An Autonomous Driving System for Unknown Environments Using a Unified Map

Inwook Shim, *Student Member, IEEE*, Jongwon Choi, *Student Member, IEEE*, Seunghak Shin, *Student Member, IEEE*, Tae-Hyun Oh, *Student Member, IEEE*, Unghui Lee, Byungtae Ahn, *Student Member, IEEE*, Dong-Geol Choi, *Student Member, IEEE*, David Hyunchul Shim, and In-So Kweon, *Member, IEEE*

***Abstract*—Recently, there have been significant advances in self-driving cars, which will play key roles in future intelligent transportation systems. In order for these cars to be successfully deployed on real roads, they must be able to autonomously drive along collision-free paths while obeying traffic laws. In contrast to many existing approaches that use prebuilt maps of roads and traffic signals, we propose algorithms and systems using *Unified Map* built with various onboard sensors to detect obstacles, other cars, traffic signs, and pedestrians. The proposed map contains not only the information on real obstacles nearby but also traffic signs and pedestrians as virtual obstacles. Using this map, the path planner can efficiently find paths free from collisions while obeying traffic laws. The proposed algorithms were implemented on a commercial vehicle and successfully validated in various environments, including the 2012 Hyundai Autonomous Ground Vehicle Competition.**

***Index Terms*—Autonomous driving system, autonomous vehicle, detection for traffic information, local map representation.**

Manuscript received March 20, 2014; revised July 30, 2014 and October 22, 2014; accepted December 22, 2014. Date of publication February 20, 2015; date of current version July 31, 2015. This work was supported by the National Research Foundation of Korea Grant funded by the Korean Government (Ministry of Science, ICT, and Future Planning) under Grant 2010-0028680. The Associate Editor for this paper was U. Nunes. *(Corresponding author: In-So Kweon.)*

I. Shim, S. Shin, T.-H. Oh, B. Ahn, and D.-G. Choi are with the Robotics and Computer Vision Laboratory, Department of Electrical Engineering, College of Information Science and Technology, Korea Advanced Institute of Science and Technology, Daejeon 305-701, Korea (e-mail: iwshim@rcv.kaist.ac.kr; shshin@rcv.kaist.ac.kr; thoo@rcv.kaist.ac.kr; dgchoi@rcv.kaist.ac.kr; btahn@rcv.kaist.ac.kr).

J. Choi was with the Robotics and Computer Vision Laboratory, Department of Electrical Engineering, College of Information Science and Technology, Korea Advanced Institute of Science and Technology, Daejeon 305-701, Korea. He is now with Seoul National University, Seoul 151-742, Korea (e-mail: jwchoi.pil@gmail.com).

U. Lee is with the Flight Dynamics and Control Laboratory, Department of Aerospace Engineering, School of Mechanical Aerospace and Systems Engineering, College of Engineering, Korea Advanced Institute of Science and Technology, Daejeon 307-701, Korea (e-mail: lamer0712@kaist.ac.kr).

D. H. Shim is with the Center of Field Robotics for Innovation, Exploration, and Defense, Korea Advanced Institute of Science and Technology, Daejeon 307-701, Korea (e-mail: hcshim@kaist.ac.kr).

I.-S. Kweon is with the Department of Electrical Engineering, College of Information Science and Technology, Korea Advanced Institute of Science and Technology, Daejeon 307-701, Korea (e-mail: iskweon77@kaist.ac.kr).

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

Digital Object Identifier 10.1109/TITS.2015.2389237

I. Introduction

AUTONOMOUS driving technologies are expected to significantly improve driving safety and convenience by alleviating the burden of a driver, particularly under adverse conditions. Currently, they are implemented as a form of an advanced driver assistance system to partially aid drivers. It is also anticipated that, in the near future, fully autonomous cars will emerge as the key component of future transportation systems, replacing human drivers. In 2013, Mercedes Benz announced their plan to commercialize autonomous vehicles by 2020, and Nissan also announced their target year as 2020.

Autonomous driving technology took a quantum leap, owing to the Defense Advanced Research Project Agency (DARPA) Grand Challenge held in 2005, which required autonomous vehicles to drive in a 367-km off-road course in a desert without any help from outside [1]–[3]. The DARPA urban challenge, which was held in 2007, evaluated autonomous navigation technologies for urban environments such as merging, intersection handling, parking lot navigation, and lane change. The speed of vehicles was limited to 48.2 km/h for safety, and a very detailed Route Network Definition File about the competition environment (e.g., lane markings, stop signs, and special check points, with a high-resolution aerial image of the area) was given to participants [4]–[7]. Google increased the chance of the commercialization of autonomous vehicles by developing *Google cars*, i.e., the first licensed robot driver [8].

The VisLab Intercontinental Autonomous Challenge (VIAC) highlighted the reliability of vehicle-following autonomous driving, without any prior knowledge of the course, during a 13 000-km intercontinental trip [9], [10]. Considering the long distance covered, this experiment exhibited great autonomous driving performance. Although the competitions and the experiment were performed in challenging environments, the developed vehicles function when detailed information about the driving environment was provided. Hence, autonomous driving in traffic scenarios such as traffic lights and crosswalks, without prior information of the course, remains a challenge.

An autonomous vehicle should drive considering the overall situation, and many kinds of detection algorithms are necessary for autonomous driving in unpredictable real environments. In order to search for traffic information, we propose color-based detectors for artificial markers on the road and an obstacle detector based on Light Detection and Ranging (LiDAR) sensors, and we implement pedestrian and vehicle detectors. However, the outputs of many different detectors complicate

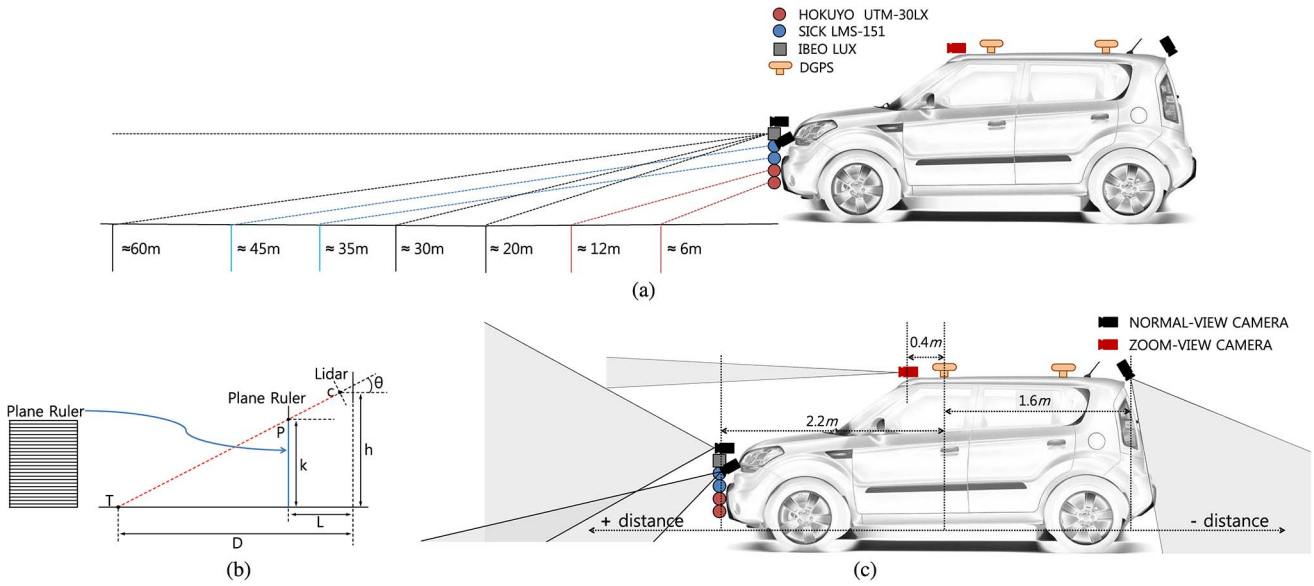


Fig. 1. System configurations of EURECAR. (a) LiDAR configuration. (b) LiDAR angle setting. (c) Camera configuration.

the decision-making process of the autonomous vehicle. To manage and consolidate the traffic information obtained from various detection algorithms, we also propose the *Unified Map* representation.

Unified Map converts the information of traffic environments into imaginary obstacles. Since this map represents the results from various algorithms as obstacles, the behavior of the vehicle can be easily determined by a path planner with only local information. The path planner can be also easily applied to an autonomous driving system without complicated decision rules. The structure of Unified Map is similar to existing map representations [11]–[14]. Although these map representations only deal with the geometric information about the environment, our map representation additionally uses high-level information (e.g., traffic information) for the path planning of autonomous vehicles.

We developed a test driving vehicle, i.e., *EURECAR*, to implement the proposed system for autonomous driving in unknown environments. In this paper, we describe the Unified Map representation, real-time algorithms for detecting traffic events, and the system architecture. This integrated system successfully completed the full course of the 2012 Autonomous Vehicle Competition (AVC), which was organized by the Hyundai–Kia Motor Group.

The remainder of this paper is organized as follows. The hardware architecture and overall system of EURECAR are introduced in Section II. Unified Map is defined in Section III. The sensor coordinate configuration of EURECAR and the detection algorithms in the modular architecture are described in Section IV. The path planner and path collision detection are explained in Section V. The experimental result in the AVC is presented in Section VI. This paper is concluded in Section VII.

