parking uses the path shown in Figure 12, where, after going in a straight line and driving backwards in a curve, the vehicle curves in the opposite direction and then drives straight forward to complete the parking.

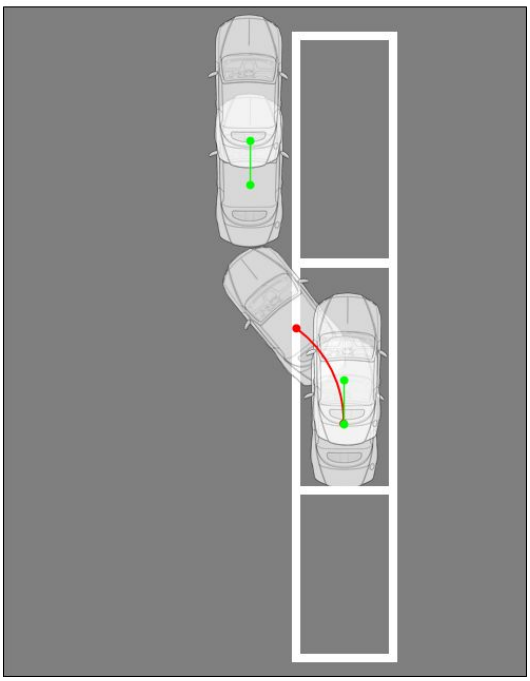
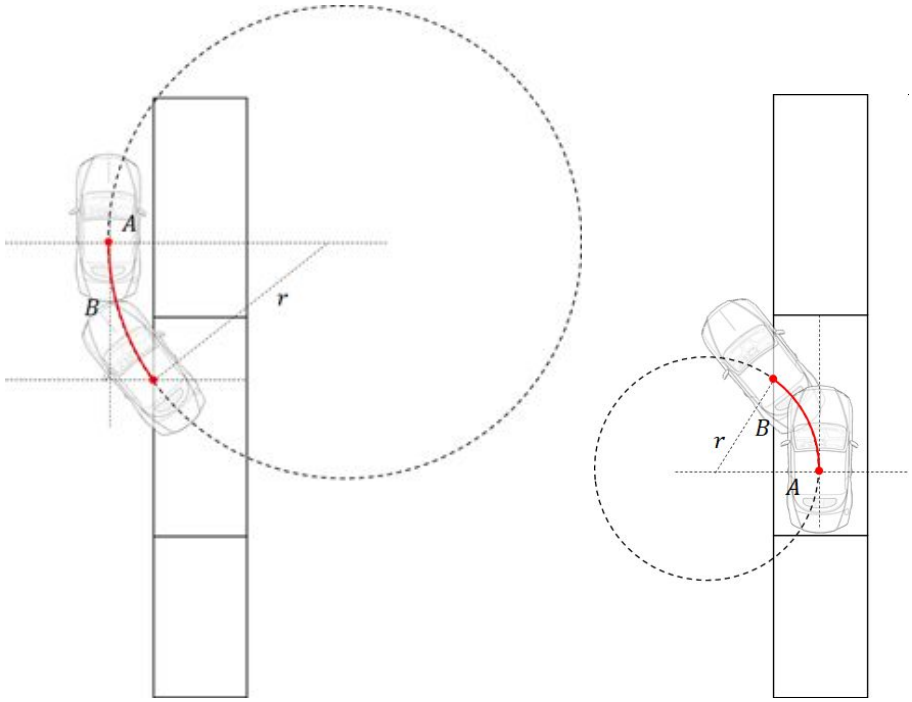


Figure 12. Parallel parking trajectory.

As with perpendicular parking, there are two curved sections, and the radius r for the two paths is calculated as shown in Figure 13. If A and B are defined, the steering control value can be calculated from r.



(**a**) (**b**)

Figure 13. Calculation of r in the parallel parking trajectory: (**a**) calculation of the first circular trajectory and (**b**) calculation of the second circular trajectory.

Angle parking uses the path shown in Figure 14, where the driver first drives forward in a straight line and then backward in a curve. Finally, the vehicle drives backwards in a straight line.

