Real-Time Downward View Generation of a Vehicle Using Around View Monitor System

Jinkyu Lee⋆, Myungchul Kim⋆, Sangjun Lee, *Student Member, IEEE,* and Sungsoo Hwang, *Member, IEEE*

***Abstract*—In the study, a method to generate the downward image of current vehicle location using a commercial around view monitor (AVM) is proposed. The proposed system consists of three stages, namely feature tracking, obstacle filtering, and downward view generation. In the feature tracking stage, the Shi-Tomasi corner detection is used and feature tracking is performed with a KLT (Kanade-Lucas-Tomasi) tracker to determine the transformation between AVM images. In the obstacle filtering stage, features on obstacles are filtered by using the difference in a tracking distance between features detected on the ground and features detected on the obstacle. This is performed by using a histogram based on the tracking distance of features. Finally, in the downward view generation stage, transformation between the current AVM image and previous AVM image is obtained by using the tracked feature pair refined via the aforementioned steps. Specifically, the random sample consensus (RANSAC) method is used to obtain transformations from which the influence of outliers is removed. The downward image of the current vehicle is generated by the obtained transformation and is synthesized to the existing AVM image. The results indicate that the proposed system synthesizes the existing AVM image and the generated downward image of a vehicle in a seamless way by determining the exact pair of matching points between current and previous AVM images. We believe the proposed system can be utilized in efficient wireless charging system and safe driving system.**

***Index Terms*—Around view monitor (AVM), advanced driver assistance system (ADAS), feature tracking, transformation estimation, obstacle filtering**

I. Introduction

ADvanced Driver Assistance Systems (ADAS) are systems that provide information and warnings, with respect to risk of accidents to assist drivers in safe driving. The system comprises several sensors that are installed on the vehicle. The system is one of the most important underlying technologies for autonomous vehicles, and several automakers currently offer a variety of ADAS options. Among them, a visual-based ADAS using a vision sensor can be installed at a relatively low cost. Hence, it is widely adopted.

An around view monitor (AVM) [1] is one of the visual-based ADAS. It consists of four cameras mounted on the front, rear, left, and right sides of a vehicle. The system synthesizes images from four cameras to provide a view from the top of the vehicle. Recently, many manufacturers are providing vision-based ADASs similar to AVM. For example, 360° view system, surround camera system, surround view monitoring system, multi-angle vision system, bird's-eye view vision system. These systems differ only in the number of cameras or the types of viewpoints available, but the main features and related technologies are the identical. As the technology which show different viewpoints using multi cameras such as AVM evolves and popularizes, several applications using this technology have been proposed. S. Hecker, et.al. [2] summarized representative examples. For instance, Advanced driver assistance systems often use the bird's eye view and the 3-D surround view, which provides the panoramic image of vehicle surrounds. This will help the driver know the driving environment and remove visual blind spots [3]–[5]. Trajectories and maneuvers of surrounding vehicles are estimated with panoramic camera arrays [6]. Algorithms for detecting and tracking pedestrians and vehicles using all around view are studied in [7]. Lane detection using AVM is investigated in [8], [9]. Transform model is proposed using one rear camera for parking assistance in [10]. The system for recognizing and detecting parking spaces using bird's eye view was studied in [11], [12]. The surround view is used to guide the driver to make parking easier in [13]. Thus, various studies have been conducted on the necessity of supporting the driver's driving and developing the vehicle for autonomous driving.

However, to the best of our knowledge, a system for displaying downward images of the vehicle has not been studied. Previous studies only provide the surrounding view of the vehicle, and the car's underside is empty or covered with a virtual image. The downward view of the vehicle is useful when the car needs to move to the correct position. With the recent mass distribution of electric vehicles, many

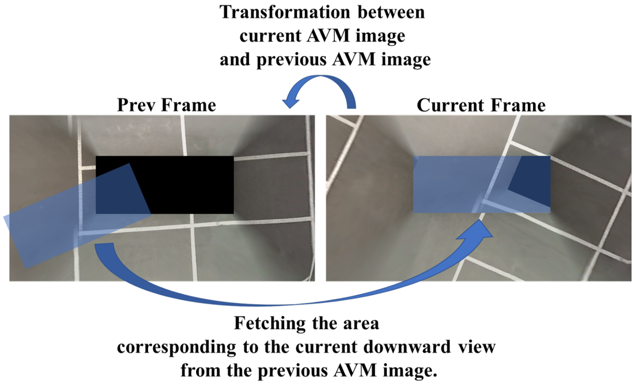


Figure 1. Method to generate downward image of a vehicle.

This work was supported by the Vadas Company.

J. Lee⋆ and M. Kim⋆ contributed equally to this work.

J. Lee⋆, M. Kim⋆, S. Lee, and S. Hwang are with the School of Computer Science and Electrical Engineering, Handong Global University, Pohang, 37554, Korea (e-mail:21200590@handong.edu; gritycda@gmail.com; eowjd4@naver.com; sshwang@handong.edu).

automobile companies are developing and commercializing wireless charging systems [14] for the safety and convenience of battery charging. Even roadway-powered electric vehicle systems [15], [16], which are charged while driving without having to go to the charging station, are also being developed. In these wireless charging systems, the charging efficiency is greatly affected by alignment between the charging pad on the ground and the charging panel of the vehicle. K. Y. Kim, et al. [17] has tested the magnetic resonance wireless charging efficiency according to misalignment to demonstrate that correct alignment is important to increase charging efficiency. Therefore, the downward view of the vehicle is needed to help drivers easily check alignment with wireless charging pad. Furthermore, the downward view of the vehicle can be helpful for safe driving in cases when cracks and obstacles on the road can cause fatal damage to tires.

In the AVM image sequence, the downward image of the current vehicle is appeared in the previous scene. The AVM image of the previous time is required to generate the downward image of the vehicle using AVM. The downward image of the current vehicle is generated by fetching the area corresponding to the current downward view from the previous AVM image as shown in Fig. 1. The process can be performed by estimating transformation between the current AVM image and previous AVM image. Generally, the transformation is estimated by determining feature correspondences between images [18]. However, it is a challenging task to obtain the transformation from the AVM image.

There are three characteristics that make it difficult to obtain the transformation from the AVM image. First, the AVM image is mainly composed of roads, which are characterized by repeatedly distributed similar textures. The determination of feature correspondences in the environment exhibits constraints. Second, there is a partial loss (for e.g., blurring or seam lines) on the AVM image owing to the process of creating the AVM. It is difficult to detect features on a low-quality image. Finally, the AVM image contains objects located on the road, and features detected on those objects make the transformation inaccurate.