II. System Overview

A. System Hardware Configuration

The sensor system has a very important role in autonomous driving. As shown in Fig. 1(a), to find drivable areas and obstacles, four single-scan LiDAR sensors and one quadruple-scan LiDAR sensor were mounted on a custom-made front rack. There are three kinds of LiDAR sensors to cover the frontal area of the vehicle. Two *HOKUYO UTM-30LX* units cover up to 30 m, two *SICK LMS-151* units cover up to 50 m, and one *IBEO LUX* unit covers up to 200 m.

For constructing multiple scan lines, the forward-scanning LiDAR sensors should be slightly tilted at different angles. However, it is difficult to directly adjust the tilt angle of each LiDAR sensor. As shown in Fig. 1(b), to direct each LiDAR sensor toward point $\mathbf{T}$, we designed a plane ruler. The plane ruler was devised to measure the vertical distance from the ground to a scan line that is projected by a LiDAR sensor. If we know point $\mathbf{P}$ on which $\mathbf{T}$ is projected, we can easily adjust angle θ using a simple proportional formula, i.e., $k = ((D - L) \times h)/D$. Although this method does not adjust the tilt angle accurately, it is satisfactory for the sensor configuration in a small working space. All LiDAR sensors were adjusted to scan approximately 6, 12, 20, 30, 35, 45, and 60 m of the road surface in front of the vehicle, as shown in Fig. 1(a). The farthest area is scanned using IBEO LUX, which has four scan lines covering 3.2° vertically, so that its highest scan line is parallel to the ground plane and the next line scan at about 60 m of the road. The largest measuring range was determined considering EURECAR's maximum braking performance (60 m of skidding distance when full brake is applied at 60 km/h) so that the vehicle can stop before colliding with the object detected by the farthest scanning laser. Other six scanners were adjusted to cover the rest of the area as evenly as possible.

As shown in Fig. 1(c), for detecting events on roads and classifying obstacles, four color cameras are used, i.e., one front-view camera, one road-view camera, one zoom-view camera, and one rear-view camera. We use *Flea3 Gigabit Ethernet* cameras, and the cameras use a *Sony ICX424 CCD* 1/3″ sensor that captures 640 × 480 resolution images at 80 Hz with a 58.72-dB dynamic range. The front-view and rear-view cameras were equipped with the same lenses, with a fixed focal length of

TABLE I
RANGE CAPABILITY OF THE VISION ALGORITHMS FROM THE FRONT DGPS

Algorithms	Cameras		
	Front and zoom	Road	Rear
Traffic Light	$10.5 \sim 69.5\,m$	–	–
Crosswalk	–	$2.5 \sim 34.5\,m$	–
Barrier gate	$2.5 \sim 26\,m$	–	–
Parking slot	–	–	$-1.7 \sim -8.4\,m$
Road lane	–	$2.5 \sim 29.5\,m$	–
Pedestrian	$5.5 \sim 97\,m$	–	–
Vehicle	$4.5 \sim 63\,m$	–	–

3.5 mm and a $77^\circ \times 57.7^\circ$ field of view (FoV), which indicates horizontal and vertical angles, respectively. The road-view camera was equipped with a lens with a fixed focal length of 6.0 mm and a FoV of $44^\circ \times 33^\circ$. The zoom-view camera was equipped with a lens with a fixed focal length of 12.0 mm and a FoV of $21^\circ \times 16^\circ$. The front-view camera was mounted on the front rack for general detection. The road-view camera, which was slightly tilted downward, was also mounted on the front. The zoom-view and rear-view cameras were mounted on the roof rack. The zoom-view camera was installed to allow long-range detection, and it assists the front-view camera.

The range capability of each camera is shown in Table I. The ranges are determined by camera intrinsic parameters, the height of the cameras, the sensor resolution, and the performance of the proposed detection algorithms. The algorithms are presented in Section IV.

The entire autonomous driving system was integrated on *Kia Soul*, which is a compact sports utility vehicle. The vehicle, which is named EURECAR, carries a computing system in its trunk. We installed a shock-absorber mounted PC rack that carries three PCs, one navigation system, and one electronic control unit (ECU) for motor control. The navigation system [15] integrates an inertial measurement unit (IMU) and two differential GPS (DGPS) units using an extended Kalman filter (EKF), and it was implemented on a PC104 computer running on a real-time operating system (OS), i.e., QNX.

B. System Architecture Flow

Our autonomous driving system has a modular architecture using a User Datagram Protocol (UDP) Ethernet network. All modules are integrated via the UDP multicast protocol and the National Instruments Publish and Subscribe Protocol (NI-PSP). NI-PSP was used for communication between the *Path Planner* and the *Motor Controller*. For other parts, the UDP multicast is used as the main network protocol. Each module processes its own tasks asynchronously. It makes the system extensible and reduces the effort needed to synchronize various kinds of modules. Herein, we describe the module structure in a hierarchical order.

The sensor interface modules (denoted as *S-Net* in Fig. 2) are on the lowest layer in order to subscribe raw data streams (images and range data) from the sensors at their own paces. The interface modules compress or modify the subscribed raw data so that it could be easily utilized at the next layer.

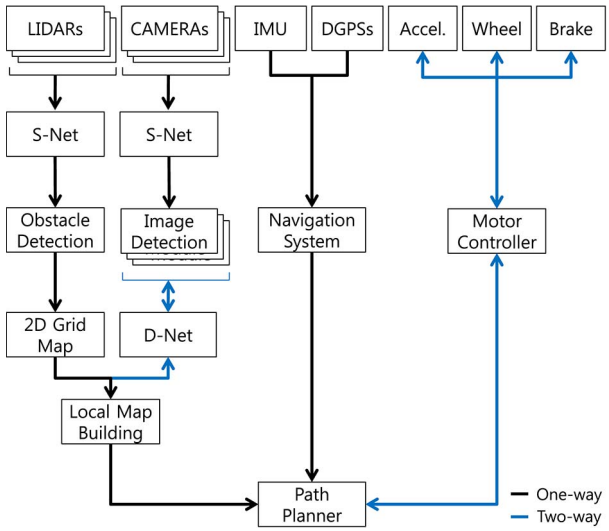


Fig. 2. Overview of the proposed modular architecture.

Given the processed data from S-Net, the detection modules (denoted as *Obstacle Detection* and the set of *Image Detection* modules in Fig. 2) find objects or events on the road such as obstacles, lanes, traffic lights, crosswalks, barrier gates, pedestrians, and vehicles. The modules report the outputs to the *Local Map Building* module via *D-Net*. The outputs are represented as the coordinates on the *Local Map*. For each detection algorithm, we provide detailed explanations in Section IV. Each module selectively subscribes the sensor data from S-Net by virtue of the multicast protocol. While the Image Detection modules are connected to the Local Map Building module via a network, the Obstacle Detection module is directly connected to the Local Map Building. In particular, all the Image Detection modules can access the latest output of other detection modules, including Obstacle Detection from the Local Map, through D-Net.

The Local Map Building module extends the metric *2D Grid Map* from the Obstacle Detection module to an informative local map by combining the results from the Image Detection modules. The specific representations for the local map are presented in Section III.

The Path Planner module consists of a path tracker and a path modification algorithm. The planner generates paths based on the local map information, the pose information from the *Navigation System*, and the feedback error from the built-in ECU (denoted as Motor Controller in Fig. 2). The Motor Controller module controls the steering wheel, the accelerator, and the brake following the generated path.

All the detection modules can be easily replaced by other novel algorithms, or an additional detection module can be also added if the consistency in the communication protocol between the detection modules and the Net modules is guaranteed. However, the more detection modules are added to the system, the more computation power is required. There is a tradeoff between the number of algorithm modules that can be loaded and the processing speed of the system. The number of the modules would be empirically determined by the capacity of the processor used. In order to determine the right number of modules to run on each computer, we considered the following

three factors: the computing power, the network traffic, and the memory usage of the modules. The modules were assigned to each computer so that the usage of any of these factors does not exceed 90% of the limit. The detailed resource allocation is automatically handled by the OS scheduler [16], [17] (one can manually assign the module to a designated processor following the work in [18]).

III. Unified Map Representation

In this section, we present the Unified Map representation that includes varying information for autonomous driving in real environments.

In unpredictable real environments, autonomous vehicles have to detect obstacles that obstruct its way and recognize unpredictable events such as a change in traffic lights. For detecting and recognizing all these cases, many detection algorithms are required. However, different algorithms produce their results in different forms. From the perspective of the path planner, it is hard to utilize a map that consists of several forms of detection results (e.g., the positions of obstacles, events such as changing traffic lights, and artificial markers on the road), and it is also difficult to generate paths considering all situations obtained from various detection algorithms.

In order to efficiently handle all such cases, we propose a local map generation by converting the results of detectors into obstacles. We call the generated local map Unified Map. The obstacles in Unified Map can be categorized as "real" and "imaginary" obstacles, i.e., real obstacles are detected by the Obstacle Detection module, and imaginary obstacles are converted from the results of the Image Detection modules. The base structure of Unified Map originates from the Obstacle Detection module. In the base structure, obstacles such as pedestrians, vehicles, and poles are represented as real obstacles, and detected traffic events, such as changing traffic lights, are augmented to the base structure as imaginary obstacles.

The path planner generates a local path using the shape of real and imaginary obstacles. The shapes of obstacles depend on the types of the detected events. Unified Map generates imaginary obstacles to stop path tracking by completely blocking around the vehicle when the vehicle should stop or by blocking a path when the vehicle has to make a detour.