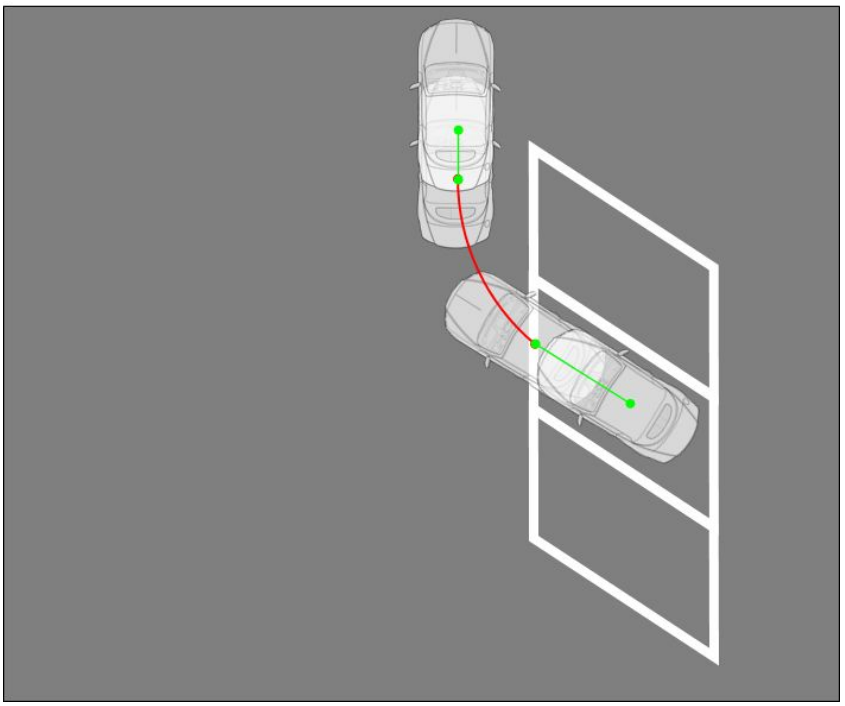


Figure 14. Angle parking trajectory.

In angle parking, one curved section appears, and the corresponding radius r can be obtained using A and B, as shown in Figure 15.

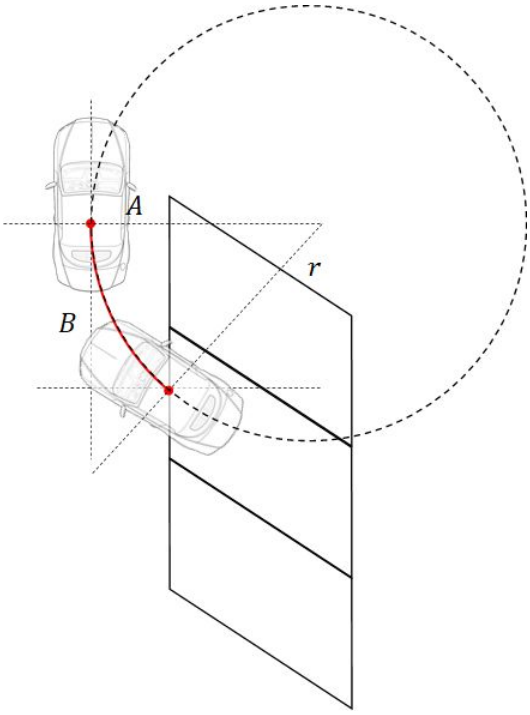


Figure 15. Calcuation of r in the angle parking trajectory.

3.2.2. Motion Tracking

Motion tracking tracks the car along the generated path. During motion tracking, acceleration, braking, steering, and gear control are required to move the vehicle [36].

Automatic parking is performed at low speeds. Therefore, the target value of drive control was set to 10 kph. Based on this value, the drive control was performed using PI control, which is expressed as follows:

$$MV(t) = K_p e(t) + K_i \int_0^t e(t)\, dt. \tag{5}$$

As shown in Equation (5), $u(t)$ is the amount of control, $e(t)$ is the difference in the target speed, and K_p and K_i represent the gain.

In this study, electronic stability control (ESC) was used for braking. When the target point is reached using ESC, the vehicle is stopped by braking. In an automatic parking situation, because the vehicle moves at a low speed, it is possible to stop at the target point with a certain amount of braking. Therefore, when the target point is reached, this method is employed.

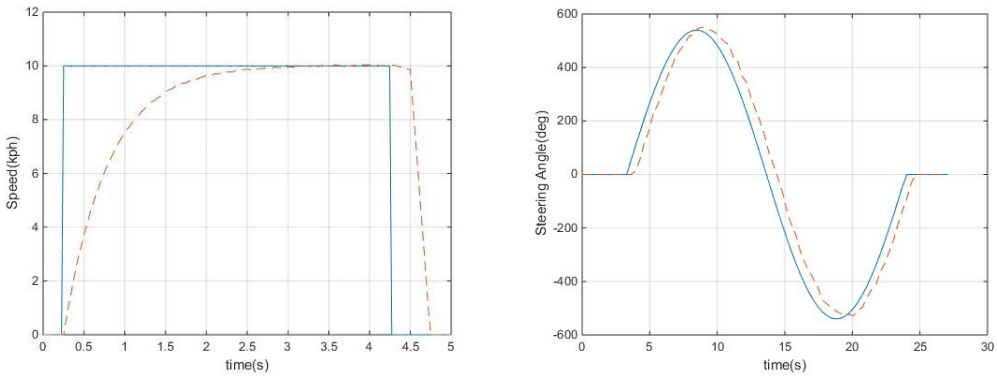
Gear control uses an electronic gearbox. The gear control is configured with forward, reverse, and parking gears.