In the present study, we propose a novel method to obtain the accurate transformation even in the challenging conditions of the AVM image. First, we use a KLT (Kanade-Lucas-Tomasi) feature tracker [19] that is robust even for images with similar textures. The other methods [20]–[22] that find the feature correspondences by describing a feature with the descriptor do not perform well on images that consist of similar textures. Second, we set the region of interest (ROI) for feature detection avoiding the defective parts of AVM image as possible. Finally, while the features detected from the road area exhibit the same motion, the features detected from the objects exhibit different motions when compared to the road area. Based on the aforementioned difference, the features from the object are removed.

The study includes the following three contributions. First, we provide the downward view of the vehicle to drivers so that they can accurately locate the vehicle. Second, we propose a robust method to determine the transformation even in the challenging environment of an AVM image that exhibits poor quality and contains other objects which are not the ground. Third, in order to find features suitable for feature-tracking in road environments, we compared the features widely used through experiments.

The rest of the study is organized as follows. Section II discusses related work. Sections III details an overview of the proposed method. Sections IV–VI describe the stages for feature tracking, obstacle filtering, and downward view generation. Section VII presents experimental results by using a commercial AVM database. Finally, section VIII concludes the study.

II. Related Work

A. Around View Monitor (AVM)

The AVM system is composed of fisheye cameras to ensure the wide field of view. In order to create an AVM image with the fisheye camera, the distortion should be corrected. There are methods [23]–[25] to model the distortion of the fisheye camera. These distortion models commonly indicate that increases in the distance from the principal point increase the degree of distortion. Therefore, the corrected image tends to exhibit higher loss toward the edge due to distortion correction.

In order to convert the corrected image to the topview, a homography is calculated by four or more pairs of points between undistorted image and targeted ground plane [26]. The image is warped with the estimated homography. While this warping into the topview exhibits the effect of removing the perspective distortion, it stretches the image of the area farther from the camera. Therefore, the blur increases when the distance from the camera increases.

The images for which distortion is corrected and converted into topview are attached into a single image by considering each position. There are several blending approaches [27], [28] used in the image stitching method to attach multiple images without seam lines. However, it is not appropriate to apply the aforementioned methods on commercial AVM operating on an embedded computing environment. Therefore, several automakers either skip the blending process or use simpler methods.

B. Retrieving Vehicle's Translation and Rotation from AVM

In order to generate downward image of a vehicle, we need to fetch the area corresponding to the current downward view from the previous AVM image. This requires transformation estimation between two AVM frames, and it is equivalent to estimate translation and rotation of a vehicle. Recently, several researchers proposed the methods to retrieve vehicle's translation and rotation by using the information around the vehicle provided by AVM. Jae Kyu Suhr and Ho Gi Jung [12] proposed a method to recognize parking slots in various shapes by detecting parking lines in the AVM image, and track the parking slots based on DCM (directional chamfer matching) [29] method. This is corresponding to obtain the translation and rotation of the vehicle. However, their method is not purely vision-based, i.e., they utilize a motion sensor.

Dongwook Kim et. al. [9] determined the lanes in the AVM image and matched the digital map that contained geo-localized lane markings to obtain a rigid transformation by using ICP (iterative closest point) [30]. Similar to the aforementioned method, they additionally use a wheel speed sensor and a yaw rate sensor of a vehicle.

Senbo Wang et. al. [31] proposed a purely vision-based method which is similar to the proposed method. They acquired feature correspondences from AVM sequence, and calculated moving distance and rotation angle of a vehicle. They filtered out the estimation results by using repeated median estimator [32] to deal with the errors on the feature correspondences. Nevertheless, repeated median estimator has a breakdown point of 50% [33], which means it only can handle for less than 50% of outliers. Therefore, it is impossible to derive the result in AVM frames covered with the obstacles more than half the area. In our study, we proposed a method to rule out the region with the obstacles by the difference in motion between the features on the ground and the features on the obstacles.

III. System Overview

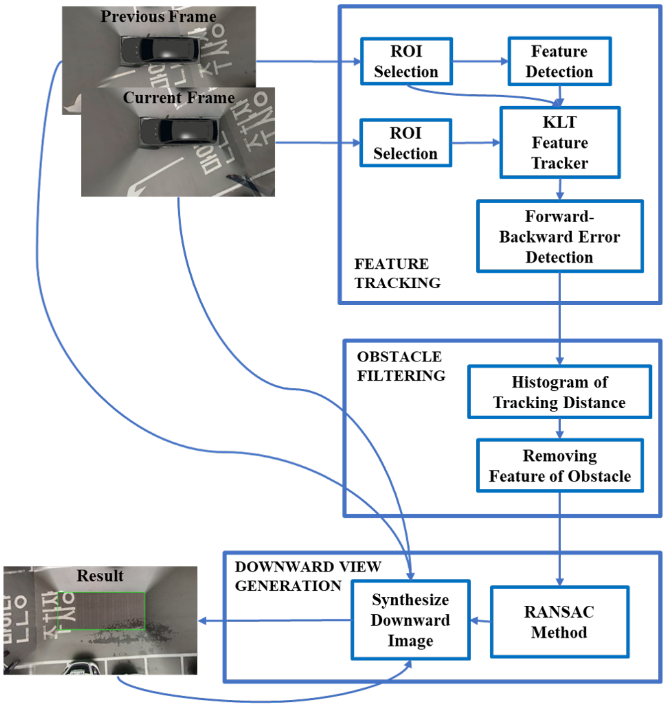


Figure 2. Structure of the downward view generation of vehicles using AVM.

The system consists of three stages, namely feature tracking, obstacle filtering, and downward image generation. First, the feature is detected and tracked among the continuously inputted AVM images in the feature tracking stage. After detecting the features by using the Shi-Tomasi corner detection method, the feature points are tracked by using the Kanade-Lucas-Tomasi feature tracker method [19]. In the obstacle filtering stage, features of the tracked features that do not correspond to the road area are considered as outliers and removed. The histogram of the distance of the tracked features is used to distinguish the road area from the obstacle area. Finally, in the downward image generation stage, transformations are obtained using the refined feature point pairs. The random sample consensus (RANSAC) algorithm [34] removes the remaining outliers. Transformation is used to generate the downward image of the vehicle by fetching the area corresponds to the location of the vehicle in the current image from the previous image.

IV. Feature Tracking

A. Region of Interest (ROI) Selection

The AVM used in the study is composed of four fisheye cameras to ensure the field of view. The building process of the AVM is followed as described in section II-A. The distortion is corrected by assuming the omnidirectional camera model [25], and the corrected image is transformed into a topview. The method based on weighted average [35] is used to blend four images into a single image while blurred seam lines remain. In the fisheye image, the pixels related to wide FoV (field of view) tend to have significantly low resolution. This leads to degrade the quality of AVM image near the seam lines. [36] The RoI should be selected away from seam lines. The red solid boxes and the blue dashed boxes in Fig.3 are properly selected RoI.