For example, when the vehicle should stop due to a traffic-signal event, an imaginary obstacle forms a rectangular block around the vehicle to let the path planner generate the stop path only. For detour events such as a direction signal, the opposite side will be blocked by imaginary obstacles to make the path planner generate a detouring path. For more illustrations and for the details of the path planner, see Section V.

IV. Detection Modules

In this section, we describe the coordinate setup for the sensor configuration and the detection algorithm modules for LiDAR sensors and cameras, which make it possible to deal with obstacles, traffic lights, crosswalk makers, barrier gates, pedestrians, vehicles, parking slots, and road lanes.

TABLE II
Image Source Usage

Image Source	Modules
Front-view image Zoom-view image	Traffic light, barrier gate, pedestrians, and vehicles detection modules
Road-view image	Crosswalk marker and road lanes detection module
Rear-view image	Parking-slot detection module

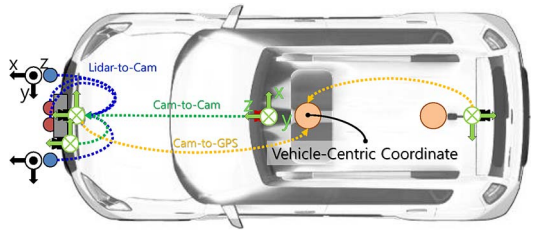


Fig. 3. Sensor coordinate configuration.

The parameters for most of the detection algorithms were determined based on our campus data set with various scenes. The algorithms were empirically set to satisfy conditions with a true positive ratio of the detectors at 0.8 in our Korea Advanced Institute of Science and Technology (KAIST) campus data set, except for the parameters of the parking slot and road lane detection modules. For more details, see Section IV-B.

A. Sensor Coordinate Configuration

As shown in Table II, since the data sources for each algorithm are different, the coordinates of each sensor have to be transformed into a vehicle-centric coordinate. The extrinsic parameters between sensors were estimated by the conventional extrinsic calibration method [19]. The vehicle-centric coordinate is set to equal the coordinate of the front DGPS. Our coordinate setup is shown in Fig. 3.

B. Obstacle Detection Module

In order to find an obstacle-free path, we detect obstacles on the road using LiDAR sensors. With an assumption that the road is flat, any object detected from a single LiDAR sensor with a certain height is easily regarded as an obstacle. However, there are some topography and objects that have height values higher than the predefined threshold value but which do not disturb the driving of the vehicle (i.e., hills and speed bumps). These kinds of topography and objects should be distinguished from real obstacles that physically obstruct the way.

As described in Section II-A, since the tilt angles of the LiDAR sensors were adjusted for scanning points in the front of the vehicle at certain intervals, the objects can be classified as obstacles when the slopes of the objects and the maximum height difference between adjacent scanning points are larger than a certain slope threshold ϕ and height threshold ω, respectively. This is illustrated in Fig. 4.

For faster processing, we project 3-D point clouds to a 2-D grid map rather than processing thousands of points from eight scan lines. Every cell of the 2-D grid map contains three values, i.e., a maximum height value $Z_{\max}$, a minimum

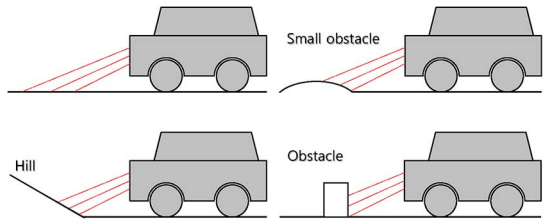


Fig. 4. Objects on the road.

height value $Z_{\min}$, and a horizontal distance value D between the maximum and minimum height points. The obstacle cells are distinguished from the others by height difference $V = |Z_{\max} - Z_{\min}|$ and slope angle $S = V/D$ in each cell. If height difference V and slope angle S are larger than height threshold ω and slope angle threshold ϕ, respectively, the cell is classified as an obstacle cell. The obstacle cells are grouped using K-nearest neighbor segmentation [20]. The 2-D grid map is expressed as follows:

$$\text{MAP}^t = \left\{\text{Cell}^t_{x,y}\right\}, \text{Cell}^t_{x,y} = \left\{Z^t_{\max}, Z^t_{\min}, D^t, G^t\right\} \quad (1)$$

where x and y are the cell coordinate indexes of the map, G is the group index, and t is the time index. In our implementation, considering the scan range of the LiDAR sensors, we set a 32 m $\times$ 128 m grid map with a 0.25-m resolution, $\phi = 30^\circ$, and $\omega = 0.228$ m. Variables ϕ and ω were determined in accordance with the radius of the equipped tires and the drivable maximum slope angle of EURECAR. MAP^t is the base structure of Unified Map.

C. Traffic-Light Detection Module

For detecting traffic lights, our proposed algorithm uses the probabilistic template matching method suggested by Levinson *et al.*, [21]. They showed a good result when checking the state of traffic lights with prior information, such as the scale and position of traffic lights. Otherwise, the proposed algorithm detects both the position and state of traffic lights without any prior information. The overall process is similar to that in [21]; however, the template shape is simplified for achieving real-time performance, and the multiscale kernels are applied to cover the various scales of traffic lights. The multiscale kernels are utilized to approximate the approximate distance to the detected traffic light.

We use a hue–saturation–value (HSV)-domain image because it is robust in a variety of illumination conditions. Since a traffic light is a luminous object, its color consistency is quite robust in the HSV domain. As a preprocessing step, we extract a probabilistic image $\mathbf{G}$. The image represents the candidate regions of traffic lights within an image $\mathbf{G}$, which is described as

$$\mathbf{G}_{i,j}(\mathbf{H}, \mathbf{S}, \mathbf{U}) = \frac{\mathbf{U}_{i,j} \sum_{k=1}^{h_{\mathbf{T^c}}} \sum_{l=1}^{w_{\mathbf{T^c}}} L_{i,j,k,l}(\mathbf{H}, \mathbf{S})}{2^{16} h_{\mathbf{T^c}} w_{\mathbf{T^c}}} \quad (2)$$

where i and j are pixel points; $h_{\mathbf{T^c}}$ and $w_{\mathbf{T^c}}$ denote the height and width, respectively, of template $\mathbf{T^c}(r_0)$ in (4); $\mathbf{H}$ and $\mathbf{S}$ are the *hue* and *saturation* of the HSV domain, respectively; $\mathbf{U}$ is a weight matrix for maximizing the luminous characteristic in the HSV domain; and function L represents the color constraints of the traffic light, which are described in detail as follows.

The $\mathbf{U}$ image represents the initial probabilistic image, which is generated by heavily weighting the saturation and high-*value* pixels. The pixel intensity of the $\mathbf{U}$ image indicates a probabilistic value for being at the center position of a traffic light. This is described as

$$\mathbf{U}_{i,j}(\mathbf{S}, \mathbf{V}) = \begin{cases} \left(\mathbf{S}_{i,j}{}^5/2^{32}\right) \times 1, & \text{for } \mathbf{V}_{i,j} \le \theta_{\mathbf{V}} \\ \left(\mathbf{S}_{i,j}{}^5/2^{32}\right) \times \mathbf{V}_{i,j}, & \text{otherwise} \end{cases} \quad (3)$$

where $\mathbf{S}$ and $\mathbf{V}$ are the saturation and value of the HSV domain, respectively.

Function L in (2) includes the color constraints and template $\mathbf{T^c}$. For applying the color constraints for traffic lights, the predefined hue range $\mathbf{R_h}$ and saturation threshold $\theta_{\mathbf{S}}$ are used. The Gaussian-filtered template $\mathbf{T^c}(r_0)$ with radius r_0 is also used for removing sparse noise. It is described as

$$L_{i,j,k,l}(\mathbf{H}, \mathbf{S}) = \frac{H_w\left(\mathbf{H}_{u(k,l)}\right) S_w\left(\mathbf{S}_{u(k,l)}\right)\left(1 - \delta(\mathbf{T^c}_{k,l}(r_0), 0)\right)}{\max\{\mathbf{T^c}(r_0)\} - \mathbf{T^c}_{k,l}(r_0) + 1} \quad (4)$$

where

$$H_w(x) = \begin{cases} 1, & \text{if } x \in \mathbf{R_h} \\ 0, & \text{otherwise} \end{cases}$$

$$S_w(x) = \begin{cases} 1, & \text{if } x > \theta_{\mathbf{S}} \\ 0, & \text{otherwise} \end{cases}$$

$$\delta(a, b) = \begin{cases} 1, & \text{if } a = b \\ 0, & \text{otherwise} \end{cases}$$

where $u(k, l)$ indicates the position of $(i + k - h_{\mathbf{T^c}}/2, j + l - w_{\mathbf{T^c}}/2)$, and $\mathbf{T^c}_{k,l}(r_0)$ is a value of (k, l) on template $\mathbf{T^c}(r_0)$.