The steering control is configured to follow the generated path, and for automatic parking, this is obtained as follows [10]:

$$\phi(t) = \phi(t-1) + K_p e_1(t) + K_d(e_1(t) - 2e_1(t-1) + e_2(t)). \tag{6}$$

Here, ϕ is the steering angle, K_p and K_d represent the gain, e_1 represents the closest difference between the curve at the target point and the straight line connecting the center point of the vehicle, and e_2 represents the closest difference between the curve at the target point and the curve of the vehicle.

Figure 16 shows the results of vehicle speed control and steering control. Figure 16a shows the speed control result, and Figure 16b shows the steering control result. As shown in Figure 16, the velocity and steering angle are tracked to the target.



(**a**) (**b**)

Figure 16. Results of speed and steering control: (**a**) result of speed control and (**b**) result of steering control.

4. Experimental Results

4.1. Experimental Environment

For the test, the experiment environment was first configured. High-definition cameras (1280 × 720) were installed on the test vehicle, which was a Cammsys CEVO-C small electric vehicle [37] that was modified for automatic parking. Electronic power steering was installed in the test vehicle for steering control, and ESC was installed in the test vehicle for braking control. In addition, the drive control used the electric vehicle's motor, and the gear was configured so that it could be changed using electronic control. Figure 17 shows the test vehicle.

Figure 17. Test vehicle.

Figure 18 shows the cameras mounted on the front, rear, left and right sides of the test vehicle.

(**a**) (**b**)

Figure 18. *Cont.*

(**c**) (**d**)

Figure 18. AVM camera installation: (**a**) front camera, (**b**) rear camera, (**c**) left camera, and (**d**) right camera.

Test images were acquired in various situations using the cameras mounted on the test vehicle. Figure 19 shows examples of images acquired at each parking slot. As shown, images were obtained in various situations, especially where it was difficult to detect parking lines.

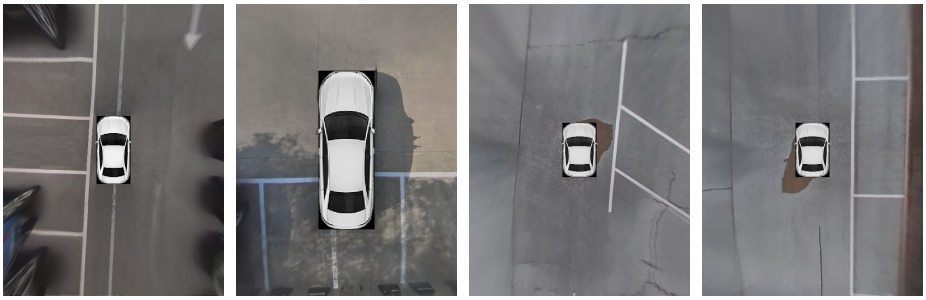
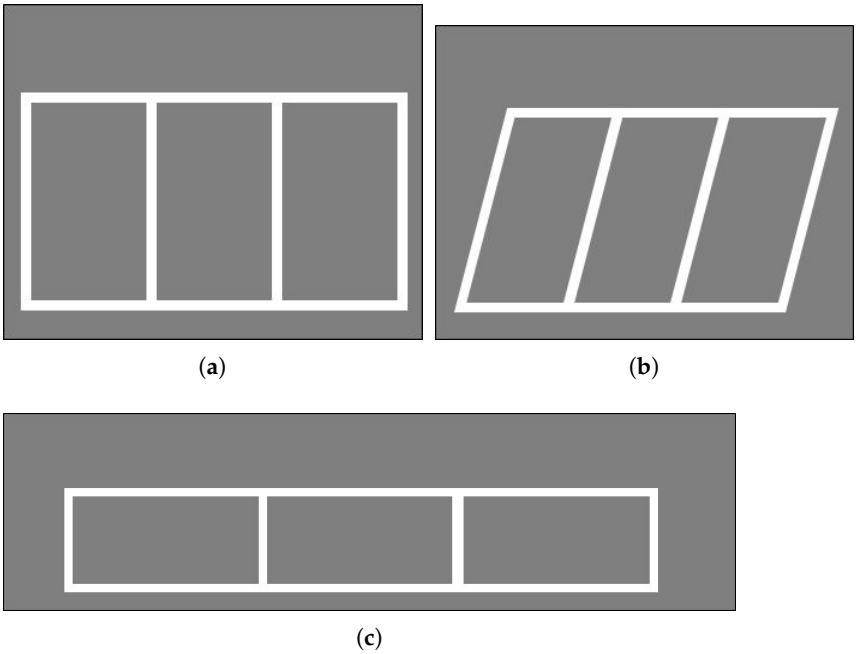


Figure 19. Examples of parking line images.