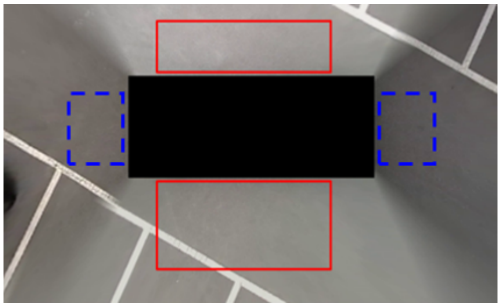


Figure 3. Selected ROI on the AVM image.

Vehicles are generally prone to move back and forth. Therefore, the front and rear sides of the vehicle are inappropriate for ROI because the tracked features last for a short period. In order to track the features for a longer period based on the motion of the vehicle, the left and right sides of the vehicle are suitable for ROI. Hence, in Fig.3 only the red boxes are selected for the RoI of our system.

B. KLT Feature Tracker

The AVM image is mainly composed of road elements. The feature matching method is inappropriate on a road environment that is mainly composed of similar textures. Therefore, the feature tracking method is used to determine feature correspondence on the AVM image. With respect to the feature detection, Harris corner [37], SIFT [20], ORB [22], and Shi-Tomasi corner [38] were considered. In the experiment (described in Section VII-B), the Shi-Tomasi corner is selected since more features are detected and the tracking lasts longer. Specifically, 100 features for each ROI are simultaneously detected by Shi-Tomasi corner detection. Subsequently, we track those features by using the forward-additive KLT method [19]. The feature detection is repeated when the tracked features are reduced to 10 or less.

C. *Forward-Backward Error Detection*

During the feature tracking process, tracking failure can occur when the feature is out of the camera view or when there is an intensive change on the feature point [39]. When the feature tracking is performed for k frames, the distance between the original point and the point tracked in the backward direction by k frames is termed as the forward-backward error [39]. The features that exhibit a forward-backward error corresponding to 2 pixels or more are considered as tracking failure, and the aforementioned features are removed. The number of k is critical for the real-time performance as the feature tracking module is a relatively high computational process. The number of k is determine by 3 through the experiment (described in Section VII-C).

V. Obstacle Filtering

When the features are mainly detected and tracked on obstacles, the transformation between the images is not accurate. This is because the AVM images are subjected to the process of (1) removing the distortion of the fisheye lens and (2) composing a bird's-eye-view image by transforming perspective images to planar images as viewed from above. Both the aforementioned processes result in distortion in an object on the ground. Therefore, in the AVM image, tracking on an object on the ground increases the error rate of the transformation.

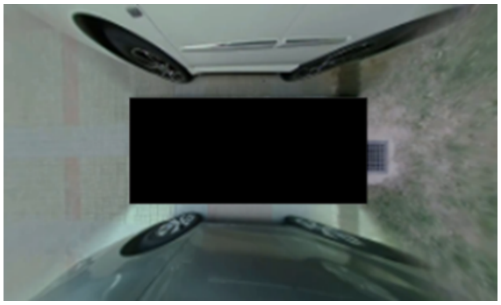


Figure 4. AVM in a situation where there are vehicles on both sides of the vehicle

Fig. 4 shows the AVM in a situation where there are vehicles on both sides of the vehicle. Significant distortions are observed in the right and left vehicle portions. Therefore, even if the feature points are well tracked in the feature tracking process, it is necessary to remove the tracked feature points in area of obstacles on the ground.

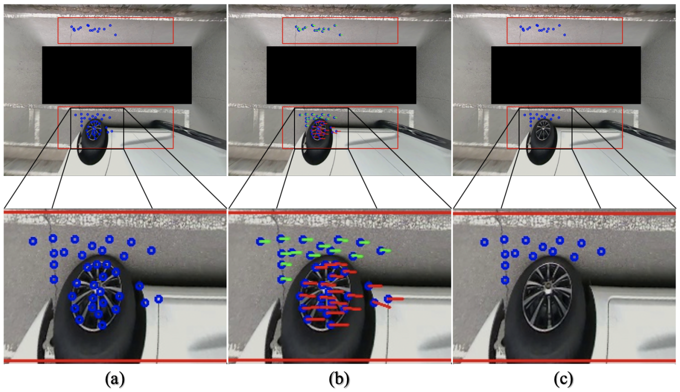


Figure 5. The process of obstacle filtering

Fig. 5 represents the process of the obstacle filtering stage. In Fig. 5 (a), the currently tracked feature points are marked with a blue circle. The *tracking distance* means the distance between the feature point of the current frame and the tracked feature point of the next frame. This *tracking distance* of all feature points in the same frame should be theoretically the identical. However, the *tracking distances* of the feature points tracked on the obstacle have a different tendency from the *tracking distance* of the feature points on the ground. Therefore, this *tracking distance* is used to distinguish the feature points on the obstacle from the feature points on the ground. Fig. 5 (b) shows the *tracking distance* between the currently tracked features(blue circles) and the tracked feature points of the next frame using a straight line. In histogram of *tracking distance* step, the *tracking distance* histogram is created based on the *tracking distance* of each feature point, and the distribution of the histogram classifies the existence of obstacles and the movement of the vehicle. Fig. 5 (C) is the result of eliminating tracked feature points on obstacles according to the classification of the histogram distribution in the removing feature of obstacle step.

A. *Histogram of* Tracking Distances

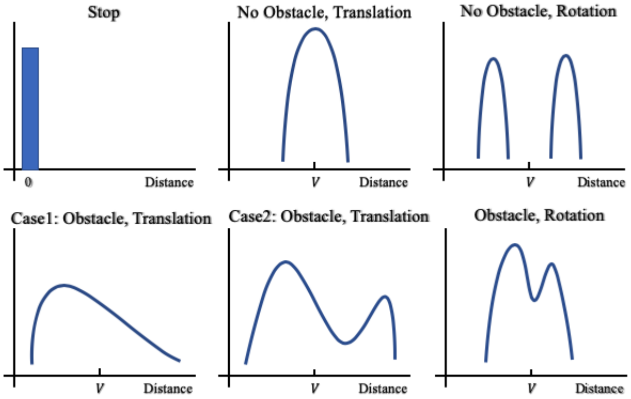


Figure 6. The *tracking distance* histograms classified by the presence of obstacles and the status of translation and rotation