Since the candidate regions of image $\mathbf{G}$ can include noncircular outliers, multiscale kernels $\mathbf{C^c}$ are convoluted for eliminating the shapes. If there is an object with a circular shape and color that could be a traffic light, the response of $\mathbf{C^c}$ is high. The traffic light is detected at the maximal extreme response of the kernels. In addition, the distance to the traffic light is approximated by the size of the maximum response kernel, which is described as

$$\mathbf{R}_{i,j,r} = \varphi\left(\frac{\sum_{k=-2r}^{2r} \sum_{l=-2r}^{2r} \mathbf{C^c}_{k,l}(r)(\mathbf{G}_{i+k,j+l})}{\pi r^2}, \theta_{\mathbf{R}}\right) \quad (5)$$

where

$$\mathbf{C^c}_{k,l}(r) = \begin{cases} 0, & \text{if } 2r < \|(k, l)\|_2 \\ 1, & \text{if } \|(k, l)\|_2 \le r \\ -\lambda, & \text{otherwise} \end{cases}$$

$$\varphi(a, b) = \begin{cases} 1, & \text{if } a \ge b \\ 0, & \text{otherwise} \end{cases}$$

where $\theta_{\mathbf{R}}$ is a threshold for removing the noncircular shape, λ is a parameter adjusting the circular degree of the detected traffic lights, and r indicates a set of sizes of the multiscale kernel.

The arrow traffic-light detection is very similar to the traffic-light detection, except for the shape of template $\mathbf{T^c}$ and multikernel $\mathbf{C^c}$. As shown in Fig. 6(a), for finding arrow traffic lights in image $\mathbf{G}$, weighting matrix $\mathbf{U}$, function L, and the barlike template $\mathbf{T^b}$ are used. There are two types of multiscale kernels $\mathbf{C^b}$, i.e., one for a left directional arrow and another for a right directional arrow.

In our implementation, the parameters were set as follows: the height $h_{\mathbf{T^c}}$ and width $w_{\mathbf{T^c}}$ of template $\mathbf{T^c}(r_0)$ were both set to 24, and the radius r_0 of the template was 6. The values previously suggested [21] were used to set $\theta_{\mathbf{V}}$, $\mathbf{R_h}$, and $\theta_{\mathbf{S}}$. The parameters for applying the multiscale kernels' step were set as follows: $\theta_{\mathbf{R}}$ was set to 200, λ was -3, and r consisted of a set of integer values from 5 to 12.

When the red traffic light is detected, a rectangle is generated around the vehicle in Unified Map as the imaginary obstacle. The position of the front edge of the rectangle is determined by the distance to the traffic light. In order to handle arrow lights, if a left arrow is detected, an imaginary obstacle is generated on the right side of the vehicle so that the path planner is encouraged to generate a collision-free path on the left side of the traffic light and the vehicle.

D. Crosswalk Marker Detection Module

A crosswalk marker is detected using its repeated patterns. Foucher *et al.*, [22] showed quite good results by extracting the repetition of similar connected components in a bird's-eye-view image. However, the approach of Foucher *et al.*, does not work well when pitching motions occur by the braking and acceleration of the vehicle. To obtain robustness against the pitching motion of the vehicle, our proposed algorithm tries to detect the repeated patterns of a crosswalk in the original image.

A 1-D mean filter, which is parallel to the x-axis in the image coordinates, is convolved with the $\mathbf{V}$ of the HSV domain. The length of the filter is set to be larger than the intervals between two nearby white lines of the crosswalk. Since the intervals appear to become wider as a crosswalk comes closer to the camera, the length of the filter is proportional to the y-coordinate in the image. This process is described as

$$\begin{aligned}\mathbf{F}_{x,y} &= \sum_{k=1}^{w} \mathbf{M}_y(k)\mathbf{V}(x+k-w/2, y)\\ \mathbf{M}_y &= [1/w_y, 1/w_y, \ldots, 1/w_y]\\ w_y &= w_{\min} + \frac{w_{\max}-w_{\min}}{h}y \end{aligned} \tag{6}$$

where $\mathbf{F}$ is a filtered image, $\mathbf{M}_y$ is the 1-D mean filter for the yth row of the image, w_y is a length of $\mathbf{M}_y$, h is the height of the image, and $w_{\min}$ and $w_{\max}$ represent the minimum and maximum lengths of the 1-D mean filter, respectively.

Two binary images, i.e., $\mathbf{B_U}$ and $\mathbf{B_L}$, are obtained from a difference image $\mathbf{D}$ representing the difference between image $\mathbf{V}$ and filtered image $\mathbf{F}$. A pixel of $\mathbf{B_U}$ is assigned "1," where the value of a pixel in image $\mathbf{D}$ is larger than θ_B. A pixel of $\mathbf{B_L}$ is assigned "1," where the value of a pixel in image $\mathbf{D}$ is smaller than $-\theta_B$. These two binary images are denoised by erosion, dilation, and blob labeling. The two denoised binary images are combined by a bitwise OR operation, and the combined binary image is noted as $\mathbf{B_S}$.

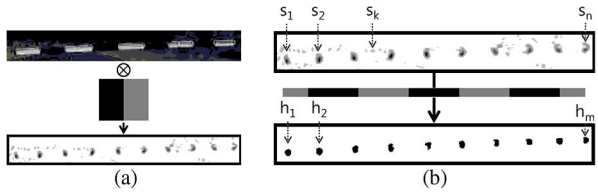


Fig. 5. Filters for barrier gate detection. (a) x-derivative filter. (b) Square-wave mask.

If there is a crosswalk marker, a bandlike binary shape appears in image $\mathbf{B_S}$. The crosswalk marker is detected by applying a 2-D mean filter to image $\mathbf{B_S}$. The width of the filter is equal to the width between two nearby road lanes, and the height of the filter is proportional to the y-coordinate. Finally, when the result of the filter is larger than θ_C, a crosswalk marker is detected at that position. The whole process is shown in Fig. 6(b).

In our implementation, θ_B and θ_C were set to 13 and 200, respectively. The sizes of the 1-D and 2-D mean filters were linearly distributed from the minimum and maximum lengths of the crosswalks. The minimum and maximum lengths of the 1-D mean filter were set to 50 and 420 pixels, respectively, and the minimum and maximum sizes of the 2-D mean filter were set to 500×4 and 560×80 pixels, respectively.

The position of the crosswalk is used for correcting the position of the front edge of the rectangle generated by the traffic-light detector in Unified Map.

E. Barrier Gate Detection Module

The barrier gate has a strong characteristic, i.e., a repeated color pattern. Using this characteristic, we propose a novel algorithm for detecting this color pattern. The barrier gate generally has a repetition of two colors, i.e., in our case, yellow and black. For maximizing the difference between these two colors, the red–green–blue domain is converted to a yellow saliency image $\mathbf{Y}$, i.e., $\mathbf{Y}_{i,j} = R_{i,j} + G_{i,j} - 2B_{i,j}$.

As shown in Fig. 5(a), an x-derivative filter is convolved on image $\mathbf{Y}$ to obtain filtered image $\mathbf{F}$, representing the horizontal changes. If a barrier gate is in image $\mathbf{F}$, the image has linearly repeated patterns.

The repeated patterns consist of the set of connected components $\mathbf{S}$ segmented by blob labeling, i.e., $\mathbf{S} \in \{\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_n\}$, where n is the number of segments. As shown in Fig. 5(b), the square-wave mask is generated. Two end points of the mask are selected from $\mathbf{S}$. The number of squares of the mask is fixed, and the width of one square and the angle of the mask are determined by the two selected components. The mask is applied to all combinations of the components of set $\mathbf{S}$, and the response of the mask would be high when there are linearly repeated patterns between the two ends.

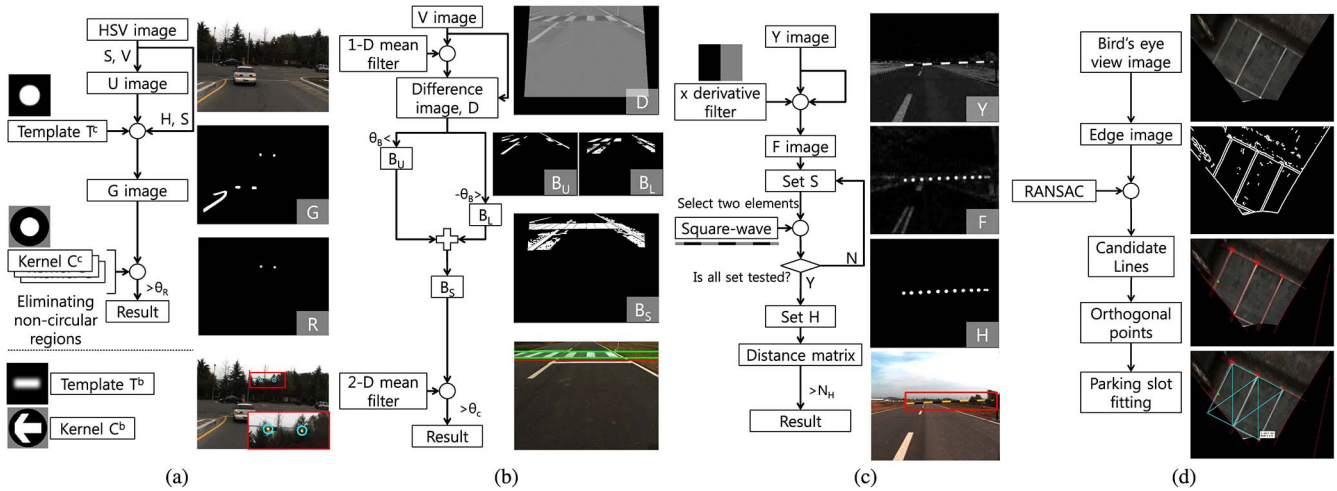


Fig. 6. Pipelines of the detection modules. (a) Traffic light. (b) Crosswalk markers. (c) Barrier gate. (d) Parking slot.