The automatic parking experiment was conducted on three types of parking slots: perpendicular, parallel, and angle parking slots (Figures 20a–c, respectively).



(**a**) (**b**)

(**c**)

Figure 20. Parking slots for the automatic parking test: (**a**) perpendicular, (**b**) angle, and (**c**) parallel parking slots.

4.2. Experimental Results

Experiments were conducted to evaluate the recognition of parking lines and automatic parking performance of the proposed method.

A dataset was used to evaluate the detection rate of the parking line detection method. The dataset consisted of images of square, angle, open, diamond, and parallel parking slots.

In addition, the detection rate was evaluated using precision and recall, which were calculated as follows [16]:

$$precision = \frac{No.\ of\ correctly\ detected\ parking\ slots}{No.\ of\ detected\ parking\ slots}. \quad (7)$$

$$recall = \frac{No.\ of\ correctly\ detected\ parking\ slots}{No.\ of\ parking\ slots}. \quad (8)$$

Tables 1 and 2 compare the results obtained without and with the CNN, respectively. When the CNN was used, the average precision was 96% and the recall was 94%, which are better results than those obtained when the CNN was not used.

Table 1. Parking slot detection results obtained without the CNN.

Parking Slot Type	No. of Parking Slots	No. of Detected Slots	No. of Miss Detections	No. of Wrong Detections	Precision	Recall
Perpendicular	468	440	47	19	95.68	89.96
Angle	259	243	27	11	95.47	89.57
Parallel	93	88	12	7	92.04	87.10
Sum	820	771	86	34	95.2	89.51

Table 2. Parking slot detection results using the CNN.

Parking Slot Type	No. of Parking Slots	No. of Detected Slots	No. of Miss Detections	No. of Wrong Detections	Precision	Recall
Perpendicular	468	458	23	13	97.16	95.08
Angle	259	253	14	8	96.84	94.59
Parallel	93	90	7	4	95.56	92.47
Sum	820	801	44	25	96.88	94.63

Figure 21 shows the detected parking slotd. The results confirm that various parking line types could be recognized in different environments.

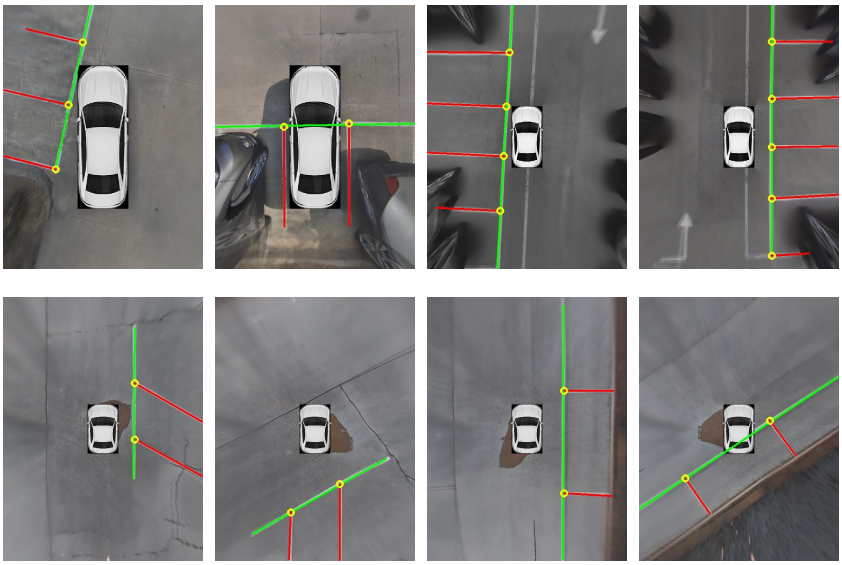


Figure 21. Parking slot detection results.

In the automatic parking experiment, we evaluated the final position after parking. We employed a method to compare the final position and target position after the direct measurement of several points, as illustrated in Figure 22.