To distinguish the feature points on the obstacle from the feature points on the ground, the *tracking distance* histogram is established based on the *tracking distance*. However, various histogram distributions are generated depending on the presence or absence of obstacles as well as the rotation and translation of the vehicle. Fig.6 shows the *tracking distance* histograms classified into six categories according to the presence of obstacles and the state of rotation and translation. In the stop state, the *tracking distances* of all feature points are zero. Since the current AVM image and the previous AVM image are the same when the vehicle is in a stop state, the *tracking distances* are zero regardless of whether there is an obstacle or not. When there is no obstacle and the translation situation, the *tracking distances* are distributed in the form of a bell-shape based on the speed V of the vehicle. For a rotating situation, the *tracking distances* near the axis of rotation are shorter than the *tracking distance* away from the axis. Therefore, the *tracking distances* show a bimodal distribution separated by the distance from the axis. When there is an obstacle and in a translation situation, it is distinguished by whether the obstacle exists only in one ROI (case 1) or both ROI (case 2). The distribution of the *tracking distances* gradually decreases or increases around the mode due to the slope of the obstacle. Finally, in the case of obstacle and rotation, the distributions vary depending on the obstacle location and distance from the axis of rotation. However, in general, the distributions of the *tracking distances* appear as the bimodal distribution.

B. Removing Feature of Obstacle

To compute an exact transformation, the *tracking distance* histograms are used to remove feature points on obstacles that deviate from the predominant tendency(the mode). The *tracking distance* corresponding to the mode is considered to be the predominant tendency because it is the *tracking distance* corresponding to the largest number of feature points in the image. If there is one mode such as distribution with bell-shaped distribution, it removes the feature points of *tracking distance* which is much different from the *tracking distance* of mode. In the implementation, feature points of the *tracking distance* that differ by more than 1 from mode are removed. If there are two modes, such as symmetric bimodal distribution, also remove feature points with a *tracking distance* that is much different from the mode. However, if one of the two modes is very dominant, such as skewed bimodal distributions, the feature points are removed in the same manner as in the case of one mode. Thus obstacle filtering keeps the feature points on the ground as much as possible while removing feature points on obstacles such as vehicles. This stage is necessary to obtain an accurate transformation with RANSAC in the downward view generation stage.

VI. Downward View Generation

It is assumed that the vehicle in the AVM image that is converted to the fixed top view point exhibits a rigid transformation relation between the AVM image frames because there is only rotation and translation. The transformation $[A \mid b]$:

$$[A \mid b] = \left[\begin{array}{cc|c} s\cos\theta & -s\sin\theta & t_x \\ s\sin\theta & s\cos\theta & t_y \end{array}\right] \tag{1}$$

where s denotes scale, θ denotes rotation, t_x denotes movement of x axis, and t_y denotes movement of y axis. It is better to consider the scale even if it is a rigid transformation, as there is no exact equivalent scale in real world. If the scale is noticeably different to 1, the prediction is deemed wrong.

When a rigid transformation is computed, a transformation from the current frame to the previous frame in the backward direction is computed, not the forward direction from the previous frame to the current frame, described in Fig.1

$$\begin{bmatrix} x_{prev} \\ y_{prev} \end{bmatrix} = A \begin{bmatrix} x_{cur} \\ y_{cur} \end{bmatrix} + b \tag{2}$$

For each feature correspondence, x_{prev}, y_{prev} are the x, y coordinates of feature in previous frame, and x_{cur}, y_{cur} are the x, y coordinates of feature in current frame. In the backward direction approach, the transformation $[A \mid b]$ is applied on the feature point of current frame. The method of backward direction is advantageous in synthesizing the downward image of the vehicle. The application of a forward rigid transformation from the previous frame to the current frame results in typecasting in integer units, thereby resulting in pixels with no value. Therefore, additional interpolation method is required to handle those holes. However, the application of the backward direction rigid transformation that fetches the corresponding region from the previous frame in the current frame is such that it can be implemented without an additional interpolation process.

While calculating the rigid transformation, we use the purified tracking feature pair in the obstacle filtering stage. The RANSAC method is used to remove outliers that may still remain. We randomly select a subset of feature pairs from all the tracked feature pairs and obtain a rigid transformation which minimize the sum of geometric error [26]:

$$[A^* \mid b^*] = \arg\min_{[A|b]} \sum_i \|X_{prev}[i]^\intercal - AX_{cur}[i]^\intercal - b\|^2 \tag{3}$$

$X_{prev}[i]$ and $X_{cur}[i]$ are i-th feature correspondence of selected subset. The calculated rigid transformation matrix is applied to all features in the current frame, and we calculate the distance to each of the corresponding feature in the previous frame. Subsequently, we count the number of features with a distance less than the threshold. The process is repeated several times, and the rigid transformation with the highest counted number is considered as a representative transformation of the aforementioned feature pairs. The experiment to determine the iteration number is described in Section VII-E.

The obtained representative rigid transformation is applied on the under-vehicle region of the current image to obtain the corresponding region in the previous frame.

$$I_{cur}(X_{cur}) = I_{prev}(A^* {X_{cur}}^\intercal + b^*), \quad \forall X_{cur} \in D \tag{4}$$

$I_{cur}(X)$ denotes the RGB value of the point X in AVM image of current frame, $I_{prev}(X)$ denotes the RGB value of the point X in the AVM image of previous frame, and D denotes the downward region of current frame. The RGB value of each pixel in the current downward region is filled with the RGB value of the corresponding point in the previous downward region. The downward image of the vehicle is synthesized to the current AVM by the following process. The AVM image including the completed downward image is used as the previous frame while creating the downward image of the next frame.

VII. Experiments

A. Description of Database

The database used in the experiment was obtained through the AVM System provided by VADAS Surround View 1st generation [40]. Fig.7 shows the four camera positions that are mounted on the vehicle. The AVM system uses images obtained from four cameras on the front, rear, left and right sides of the vehicle. The resolution of the four cameras acquired from the fisheye lens is 1280 * 720, and the resolution of the AVM generated using the camera is 1280 * 720. Our database is 43 datasets composed of 534694 frames in total.

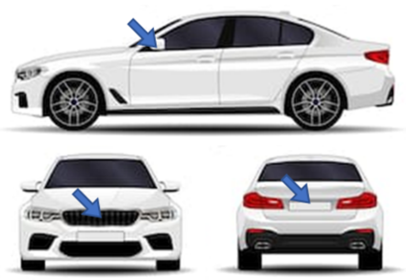


Figure 7. Location of cameras of the AVM system.