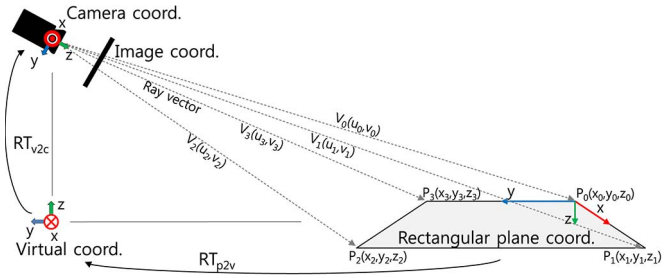


Fig. 7. Calibration between the rear camera and plane surface.

Hypothesis set $\mathbf{H}$ represents the set of masks with responses larger than a threshold t, i.e., $\mathbf{H} \in \{\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_m\}$. Each $\mathbf{h}$ is represented by three features, i.e., the y-coordinate in the image, the angle, and the length of the square-wave mask. Hypothesis $\mathbf{H}$ is clustered by the Euclidean distance in the feature space, and the barrier gate is detected when the number of elements of a subset is larger than a threshold, i.e., $\mathbf{N_H}$. The whole process is shown in Fig. 6(c).

In our implementation, t was set to 0, and $\mathbf{N_H}$ was 2. We ignore the component pairs when their Euclidean distance is less than 60 pixels for computational efficiency.

The barrier gate detector plays the same role as the traffic-light detector in Unified Map. When a barrier gate is detected, an imaginary rectangular obstacle is generated to stop the vehicle.

F. Parking Slot Detection Module

The rear-view camera is used for detecting parking lines. As part of a preprocessing sequence, the images are transformed to bird's-eye-view images. As shown in Fig. 7, for calibrating the parameters between an image coordinate of the rear camera $\mathbf{I_C}$ and the coordinate of rectangular plane coordinate $\mathbf{P_C}$, a rectangular plane and a virtual coordinate $\mathbf{V_C}$ are set.

The four ray vectors $\mathbf{V}$ corresponding to four corner points $\mathbf{P}$ are calculated by camera intrinsic and radial distortion parameters. Since the size of the rectangular plane is known, the corner positions in $\mathbf{P_C}$ is set to $P_0(0,0,0)$, $P_1(0,w,0)$, $P_2(h,w,0)$, and $P_3(h,0,0)$, where w and h are the width and height of the rectangular plane, respectively.

The initial parameters of $\mathbf{RT_{v2c}}$ are determined by measuring the height and tilting angle of the rear-view camera. Supposing that a surface is a ground plane, the initial relative pose $\mathbf{RT_{p2v}}$ consists of five parameters, which are 2-D rotation and 3-D translation parameters. Then, the initial $\mathbf{RT_{p2v}}$ parameters are estimated by ray vectors $\mathbf{V}$ and their corresponding corner points $\mathbf{P}$ as

$$\begin{aligned} \hat{\mathbf{P}} &= [\mathbf{R_{p2v}}|\mathbf{t_{p2v}}]\,([\mathbf{R_{v2c}}|\mathbf{t_{v2c}}]\mathbf{P}) \\ \mathbf{V} &= (\hat{\mathbf{x}}/\hat{\mathbf{z}}, \hat{\mathbf{y}}/\hat{\mathbf{z}}) \end{aligned} \tag{7}$$

where $\mathbf{R_{p2v}}$ and $\mathbf{t_{p2v}}$ are the rotation matrix and translation matrix of $\mathbf{RT_{p2v}}$, respectively, $\mathbf{P}$ is a vector of four 3-D points, and $\hat{\mathbf{P}}(\hat{\mathbf{x}}, \hat{\mathbf{y}}, \hat{\mathbf{z}})$ is a position of $\mathbf{P}$ in camera coordinate $\mathbf{C_C}$. The initial $\mathbf{RT_{p2v}}$ is obtained from (7).

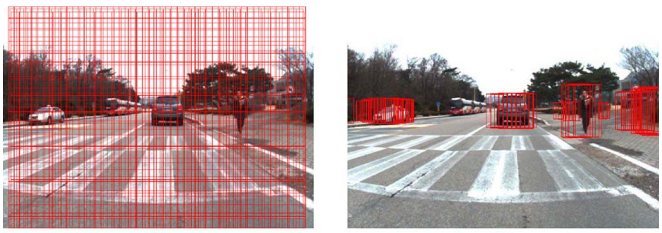
The Levenberg–Marquardt method [23] is then applied to optimize the extrinsic parameters, i.e., $\mathbf{RT_{v2c}}$ and $\mathbf{RT_{p2v}}$, by minimizing the errors between ray vector $\mathbf{V}$ and a vector generated by the extrinsic parameters and corner points $\mathbf{P}$. Zhang's method [24] can be used to estimate the camera intrinsic and radial distortion parameters. The bird's-eye-view image can be generated using the optimized extrinsic parameters.

Edge image $\mathbf{E}$ is generated from a bird's-eye-view image using canny edge detection, as shown in Fig. 6(d). Random sample consensus (RANSAC)-based line fitting is then used to extract all candidate lines. The candidate corner points of parking lines are obtained from the intersections of the candidate perpendicular lines, and the perpendicularity was estimated as

$$\arctan\left(\left|\frac{m_1 - m_2}{1 + m_1 m_2}\right|\right) = \alpha \tag{8}$$

where m_1 and m_2 are the slopes of the lines, and α is the angle between the two lines.

The parking slot corners are selected by the constraint of the distance between the candidate corners because the size of a parking slot is known. If the seed point in the target parking



(a) (b)

Fig. 8. Example of the candidate search space. (a) Full search space. (b) Reduced search space.

slot is not given, at least three corners are required to detect the target slot. In this case, the parking slot model is fitted to the three points. When the seed point is given, the parking slot can be detected with two corner points. The slot is fitted to two corner points, containing the seed point. The seed point is a GPS point for indicating a center position of a target parking slot.

In Unified Map, the parking slot detector generates a rectangular shape with one of the sides open, which will be the entrance to the detected parking slot.

G. Pedestrian and Vehicle Detection Module

For detecting pedestrians and vehicles, we used a HOG + SVM detector provided by OpenCV.

A general framework for pedestrian and vehicle detectors is the sliding-window-based algorithms. The computational time of this type of detector is large due to the size of the candidate search space. The works in [25] and [26] are representative studies that aimed to address this problem. Fast object detection with coarse-to-fine search [25] is a popular method, but it is not fast enough for rapidly moving vehicles. In one study [26], the detection speed is dramatically accelerated, but this method requires a stereo-based camera system.

As described in Section IV-A, since we know the relative pose between the vehicle-centric coordinate and each sensor coordinate, the number of candidate bounding boxes in the image coordinates can be reduced by the obstacle detection module. The reduced number of candidate boxes has the effect of rejecting many false positives (FPs) and improving the speed of the detectors. Fig. 8 shows why the performance of the detectors is improved by reducing the candidate search space and estimating the scale of pedestrians and vehicles.

Assuming an image of 640×480 pixels, the sliding windows are positioned in steps of (1/10) $\times$ the width of windows or the height, and six different scales are used; the number of full search windows is about 25 800. On the other hand, if the distance of obstacles from the vehicle-centric coordinate is known, we can fix the scale and size of the bounding boxes. In this case, the number of search windows is 0–276.

In our implementation, for pedestrian detection, the minimum and maximum sizes of the bounding box were set to 30×15 and 160×80, respectively. For vehicle detection, the minimum and maximum sizes of the box were set to 53×53 and 206×206, respectively. The threshold values of the support vector machine (SVM) for pedestrians and vehicles were 1.5 and 1.45, respectively.

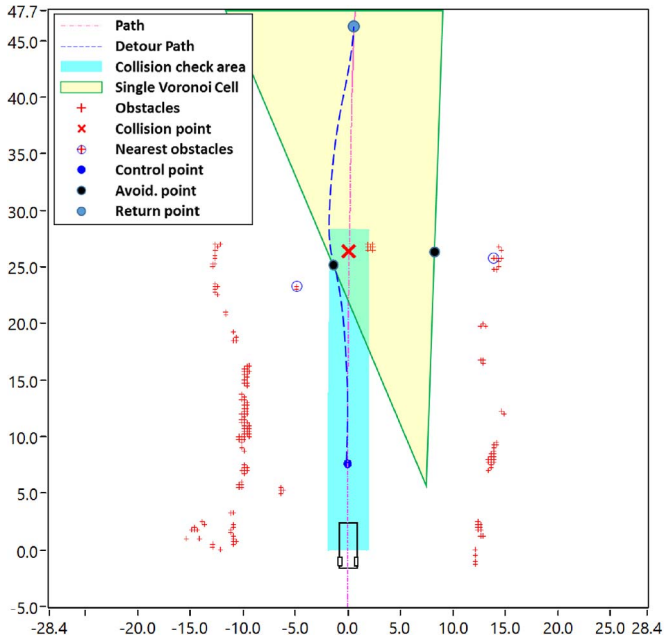


Fig. 9. Path generation using a single Voronoi cell's edge algorithm.

In Unified Map, the pedestrian detector functions similar with the traffic-light and the barrier gate detectors. When the vehicle to pass is detected, imaginary obstacles are generated as the guide to overtake the detected vehicle. The passing path is always generated on the left of the detected vehicle using the road lane information, which is described in the next section.