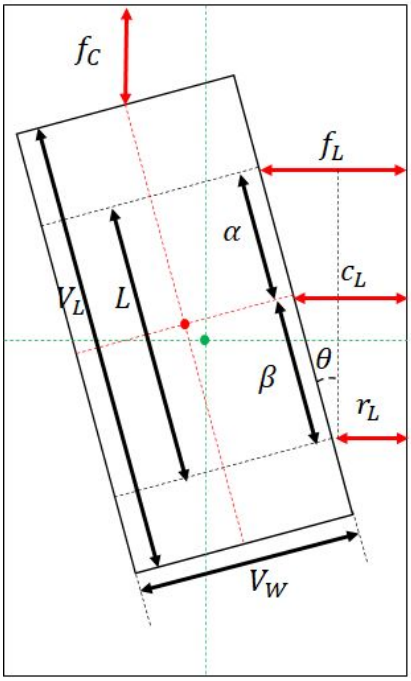


Figure 22. Measurement points for automatic parking result evaluation.

As shown in Figure 22, the distance between each wheel and the parking line as well as the distance between the center of the front of the vehicle and the parking slot were measured. From the positions of these three points, the amount of offset from the center and the heading angle can be obtained. Using the configuration shown in Figure 22, the angle of the vehicle can be obtained as follows:

$$\theta = \sin^{-1}\left(\frac{f_L - r_L}{L}\right). \tag{9}$$

Using angle θ, the difference in the y-axis direction away from the center can be expressed as

$$D_y = \left|\frac{P_y}{2} - \left(\frac{V_L}{2}\cos\theta + f_c\right)\right|. \tag{10}$$

In addition, the x-axis offset can be obtained from

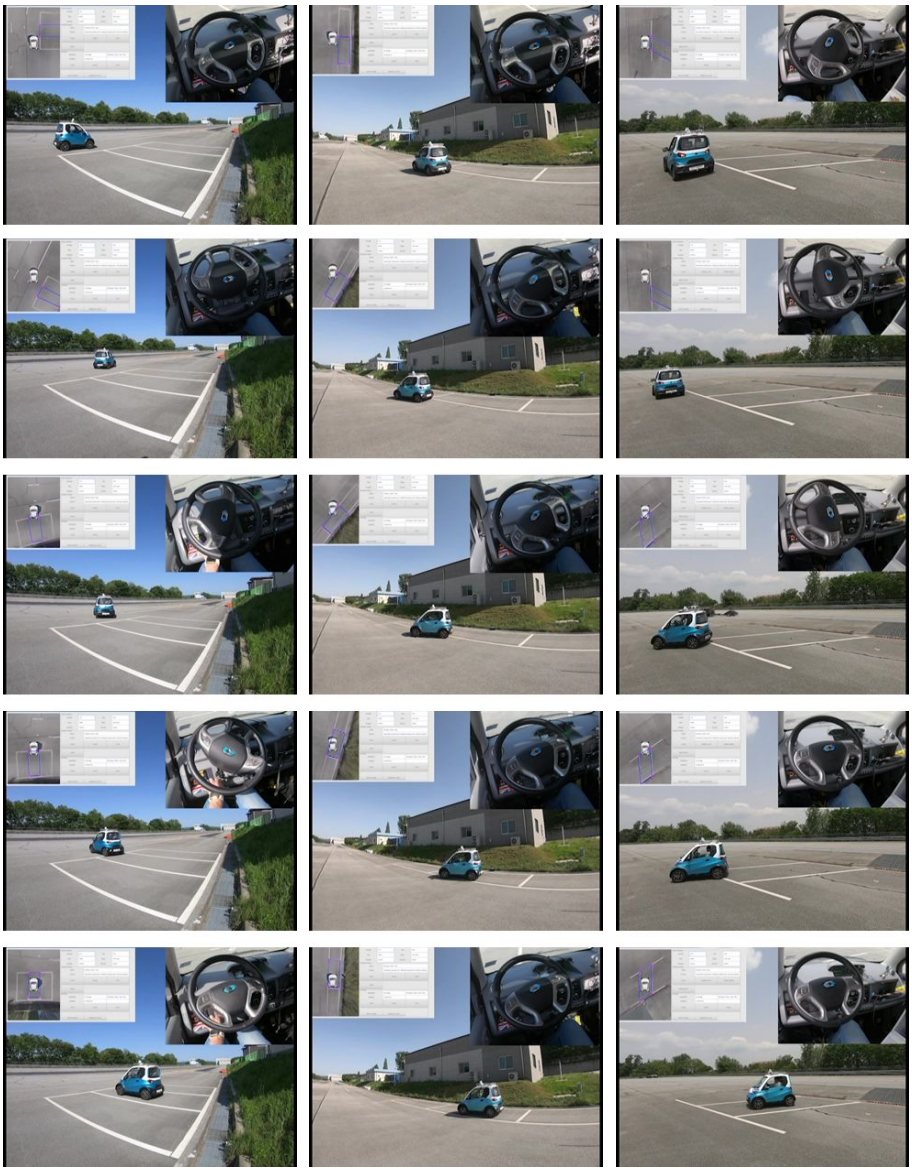
$$D_x = \left|\frac{P_x}{2} - \left(\frac{V_W}{2}\cos\theta + c_L\right)\right|. \tag{11}$$