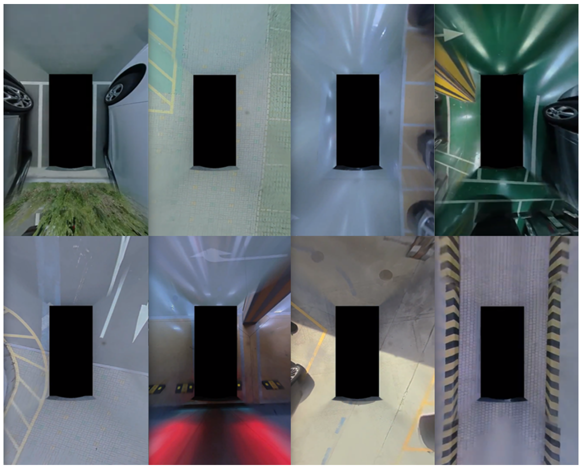


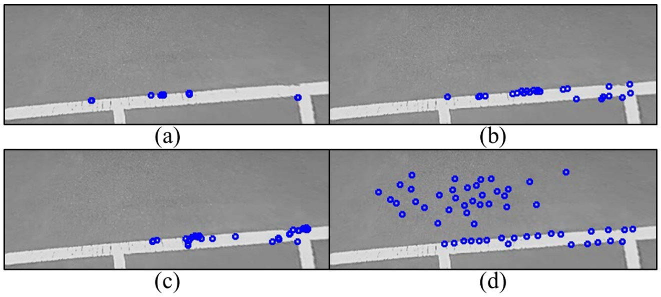
Figure 8. VADAS Surround View 1st generation

The AVM images are obtained in various environments such as cement, concrete, cracked floor as shown Fig.8, and includes general driving environment such as shadow or downhill. Indoor environments such as parking lots are coated with epoxy or urethane for waterproofing. Therefore, the reflected light of the room illumination is included in the AVM image. The transformations between images are obtained based on the feature point tracking, and thus the indoor environment in which the indoor light is reflected and the nighttime environment are not the proposed system environments. Therefore, it was experimented with various daytime outdoor environments.

B. Feature Decision

Table I
Feature Decision

Category	Type of feature	Dataset		
		A	B	C
Number of detected features	Harris Corner	31.45	14.85	90.7
	SIFT	42.82	7.23	97.95
	ORB	27.47	5.45	73.21
	Shi-Tomasi	96.11	87.45	100
Number of frames feature tracked	Harris Corner	2.18	1.25	12.45
	SIFT	1.65	1.37	6.72
	ORB	11.14	1.50	31.26
	Shi-Tomasi	17.49	8.15	21.89



(a) (b) (c) (d)

Figure 9. Captured ROI frames of different features being tracked, on the same moment of the dataset C. (a) is Harris corner, (b) is SIFT, (c) is ORB, and (d) is Shi-Tomasi corner.

Table I shows the result of an experiment to determine the feature to be used for the feature tracking described in Section IV-B. The experiment was done on 3 datasets composed of 4644 frames. The dataset A is an asphalt road environment, the dataset B is a brick road environment, and the dataset C is a parking lot environment. Each feature was detected up to 100, and tracking was performed with KLT tracker to count the number of frames in which feature points were maintained. The new set of features will be detected when the number of tracked features is less then 10. The experiment was arbitrarily performed on 3 datasets. The Shi-Tomasi corner detection method was able to detect the maximum number of features in most frames. Additionally, in most datasets, feature tracking with the Shi-Tomasi corner detection was the most stable method and maintained for a long time. In Fig.9 the features on the road area are detected well by Shi-Tomasi corner detection, but others are not. In conclusion, the Shi-Tomasi corner detection is the most appropriate method for feature tracking in the road environment.

C. Decision for the Number of Frame in Forward-backward Error Detection

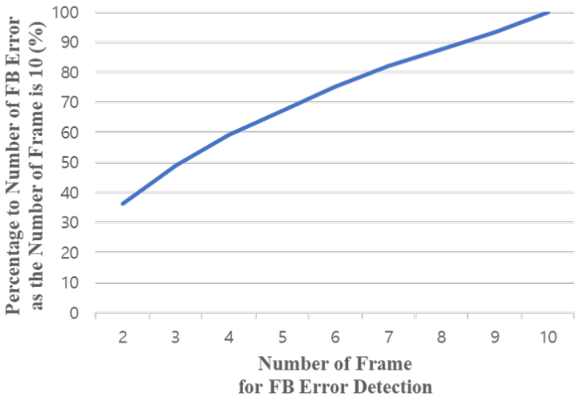


Figure 10. Experiment result to determine the number of frames in the forward-backward error detection.

Fig. 10 shows the results of an experiment to determine the number in the forward-backward error detection step described in Section IV-C. As the length of frame for the forward-backward error detection increases, more forward-backward errors are detected. We empirically found that almost all the forward-backward errors, which are likely to occur, are detected with the 10 frames. Hence, we set it to 100% when the number of frame is 10, in the Fig. 10. The feature tracking step is a relatively large operation because of the recursive feature tracking process on the backward direction. When the number of frames is three, more than 50% of forward-backward errors are detected. And this is acceptable, as we still have two more steps (obstacle filtering, and RANSAC) that handle the outliers. Thus, the number of frame is considered by 3 to guarantee the real-time process.

D. Obstacle Filtering Performance

This section evaluates the performance of obstacle filtering that removes feature points that are initialized in the vehicle. The evaluation uses only the image of the vehicle with the ROI in the existing database. The performance is measured by using recall and precision.

$$recall = \frac{No.ofcorrectlyremovedfeaturepoints}{No.offeaturepointsonvehicle} \tag{5}$$

$$precision = \frac{No.ofcorrectlyremovedfeaturepoints}{No.ofremovedfeaturepoints} \tag{6}$$

The experiment was executed in an environment with an obstacle vehicle. The rotation and the translation situation are separated and evaluated separately. When the number of features is less than a certain number in the tracking process, feature detection is performed again, and feature points caught on the vehicle are removed within a few frames after detection. Therefore, recall and precision were obtained by counting from the moment of feature detection in 5616 images with the vehicle in ROI.

Table II
Performance of the Proposed obstacle filtering Stage

Type of distribution	**# of correctly removed feature points**	**# of feature points on vehicle**	**# of removed feature points**	**Recall**	**Precision**
Obstacle, Translation	448	499	482	89.8%	92.9%
Obstacle, Rotation	360	515	427	69.9%	84.3%
Total	**808**	**1014**	**909**	**79.7%**	**88.9%**

Table II shows the results of the experiment. In the translation situation with obstacles, recall is 89.8% and precision is 92.9%. In the rotation situation, recall is 69.9% and precision is 84.3%, by removing the feature points on the vehicle to the maximum possible extent and maintaining the feature points on the road by the proposed method. In the stage of obtaining the transformations, we obtain more precise transformations by sampling the feature points on the road. Thus, it is synthesized stably without being affected by the outliers in the image synthesis step.