H. Road Lane Detection Module

We use the method in [22] to extract a feature map of road markers. The feature map is denoised by dilation, erosion, and blob labeling. If the width of each blob is greater than width threshold $\theta_{\mathbf{W}}$, the blob is not regarded as a road lane. The value of $\theta_{\mathbf{W}}$ is proportionally set to the y-coordinate in an image.

The road lane detection module provides the information about the reference points used for generating imaginary obstacles when the situation of passing another vehicle occurs.

In our implementation, the minimum and maximum values of $\theta_{\mathbf{W}}$ were set to 10 and 100 pixels, respectively.

V. Path Planning

Conventional navigation methods using waypoints have sharp corners at each waypoint, and this causes a jerky steering movement. In addition, when a vehicle is located between waypoints, a conventional method cannot clearly represent a pose of the vehicle. This uncertainty of the pose has made a problem of detecting collisions. For eliminating the jerky steering and the uncertainty, the waypoint-based path is converted into a continuous path using cubic polynomial fitting. This cubic-polynomial-based path has better performance for steering movement and collision detection than do using waypoints directly, and it can be easily modified [27]–[29].

As shown in Fig. 9, our local path planner algorithm uses a single Voronoi cell's edge algorithm for a continuous path [30]. This algorithm works well in real time and can achieve a lower

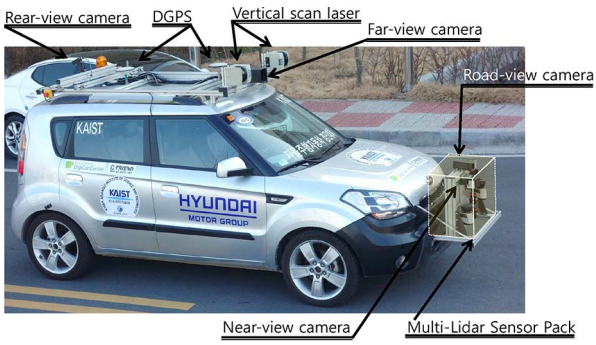


Fig. 10. KAIST autonomous vehicle, i.e., EURECAR.

computational complexity by only considering a single Voronoi cell and not all Voronoi cells, as indicated in a conventional Voronoi diagram [31].

The path planner consists of a path tracker and a path modification algorithm. The path tracker follows the path by calculating the feedback errors from a control point on the path, a current position, and the estimated pose from the Navigation System.

The path modification algorithm is triggered when the path intersects with obstacles in the collision check area. When obstacles are detected in this area, collision points are determined by constructing a perpendicular line to the path through the positions of each detected obstacle. Each intersection point between the perpendicular line and the path means a collision point. Among the collision points, the closest collision point is only used for a detour path. For determining the detour path, three kinds of new points are generated, i.e., a control point, avoidance points, and a return point. The control point is an imaginary waypoint that is the reaching point for the vehicle, and it is always on the path. The avoidance points are candidate anchor points for generating a collision-free path. The return point is also an imaginary waypoint on the path. The distance between the return point and the closest collision point is equal to the distance between the control point and the closest collision point. The single Voronoi cell's edge algorithm is used for calculating the avoidance points. This algorithm determines a detour path by only evaluating a few avoidance points. Each candidate avoidance point is determined at the midpoint between the collision point and the nearest obstacles, except for the nearest obstacles (located less than a certain distance from the collision point).

VI. Experimental Results

Fig. 10 shows our autonomous vehicle system, i.e., EURECAR. To validate the performance of the proposed autonomous vehicle system with built-in modules, we analyzed the results of the competition we participated in. For further evaluation of the proposed modules, we used additional data collected while driving EURECAR through our campus. A movie clip and short descriptions for the competition can be found in the supplementary material.[1]

[1]http://rcv.kaist.ac.kr/ iwshim/ITS2014/Supplementary.zip

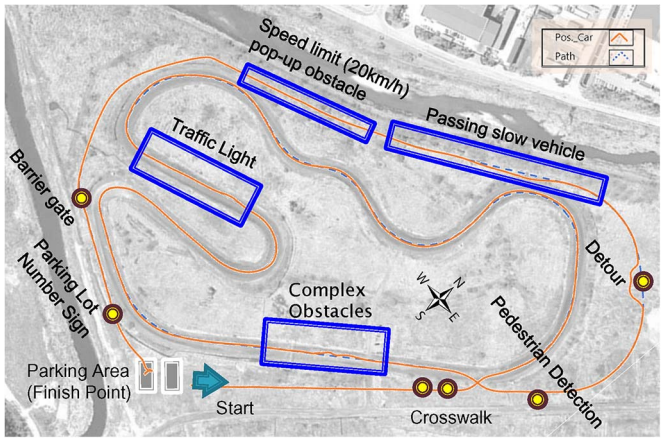


Fig. 11. Full course of the AVC at the Hyundai–Kia R&D Center. (Orange line) Actual driving path of EURECAR in the competition. (Blue dashed line) Reference waypoints. (Yellow dots and blue boxes) Locations of the mission events taking place. For the best view, see the digital color version.

A. Analysis of Competition Results

Our research group participated in the AVC hosted by Hyundai Motor Group. The AVC took place on October 7, 2012 at the Hyundai–Kia Research and Development (R&D) Center (Hwaseong, Korea). In this section, we describe the details of the missions of the competition and the vehicle behaviors during the competition. In the latter part, we discuss and analyze our data-log records. The full course of the competition is shown in Fig. 11.

The competition consisted of nine missions as follows: 1) stopping the vehicle within 2 m from a crosswalk marker when a traffic light is red; 2) switching driving paths according to the direction of an arrow traffic light; 3) passing complex obstacle regions without any collision; 4) stopping a vehicle within 5 m from a position where a person is standing on a roadside; 5) detouring from a road section obstructed by a large road block; 6) overtaking a vehicle in front; 7) stopping when an obstacle suddenly appeared in the road (a pop-up obstacle); 8) stopping when a barrier gate blocks the road; and 9) reverse parking in a parking slot designated by a roadside sign. The detailed rules of the competition can be found in the supplementary material.

As shown in Fig. 11, the track for the competition consisted of 1.4 km of off-road (the inner course) and 1.9 km of paved sections (the outer course). The orange line is the actual driving path of EURECAR, and the blue dashed line is given at reference waypoints. The yellow dots and the blue boxes show where the mission events took place. Since we did not know the exact locations of the mission events before we started, prior map information could not be used. EURECAR always had to search all the surrounding regions that it could scan and run its algorithms parallel for real-time results.

The top row in Fig. 12 shows the profiles of the built-in vision detectors. This graph shows the number of true signals for each vision detection module. To significantly reduce FPs and to enhance precision, we adopted a voting scheme with ten times per second as a threshold. When positive signals are detected over ten times per second from a detector, our system recognizes the signal as true.

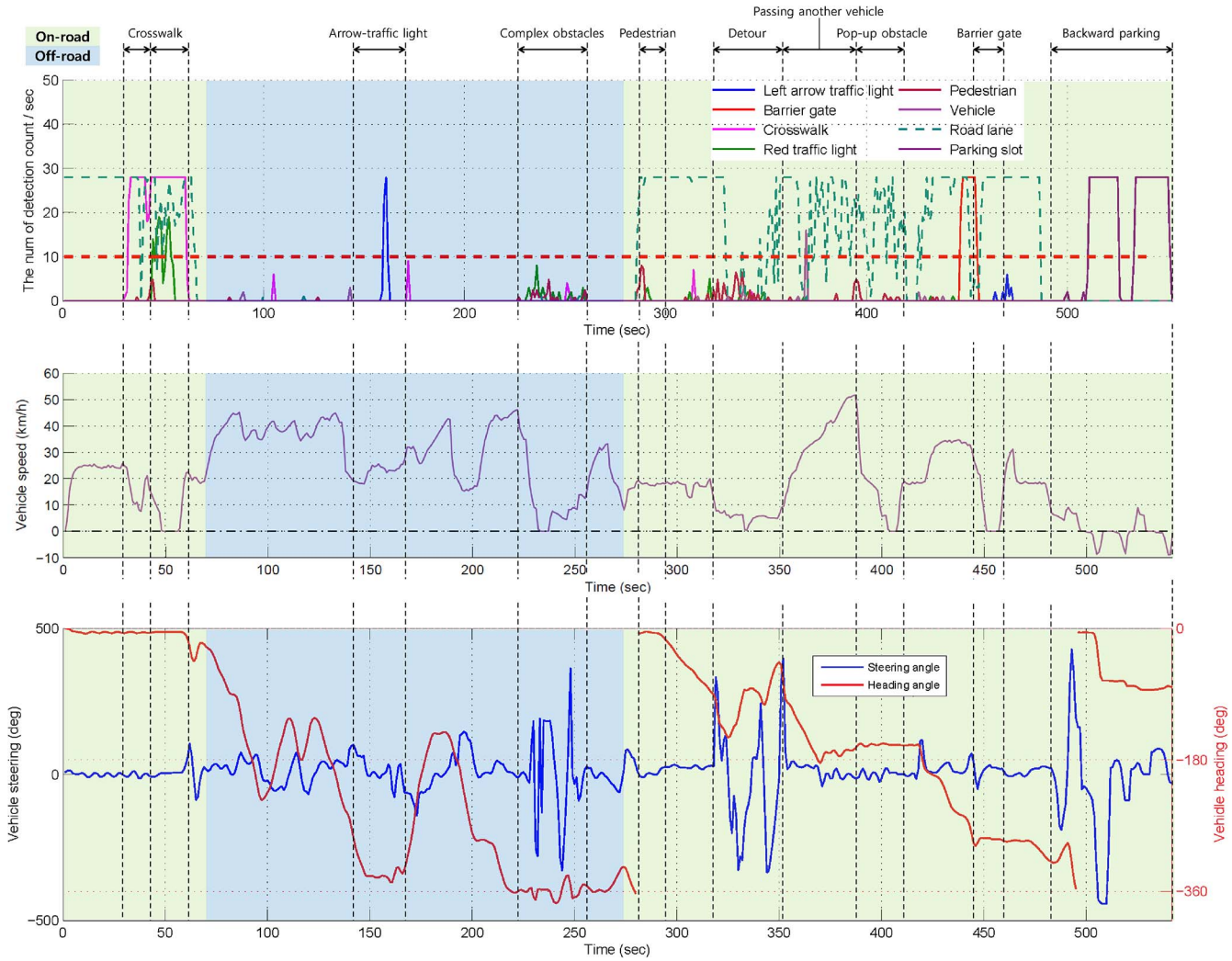


Fig. 12. Profiles of EURECAR behaviors in the AVC. (Top) Detection signal counts of EURECAR's built-in detectors. (Middle) Velocity transition. (Bottom) Steering angle (blue) and heading angle (red) transitions.