The experiment was conducted considering the shape of each parking slot. There were three types of parking division lines in the experiment: perpendicular, parallel, and angle parking lines. The results were obtained by detecting the parking slot according to its type and performing automatic parking. Table 3 presents the experimental results (the lateral error, longitudinal error, and heading angle) for each parking slot type.

Table 3. Results of automatic parking using the AVM-based system.

Parking Type	No.	X Offset	Y Offset	Heading Angle
Perpendicular parking	1	0.053	0.204	0.1
	2	0.27	0.211	0.5
	3	0.075	0.197	0.4
	4	0.11	0.023	0.2
	5	0.104	0.024	0.2
	6	0.064	0.01	0.2
	7	0.028	0.173	0.4
	8	0.072	0.162	0.3
	9	0.083	0.074	0.2
	10	0.053	0.204	0.1
Angle parking	1	0.069	0.083	2.4
	2	0.143	0.004	1.7
	3	0.2	0.016	1.1
	4	0.178	0.053	1.4
	5	0.112	0.002	1.3
	6	0.191	0.067	2.1
	7	0.136	0.048	2.2
	8	0.105	0.009	1.2
	9	0.254	0.007	1.0
	10	0.154	0.023	1.6
Parallel parking	1	0.073	0.108	0.8
	2	0.096	0.114	1.2
	3	0.058	0.099	0.7
	4	0.041	0.104	0.6
	5	0.109	0.089	1.1
	6	0.172	0.104	0.7
	7	0.138	0.098	1.2
	8	0.077	0.087	1.4
	9	0.048	0.112	1.0
	10	0.04	0.097	0.9
Average		0.1027	0.0889	0.94

Figure 23 shows the automatic parking results; the upper-left inset image shows the parking slot detection and the upper-right inset image shows the inside of the car. These results confirm that automatic parking using the proposed method was successful.



(**a**) (**b**) (**c**)

Figure 23. Automatic parking results: (**a**) perpendicular parking result, (**b**) parallel parking result, and (**c**) angle parking result.

5. Conclusions and Future Works

5.1. Conclusions

In this paper, we proposed a method for detecting parking slots using an AVM system and using it to perform automatic parking. In the method proposed in this study, feature maps were extracted using a line filter and parking lines were extracted via the Hough transform. In addition, the corners were detected by extracting the parking lines, and the parking slot corner were identified using a deep-learning method (a CNN) that was trained on a database of various environments for more reliable results. In addition, to handle the situation where the parking line is occluded by the movement of the vehicle during automatic parking, the accuracy was improved by tracking the parking line using a Kalman filter from the start of the parking until its completion. This maintained the information on the position of the parking line until the end of the actual automatic parking.

Using the detected parking lines, a parking path was created using circles and lines. Parking was performed along the created parking path. It was evaluated using an actual test

vehicle, and it was confirmed that automatic parking could be completed within a heading angle error of 2.5°, an x-axis offset of 26 cm, and a y-axis offset of 27 cm.

5.2. Future Works

In parking slot detection, because there are various types of corner points, it takes a lot of time to collect the data and train the model. We plan to improve detection performance by collecting data to train the model to detect various types of corner points. In addition, when tracking the parking slot, a Kalman filter was used, but the vehicle motion information was not used. The accuracy of the Kalman filter tracking would be improved by also using the vehicle's motion information.

The proposed automatic parking control method used circular and linear motions. It currently does not consider the case where the opposite side is blocked. Therefore, we plan to create a parking trajectory considering the case in which an obstacle such as a wall or vehicle blocks the way on the opposite side. Parking in a narrow space is possible by increasing the number of forward and backward motions. Such changes to the method of creating the parking trajectory will enable automatic parking even in tight spaces.

In addition, obstacle detection will be added to detect parking slots with obstacles. Deep learning can be used for obstacle detection. This will be implemented so that a vehicle can be automatically parked in an empty parking lot using the obstacle information and parking corner points.