E. Decision for the Number of RANSAC Iteration

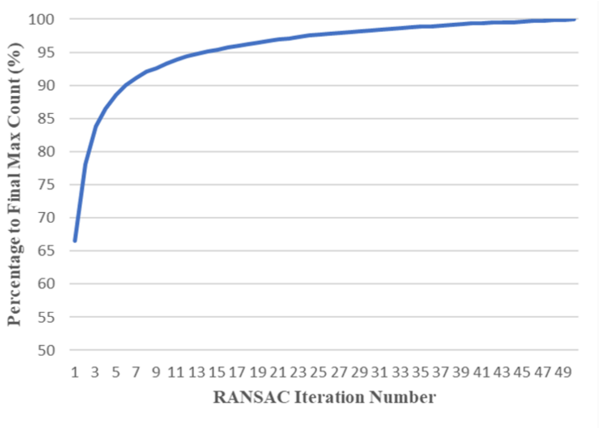
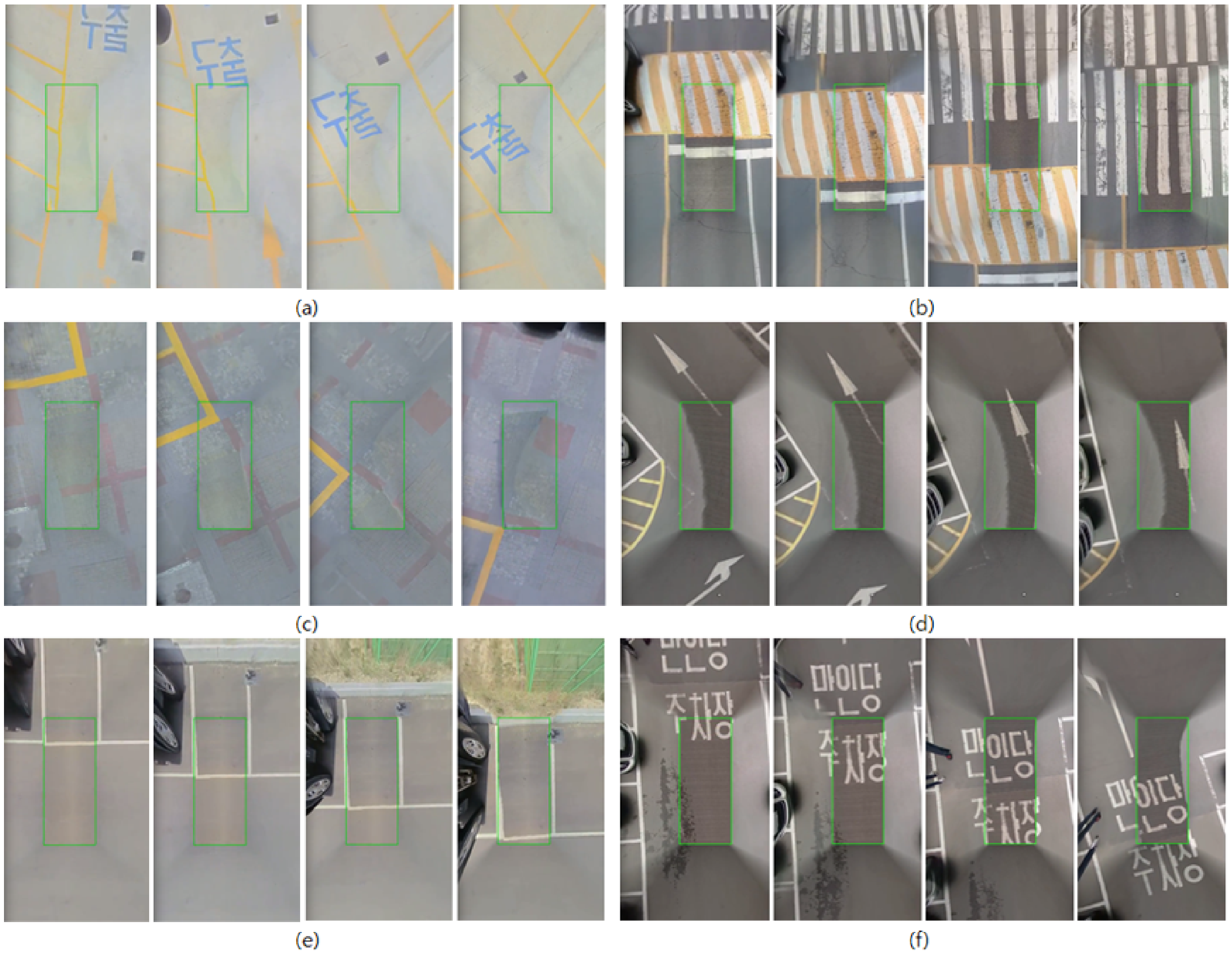


Figure 11. Experiment result to determine the number of RANSAC iterations.

Fig. 11 shows the results of an experiment to determine the number of iterations in the RANSAC method. Based on the maximum count obtained as a result of the RANSAC method when the number of iterations corresponds to 50, we measured the percentage of the maximum count for each iteration number. When the number of iterations corresponds


(a)

(b)

(c)

(d)

(e)

(f)

Figure 12. Example of the downward view of vehicle generation using AVM

to 10, it already satisfies more than 90% of the maximum count. Therefore, it is expected that the RANSAC method will be successful even when the number of iterations only corresponds to 10.

F. Execution Time

The execution time was measured on a 2.30-GHz Intel Core i5-6200U CPU with an 8-GB RAM laptop computer. The number of frames for the forward-backward error detection is 3, and the number of RANSAC iterations is 10.

Table III
EXECUTION TIME

	Preprocess	**Feature Tracking**	**Obstacle Filtering**	**Downward View Generation**	**Total**
Time (ms)	7.03	18.88	0.02	7.87	33.80

The total execution time corresponds to 33.80 ms per frame, namely 30 fps. Therefore, the complete process operates in real-time.

G. Result

Fig.12 shows the result of the proposed system. In order to ensure that the correct transformations are actually obtained, we select images with linear components such as lanes and images with information on the roads. Fig.12 (a) and (b) show the images obtained from cement and asphalt roads, respectively, and correspond to the results of the translation situation. It is observed that the straight line in the image is maintained as straight. Fig.12 (c) and (d) show the results obtained from the sway block and asphalt. The parking lines and directional lines maintain the shape well during the rotation. Fig.12 (e) and (f) show the images obtained from asphalt with obstacles on both sides of the ROI. There are obstacles although parking lines and letters on the road are composited correctly.