In the following sections, we discuss and analyze the behaviors of EURECAR from the log in Fig. 12.

Analysis by Detections: To understand what happened during the competition, the behavior of the vehicle is reported in Fig. 12. Among nine missions, we completed eight and failed one, which was the pedestrian detection. The first mission was to stop the vehicle in front of a crosswalk when the traffic light turned red. The crosswalk and traffic-light mission was composed of two sets of crosswalks and traffic lights. On our turn, the first traffic light was green, and the second turned red. EURECAR passed the first crosswalk and stopped in front of the second crosswalk successfully. Our vehicle then entered the off-road course, which included two missions. One mission was the recognition of an arrow traffic light and the direction of the vehicle to the designated side of the traffic light. In the arrow traffic-light mission, the left arrow was on. Our system correctly detected the arrow direction and commanded the vehicle to steer left. The next mission was to pass complex obstacles composed of many traffic cones and barrels without any bumps. In the complex obstacles section, since the vehicle made a sudden stop, which raised a cloud of dust, the dust was recognized as an obstacle by the LiDAR sensors. That is the reason why the vehicle stopped for a few seconds at $t = 230$ s. After the dust cleared, EURECAR resumed its mission at the complex obstacle section and clearly passed without any bumps. In the pedestrian detection mission, a mannequin stood on the roadside, and the car had to detect it and stop right next to it. Unfortunately, EURECAR did not make a stop. In the top row in Fig. 12, our pedestrian detector detected the pedestrian 20 times ($t = 286$–289 s), but the maximum count per second of the positive signals of the detector was just 8, which is lower than the threshold of 10. (This will be discussed later.) In the detour section, the vehicle successfully sensed the road block and searched for a detour path. Since no waypoint of the detour path was given, the velocity of the vehicle was enforced to be low while it computed a safe path through the detour section marked by two sets of traffic cones on each side of the vehicle. The cones were detected by the onboard LiDAR sensors. Next, EURECAR passed a slowly moving vehicle on the road by crossing the center line to the left. After passing, the vehicle returned to the right side of the road. It stopped at a pop-up obstacle (a large box with a string pulled by a person). The vehicle waited in front of the barrier gate when it was lowered to block the road and then passed when it was open.

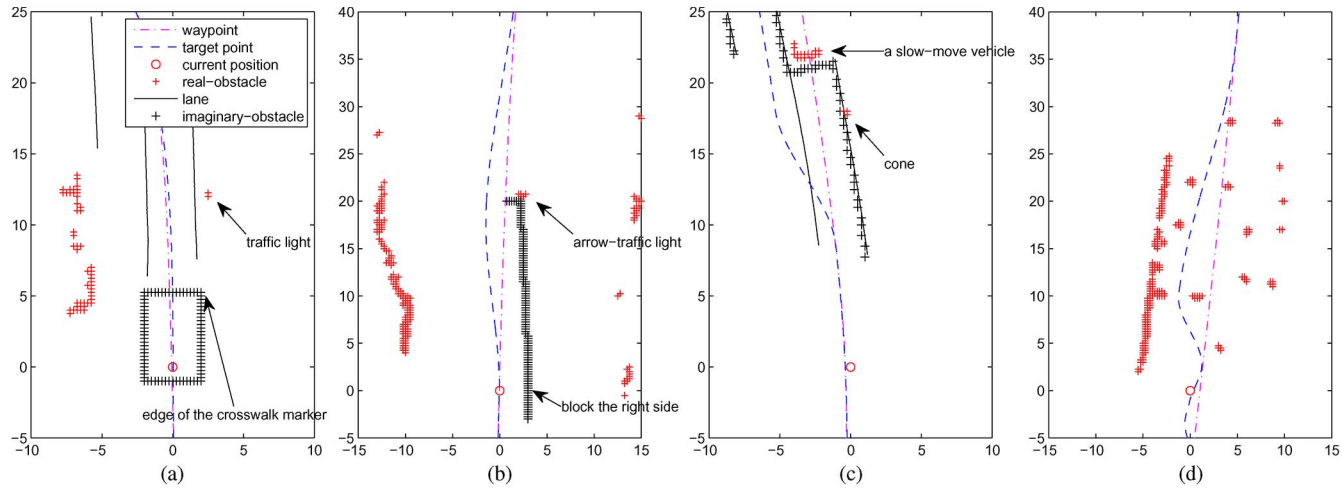


Fig. 13. Key moments during the AVC. (a) Second crosswalk. (b) Arrow traffic light. (c) Passing another vehicle. (d) Complex obstacles.

After $t = 490$, EURECAR started the last mission to park on a slot designated by a sign plate indicating the number 1 or 2. EURECAR succeeded in the reverse parking mission.

The counts of the road lane detection fluctuated for $t =$ 330–460. During this time interval, the road surface was occluded by some objects for missions, such as traffic cones.

Movement Analysis: From the driving path in Fig. 11 and the log in Fig. 12, we were able to analyze the vehicle behavior during the competition. The middle row in Fig. 12 presents the vehicle speed, and the bottom row presents the measured steering angle (from the encoder) and the heading angle (from the Navigation System) [15]. The Navigation System generated the heading angles by the IMU integrated with the DGPS using the EKF [32]. The angles are expressed in degrees. Looking at the log of the steering angle (the blue line) in Fig. 12, it could be seen that the steering angle largely fluctuated in the complex obstacle and detour sections, whereas other sections show plausible transitions of the angle. In these two sections, since obstacles were very close together, the scan lines of the LiDAR were occluded by the traffic cones and barrels in front of the vehicle. The vehicle could not detect all the obstacles at one scanning. When the occluded obstacles were made visible by the movement of the vehicle, new obstacles were detected. Due to the newly detected obstacles, the path was frequently updated; thus, the steering angles also frequently changed to follow the updated path. Although the steering angle transitions in the two sections seem like large movements, the vehicle velocity was low, and the movement was indeed smooth, considering the actual driving path shown in Fig. 11 and the heading angle (the red line) shown in Fig. 12.

Key Moment Analyses: In Fig. 13, key moments in the competition are reconstructed from the log data. When the red traffic light and the crosswalk marker were detected at the same time, the corresponding imaginary obstacles appeared, as shown in Fig. 13(a). The vehicle was forced to stop by the imaginary obstacles surrounding it. The front part of the blocks was located at the estimated crosswalk marker position. The blocks were removed when the traffic light changed to green. The arrow traffic-light case is depicted in Fig. 13(b), where the imaginary obstacles blocked the region from the right side of the vehicle to the left side of the traffic light. For the case of detecting another vehicle, Fig. 13(c) shows how the imaginary obstacles were generated. The imaginary obstacles were added due to the position of the detected vehicle and the width of one lane on the road. The road lane information was only used for passing another vehicle. An imaginary obstacle was not generated in the complex obstacle section in Fig. 13(d).

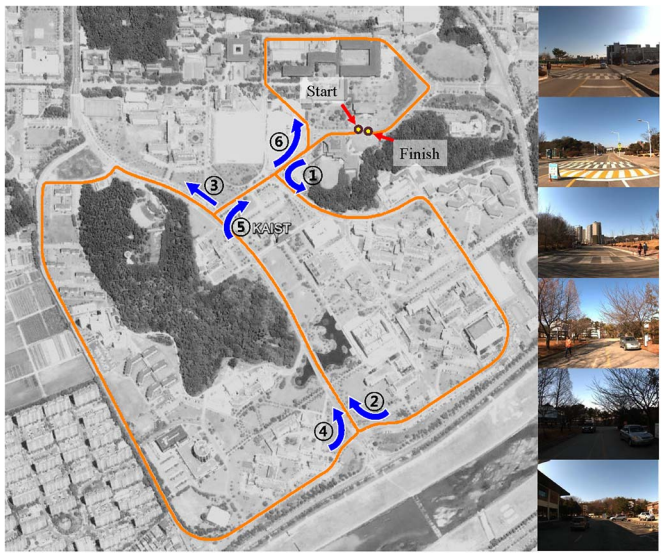


Fig. 14. KAIST campus environment used for collecting additional data.