Author Contributions: Y.L. developed the algorithm and performed the experiments. M.P. contributed to the development of the algorithm, validation of the experiments, and research supervision. Y.L. contributed to the writing of the manuscript. M.P. contributed to the review and editing of the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Technology Innovation Program (or Industrial Strategic Technology Development Program—Automobile Industrial Core Technology Development Project) (grant no. K_G012000307004, Development of rear automatic braking system for NCAP) funded by the Ministry of Trade, Industry and Energy (MOTIE, Korea). This work was further supported by the System Industrial Strategic Technology Development Program (grant no. 10079961, Development of a deterministic DCU platform with less than 1 us synchronization for autonomous driving system control) funded by the Ministry of Trade, Industry, and Energy (MOTIE, Korea). Support was also provided by the Basic Science Research Program of the National Research Foundation of Korea (NRF) funded by the Ministry of Education under grant no. 2018R1D1A1B0704814314.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Trivedi, M.M.; Gandhi, T.; McCall, J. Looking-in and looking-out of a vehicle: Computer-vision-based enhanced vehicle safety. *IEEE Trans. Intell. Transp. Syst.* **2007**, *8*, 108–120. [CrossRef]
2. Jung, H.G.; Kim, D.S.; Yoon, P.J.; Kim, J. Parking slot markings recognition for automatic parking assist system. In Proceedings of the 2006 IEEE Intelligent Vehicles Symposium, Tokyo, Japan, 13–15 June 2006; pp. 106–113.
3. Satonaka, H.; Okuda, M.; Hayasaka, S.; Endo, T.; Tanaka, Y.; Yoshida, T. Development of parking space detection using an ultrasonic sensor. In Proceedings of the 13th ITS World Congress, London, UK, 8–12 October 2006.
4. Lee, Y.; Chang, S. Development of a verification method on ultrasonic-based perpendicular parking assist system. In Proceedings of the 18th IEEE International Symposium on Consumer Electronics (ISCE 2014), Jeju, Korea, 22–25 June 2014; pp. 1–3.
5. Liu, Y.C.; Lin, K.Y.; Chen, Y.S. Bird's-eye view vision system for vehicle surrounding monitoring. In *International Workshop on Robot Vision*; Springer: Auckland, New Zealand, 18–20 February 2008; pp. 207–218.
6. Kum, C.H.; Cho, D.C.; Ra, M.S.; Kim, W.Y. Lane detection system with around view monitoring for intelligent vehicle. In Proceedings of the 2013 International SoC Design Conference (ISOCC), IEEE: Busan, Korea, 17–19 November 2013; pp. 215–218.
7. Lee, Y.H.; Kim, W.Y. An automatic calibration method for AVM cameras. *IEEE Access* **2020**, *8*, 192073–192086. [CrossRef]
8. Hamada, K.; Hu, Z.; Fan, M.; Chen, H. Surround view based parking lot detection and tracking. In Proceedings of the 2015 IEEE Intelligent Vehicles Symposium (IV), Seoul, Korea, 28 June–1 July 2015; pp. 1106–1111.
9. Lee, S.; Hyeon, D.; Park, G.; Baek, I.J.; Kim, S.W.; Seo, S.W. Directional-DBSCAN: Parking-slot detection using a clustering method in around-view monitoring system. In Proceedings of the 2016 IEEE Intelligent Vehicles Symposium (IV), Gothenburg, Sweden, 19–22 June 2016; pp. 349–354.