H. Discussions and Future Works

Fig.13 shows bad results due to unstable transformation. Fig.13 (a) shows the illumination of the indoor environment that does not match the situation of the proposed system. The experimental results of the indoor environment indicate that the reflected light causes the feature points to scatter or wash down. Fig.13 (b) shows the trembling of the transformations obtained from various obstacle height environments. It is necessary for obstacle filtering to be robust relative to various obstacles other than obstacles with the same height as the vehicle.

In the obstacle filtering stage, the histogram exhibits a different tendency based on the location of the vehicle's axis of rotation and the size and height of the object. We classify and respond to various situations by extending the histogram in three dimensions by using X gradient and Y gradient components in the histogram based on the current distance. Additionally, instead of obtaining the degree of change in feature points for each ROI, a method to determine the degree of change in the center point of both ROIs is used such that it is improved as robust relative to the change in the rotation axis.

The proposed method in the image synthesis stage determines the transformations in all the frames, and thus there is a high difference between the transformations and neighboring transformations when the roughness changes suddenly or the image is severely blurred. Therefore, we can use the average and variance of the transformations to determine the outlier for every frame and improve it by using the average transformation of the past several frames as opposed to the obtained transform. Additionally, to reduce the visual difference between the generated downward image and the AVM, in the case of lane and leader lines, the line segments are improved such that they are aligned and synthesized to lead to line segments.

VIII. Conclusion

In the study, we proposed a method to generate a downward image of current vehicle location by using commercial AVM. In the tracking stage, accuracy is improved by performing tracking by using good feature to track as a feature point suitable for the KLT tracker. In the outlier detection stage, the feature points on the ground are removed by using the change in the distance between feature points. The experimental results indicate that on average recall is 79.7% and precision is 88.9%, while eliminating the feature points captured by the object, maintaining the feature points on the road, and enabling stable sampling at the stage of obtaining the transformation. Finally, in the downward view generation stage, the refined feature point obtained through the above two stages are used, and the transformation matrix is estimated by using RANSAC.

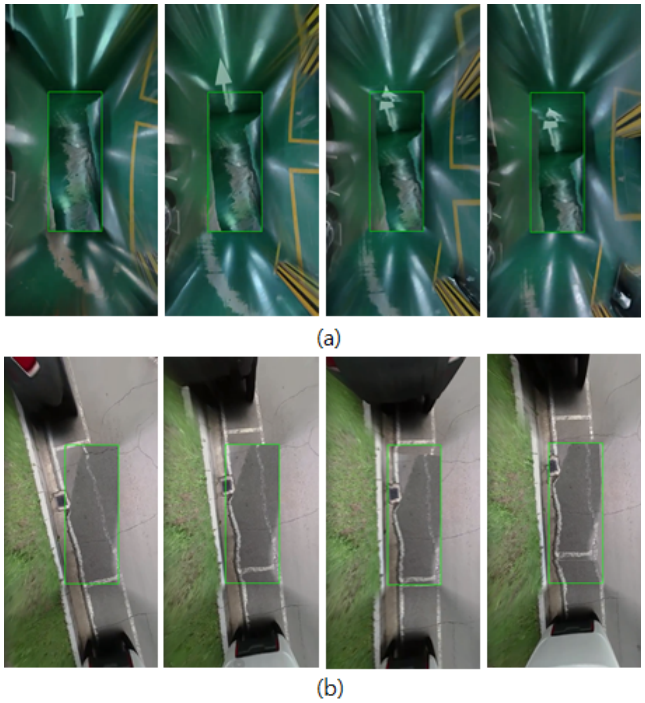


Figure 13. Example of Bad Results

References

[1] Nissan Motor Corporation, “Around view monitor.” [Online]. Available: https://www.nissan-global.com/EN/TECHNOLOGY/OVERVIEW/avm.html. [Accessed: 25-Oct-2018].

[2] S. Hecker, D. Dai, and L. V. Gool, “End-to-end learning of driving models with surround-view cameras and route planners,“ in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 435–453, 2018

[3] Y. C. Liu, K. Y. Lin, and Y. S. Chen, “Bird's-eye view vision system for vehicle surrounding monitoring,“ in *International Conference on Robot Vision*, 2008

[4] Y. Gao, C. Lin, Y. Zhao, X. Wang, S. Wei, and Q. Huang, “3-d surround view for advanced driver assistance systems,“ in *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 1, pp. 320–328, 2018.

[5] T. Ehlgen, T. Pajdla, and D. Ammon, “Eliminating Blind Spots for Assisted Driving,“ in *IEEE Transactions on Intelligent Transportation Systems*, vol. 9, no. 4, pp. 657-665, Dec. 2008.

[6] J. V. Dueholm, M. S. Kristoffersen, R. K. Satzoda, T. B. Moeslund and M. M. Trivedi, “Trajectories and maneuvers of surrounding vehicles with panoramic camera arrays,“ in *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 2, pp. 203–214, 2016.

[7] M. Bertozzi, L. Castangia, S. Cattani, A. Prioletti and P. Versari, “360 ° detection and tracking algorithm of both pedestrian and vehicle using fisheye images,“ in *IEEE Intelligent Vehicles Symposium (IV)*, pp. 132–137, 2015.

[8] C. Kum, D. Cho, M. Ra and W. Kim, “Lane detection system with around view monitoring for intelligent vehicle,“ in *International SoC Design Conference (ISOCC)*, pp. 215–218, 2013.

[9] D. Kim, B. Kim, T. Chung, and K. Yi, “Lane-level localization using an avm camera for an automated driving vehicle in urban environments,” *IEEE/ASME Transactions on Mechatronics, vol. PP*, no. 99, pp. 1–1, 2017.

[10] C. Lin and M. Wang, “Topview transform model for the vehicle parking assistance system,“ in *2010 International Computer Symposium (ICS2010)*, pp. 306–311, 2010.

[11] C. Wang, H. Zhang, M. Yang, X. Wang, L. Ye, and C. Guo, “Automatic Parking Based on a Bird's Eye View Vision System,” in *Advances in Mechanical Engineering*, vol.6, 2014.

[12] J. K. Suhr and H. G. Jung, “Sensor fusion-based vacant parking slot detection and tracking,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 1, pp. 21–36, 2014.

[13] Yu, M., Ma, G., “A visual parking guidance for surround view monitoring system,“ in *IEEE Intelligent Vehicles Symposium (IV).*, 2015.

[14] Press.bmwgroup.com, “Charging even easier than refuelling..” [Online]. Avilable: https://www.press.bmwgroup.com/global/article/detail/T0281369EN/charging-even-easier-than-refuelling?language=en. [Accessed: 25-Oct-2018].

[15] J. Shin, S. Shin, Y. Kim, S. Ahn, S. Lee, G. Jung, S. Jeon, and D. Cho, ”Design and Implementation of Shaped Magnetic-Resonance-Based Wireless Power Transfer System for Roadway-Powered Moving Electric Vehicles,” in *IEEE Transactions on Industrial Electronics*, vol. 61, no. 3, pp. 1179–1192, 2014.

[16] S. Y. Choi, B. W. Gu, S. Y. Jeong, and C. T. Rim, ”Advances in Wireless Power Transfer Systems for Roadway-Powered Electric Vehicles,” in

IEEE Journal of Emerging and Selected Topics in Power Electronics, vol. 3, no. 1, pp. 18–36, 2015.

[17] K. Y. Kim, Y. Ryu, E. Park, K. Song, and C. Anh, ”Analysis of misalignments in efficiency of mid-range magnetic resonance wireless power link,” in *Proceedings of the 2012 IEEE International Symposium on Antennas and Propagation*, pp. 18–36, 2015.

[18] E. Dubrofsky, “Homography estimation,” *Diplomová práce. Vancouver: Univerzita Britské Kolumbie*, 2009.

[19] S. Baker and I. Matthews, “Lucas-kanade 20 years on: A unifying framework,” *International journal of computer vision*, vol. 56, no. 3, pp. 221–255, 2004.

[20] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[21] H. Bay, T. Tuytelaars, and L. Van Gool, “Surf: Speeded up robust features,” in *European conference on computer vision*, pp. 404–417, Springer, 2006.

[22] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “Orb: An efficient alternative to sift or surf,” in *Computer Vision (ICCV), 2011 IEEE international conference on*, pp. 2564–2571, IEEE, 2011.

[23] Z. Zhang, “A flexible new technique for camera calibration,” *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, 2000.

[24] F. Devernay and O. Faugeras, “Straight lines have to be straight,” *Machine vision and applications*, vol. 13, no. 1, pp. 14–24, 2001.

[25] D. Scaramuzza, A. Martinelli, and R. Siegwart, “A flexible technique for accurate omnidirectional camera calibration and structure from motion,” in *Computer Vision Systems, 2006 ICVS’06. IEEE International Conference on*, pp. 45–45, IEEE, 2006.

[26] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[27] A. Levin, A. Zomet, S. Peleg, and Y. Weiss, “Seamless image stitching in the gradient domain,” in *European Conference on Computer Vision*, pp. 377–389, Springer, 2004.

[28] M. Brown and D. G. Lowe, “Automatic panoramic image stitching using invariant features,” *International journal of computer vision*, vol. 74, no. 1, pp. 59–73, 2007.

[29] B. Stenger, A. Thayananthan, P. H. S. Torr, and R. Cipolla, “Model-based hand tracking using a hierarchical bayesian filter,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 9, pp. 1372–1384, Sep. 2006

[30] P. J. Besl and N. D. McKay, “Method for registration of 3-D shapes,” in *Robot.-DL Tentative. Int. Soc. Opt. Photon*, pp. 586–606, Apr. 1992.

[31] S. Wang, J. Yue, and Y. Dong, “Obstacle detection on around view monitoring system,” in *Systems, Man, and Cybernetics (SMC), 2017 IEEE International Conference on*, pp. 1564–1569, IEEE, 2017.

[32] A. Goshtasby, “Image registration: Principles, tools and methods,” in *Springer*, 2012.

[33] P. Rousseeuw, N. Netanyahu, and D. Mount, “New statistical and computational results on the repeated median line,” in *University of Maryland, Center for Automation Research, Computer Vision Laboratory*, 1992.

[34] M. A. Fischler and R. C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography,” *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[35] M. Yu and G. Ma, “360 surround view system with parking guidance,” *SAE International Journal of Commercial Vehicles*, vol. 7, no. 2014-01-0157, pp. 19–24, 2014.

[36] B. Jeon, G. Park, J. Lee, S. Yoo and H. Jeong, “A Memory-Efficient Architecture of Full HD Around View Monitor Systems,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 6, pp. 2683–2695, 2014.

[37] C. Harris and M. Stephens, “A combined corner and edge detector.,” in *Alvey vision conference*, vol. 15, pp. 10–5244, Citeseer, 1988.

[38] J. Shi and C. Tomasi, “Good features to track,” tech. rep., Cornell University, 1993.

[39] Z. Kalal, K. Mikolajczyk, and J. Matas, “Forward-backward error: Automatic detection of tracking failures,” in *Pattern recognition (ICPR), 2010 20th international conference on*, pp. 2756–2759, IEEE, 2010.

[40] Vadas Co., Ltd., “Surround view.” [Online]. Avilable: http://vadas.co.kr/technology/surround-view/. [Accessed: 25-Oct-2018].

Jinkyu Lee was born in Chungju, Korea in 1993. He is currently pursuing the B.S. degree in computer science in the School Of Computer Science And Electrical Engineering at Handong Global University.

From 2016 to now, he is a research assistant with the computer graphics and vision laboratory. His current research interests are computer vision, ADAS, and simultaneous localization and mapping system.

Myungchul Kim was born in Iksan, Korea in 1994. He is currently pursuing the B.S. degree in computer science in the School Of Computer Science And Electrical Engineering at Handong Global University.

From 2016 to now, he is a research assistant with the computer graphics and vision laboratory. His current research interests are computer vision, ADAS, and simultaneous localization and mapping system.

Sangjun Lee was born in Incheon, Korea in 1993. He received a B.S. degree in Computer science and Engineering from Handong Global University, Pohang, Korea, in 2017. He is currently pursuing a M.S. degree in the Dept. of Information Technology at the Handong Global University.

From 2015 to now, he is a research assistant with the computer graphics and vision laboratory. His research interests include the SLAM system for the localization of self-driving cars, robotics, or users who use augmented reality, virtual reality, as well as 3D reconstruction, and optimization of these technologies using machine learning.

Mr. Lee received two best papers from Image Processing and Image Understanding in 2018 and The Korean Institute of Information Scientists and Engineers in 2017, with one patent in Korea.

Sungsoo Hwang was born in Busan, South Korea in 1983. He received B.S. degree in Electrical Engineering and Computer Science from Handong Global Unveristy, Pohang, South Korea in 2008, and M.S and Ph. D degrees in Korea Advanced Institute of Science and Technology, Daejeon, South Korea in 2010 and 2015, respectively.

From 2015 to now, he has been an Assistant Professor with School of Computer Science and Electrical Engineering, Handong Global University, Pohang, South Korea. His research interests include image-based 3D modeling, 3D data compression, augmented reality, and simultaneous localization and mapping system.