10. Wang, C.; Zhang, H.; Yang, M.; Wang, X.; Ye, L.; Guo, C. Automatic parking based on a bird's eye view vision system. *Adv. Mech. Eng.* **2014**, *6*, 1–10. [CrossRef]
11. Houben, S.; Komar, M.; Hohm, A.; Lüke, S.; Neuhausen, M.; Schlipsing, M. On-vehicle video-based parking lot recognition with fisheye optics. In Proceedings of the 16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013), Hague, The Netherlands, 6–9 October 2013; pp. 7–12.
12. Suhr, J.K.; Jung, H.G. A universal vacant parking slot recognition system using sensors mounted on off-the-shelf vehicles. *Sensors* **2018**, *18*, 1213. [CrossRef] [PubMed]
13. Chen, J.Y.; Hsu, C.M. A visual method tor the detection of available parking slots. In Proceedings of the 2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Banff, AB, Canada, 5–8 October 2017; pp. 2980–2985.
14. Lee, M.; Kim, S.; Lim, W.; Sunwoo, M. Probabilistic occupancy filter for parking slot marker detection in an autonomous parking system using avm. *IEEE Trans. Intell. Transp. Syst.* **2018**, *20*, 2389–2394. [CrossRef]
15. Zhang, L.; Li, X.; Huang, J.; Shen, Y.; Wang, D. Vision-based parking-slot detection: a benchmark and a learning-based approach. *Symmetry* **2018**, *10*, 64. [CrossRef]
16. Kim, S.; Kim, J.; Ra, M.; Kim, W.Y. Vacant parking slot recognition method for practical autonomous valet parking system using around view image. *Symmetry* **2020**, *12*, 1725. [CrossRef]
17. Kim, C.; Cho, S.; Jang, C.; Sunwoo, M.; Jo, K. Evidence filter of semantic segmented image from around view monitor in automated parking system. *IEEE Access* **2019**, *7*, 92791–92804. [CrossRef]
18. Zhang, L.; Huang, J.; Li, X.; Xiong, L. Vision-based parking-slot detection: A DCNN-based approach and a large-scale benchmark dataset. *IEEE Trans. Image Process.* **2018**, *27*, 5350–5364. [CrossRef] [PubMed]
19. Allodi, M.; Castangia, L.; Cionini, A.; Valenti, F. Monocular parking slots and obstacles detection and tracking. In Proceedings of the 2016 IEEE Intelligent Vehicles Symposium (IV), Gothenburg, Sweden, 19–22 June 2016; pp. 179–185.
20. Lee, S.; Seo, S.W.; Available parking slot recognition based on slot context analysis. *IET Intell. Transp. Syst.* **2016**, *10*, 594–604. [CrossRef]
21. Zips, P.; Böck, M.; Kugi, A. A fast motion planning algorithm for car parking based on static optimization. In Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, 3–7 November 2013; pp. 2392–2397.
22. Zips, P.; Böck, M.; Kugi, A. Optimisation based path planning for car parking in narrow environments. *Robot. Auton. Syst.* **2016**, *79*, 1–11. [CrossRef]
23. Lin, L.; Zhu, J.J. Path planning for autonomous car parking. In Proceedings of the Dynamic Systems and Control Conference, American Society of Mechanical Engineers, Atlanta, GA, USA, 30 September–3 October 2018; pp. 1–10.
24. Sedighi, S.; Nguyen, D.V.; Kuhnert, K.D. A new method of clothoid-based path planning algorithm for narrow perpendicular parking spaces. In Proceedings of the 5th International Conference on Mechatronics and Robotics Engineering, Rome, Italy, 16–18 February 2019; pp. 50–55.
25. Kim, D.J.; Chung, C.C. Automated Perpendicular Parking System with Approximated Clothoid-Based Local Path Planning. *IEEE Control. Syst. Lett.* **2020**, *5*, 1940–1945. [CrossRef]
26. Vorobieva, H.; Glaser, S.; Minoiu-Enache, N.; Mammar, S. Automatic parallel parking in tiny spots: Path planning and control. *IEEE Trans. Intell. Transp. Syst.* **2015**, *16*, 396–410. [CrossRef]
27. Li, T-H; Chang, S.-J. Autonomous fuzzy parking control of a car-like mobile robot. *IEEE Trans. Syst. Man Cybern.-Part A Syst. Hum.* **2003**, *33*, 451–465. [CrossRef]
28. Kim, D.; Chung, W. Motion planning for car-parking using the slice projection technique. In Proceedings of the 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, Nice, France, 22–26 September 2008; pp. 1050–1055.
29. Kwon, H.; Chung, W. Performance analysis of path planners for car-like vehicles toward automatic parking control. *Intell. Serv. Robot.* **2014**, *7*, 15–23. [CrossRef]
30. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 1097–1105. [CrossRef]
31. Zeiler, M.D.; Fergus, R. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision*; Springer: Zurich, Switzerland, 6–12 September 2014; pp. 818–833.
32. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
33. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition(CVPR), Boston, MA, USA, 7–12 June 2015; pp. 1–9.
34. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]
35. Peizhi, Z.; Zhuoping, Y.; Lu, X.; Dequan, Z. Research on Parking Slot Tracking Algorithm Based on Fusion of Vision and Vehicle Chassis Information. *Int. J. Automot. Technol.* **2020**, *21*, 603–614.
36. Park, M.; Lee, S.; Kim, M.; Lee, J.; Yi, K. Integrated differential braking and electric power steering control for advanced lane-change assist systems. *Proc. Inst. Mech. Eng. Part D J. Automob. Eng.* **2015**, *229*, 924–943. [CrossRef]
37. Cammsys Corp. CEVO-C. Available online: https://www.cevo.co.kr/ (accessed on 25 October 2021).