B. Evaluation of Built-In Modules

For further validation, an additional data set was collected while driving EURECAR in our KAIST campus. Fig. 14 depicts the trajectory of our campus course and its thumbnail images for the collected data. The total length of the course was about 5.7 km, and we drove the same path three times. The Institut National de Recherche en Informatique et en Automatique (INRIA) [33] and Karlsruhe Institute of Technology–Toyota Technological Institute at Chicago (KITTI) [34] data sets were used to train the pedestrian and vehicle detectors, respectively.

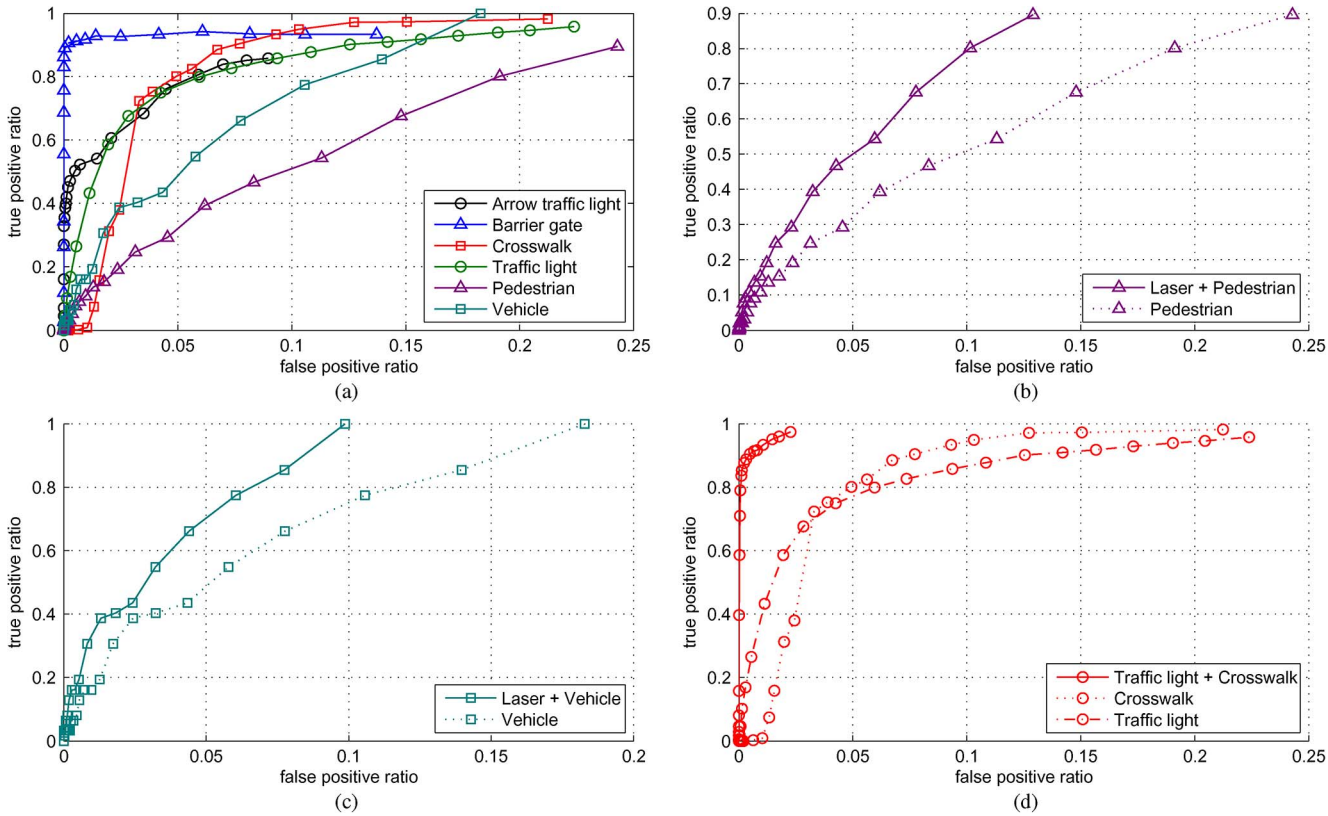


Fig. 15. Quantitative evaluation result of the proposed built-in modules. (a) Quantitative results of the individual detector without combining. (b) Comparison of two pedestrian detectors using the full search space and the reduced search space. (c) Comparison of two vehicle detectors using the full search space and the reduced search space. (d) Comparison between two algorithms and the combined algorithms.

Fig. 15(a)–(d) shows the quantitative evaluation results of the detection experiments. Here, we provide a true positive ratio against the FP ratio. For this evaluation metric, better performance is indicated by the bending of the curve toward the top-left side of the graph. In Fig. 15(a), the original performance of the proposed vision detectors is shown. Since artifacts on the road such as lane marks and traffic lights are designed with noticeable colors, the color-feature-based detectors show good performance.

Fig. 15(b) and (c) shows the performance of the pedestrian and vehicle detectors with reduced candidate search windows, as described in Section IV-G. Since the obstacle detector rejects FPs in advance, the FP errors are reduced by integrating the pedestrian and vehicle detectors with the proposed Unified Map. Fig. 15(d) shows the performance regarding traffic lights and crosswalks, as well as that after combining these two algorithms. In the competition, since the traffic lights and the crosswalks were always located at the same place, we could combine these two algorithms to improve the performance of the detectors. The dotted lines denote the original performance, and the solid lines denote the performance by combining the algorithms.

In the following section, we discuss some issues for further improvements of the proposed algorithms.

Pedestrian Detection: Among the problems related to the competition, the pedestrian detection was the most challenging problem. To make it practical, we adopted the voting scheme and the LiDAR fusion, and we have shown that it successfully handled FP signals and the other missions. However, during the competition, the special treatment produced missing true positives. We analyzed the fundamental reasons and discuss about future directions here.

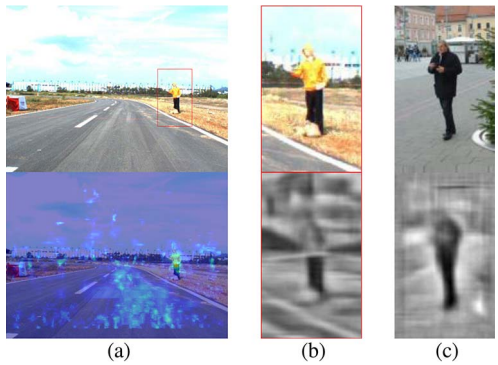


Fig. 16. Images for the pedestrian detection mission. (a) Original image captured by the (top) front-view camera and (bottom) HOG response map of the image. (b) Pedestrian image cropped from the (top) original and (bottom) its visualizations of feature spaces using HOGgles [36]. (c) Pedestrian image from the INRIA data set [35] for comparison with our captured image.

As shown in an example in Fig. 16(a), since the response of the histogram of gradients (HOG) detector was lower than we expected, the detector could not generate enough positive signals. One reason for this was the poor quality of the captured images due to the camera exposure setting. We manually set the gain level and shutter speed of the cameras to fit the road surface. At that time, since the illumination was unstable and clouds were also changing, some regions of the images were

saturated. As shown in the top row in Fig. 16(b) and (c), to better visualize the quality of our captured pedestrian image, we compare our image with a well-captured pedestrian image from the INRIA data set [35]. In addition, we use a HOGgles [36] for visualizing from the perspective of the HOG feature space.[2] By only seeing the bottom in Fig. 16(b), it is hard even for a human to be convinced, whereas Fig. 16(c) is more plausible.

Almost all the vision-based algorithms may be vulnerable to image quality. Robust and adaptive image quality adjustments such as in [37] and [38] can help stabilize the vision-based algorithms so that the overall intelligent vehicle system works robustly. Developing adaptation algorithms' built-in cameras is one of the crucial directions for making intelligent systems practical.

Global Coordinate Map: Throughout our research, we relied on DGPS-based localization, which provides up to 2-cm accuracy. If DGPS is available and maintains its accuracy and integrity along the entire track during the competition, it is unquestionably the best localization method with a negligible computation load. However, DGPS is not a viable option for real-world applications due to its high cost, limited availability, and poor reliability. Therefore, a number of alternative approaches such as simultaneous localization and mapping, and structure from motion have been studied [39]–[41]. Using such approaches, our local map and the output of each detection module can be fused spatiotemporally. However, these approaches require high computing loads with a larger error that can be accumulated over time [42]. Therefore, since DGPS was allowed for this competition event, we chose it as the primary position measurement for the sake of smaller measurement error and computing load.

Reliability Issues: Generally, the reliability of autonomous vehicle systems depends on the maximum speed tolerated and the sensor configuration. The sensing performance and the control limitation should be balanced. The speed limitation can be determined by the detection range and computation times of the detection algorithms.

In our system, many 2-D LiDAR sensors and cameras were installed for building the local map (Unified Map). The number of sensors can be reduced by replacing several sensors with a single high-end sensor (e.g., for the cameras, replacing them with a high-resolution, high-frame-rate, and larger FoV or omnidirectional camera; for the LiDAR sensors, replacing them with a 3-D LiDAR sensor such as *Velodyne*). In that case, the lowered reliability of detection due to blind spots may be reduced. By complements of multiple sensors, we can reduce blind spots and improve the reliability of the system.

When using multiple sensors, the sensor independence should be also guaranteed for safety. Once a sensor malfunctions, the vehicle may come to a halt, as demonstrated by one competitor vehicle that crashed into another vehicle when one sensor was damaged by a minor crash in the preceding mission. To develop a reliable autonomous vehicle, the sensor system should be designed to be independent so that other sensors continue to function regardless of any sensor malfunction.

[2]By HOGgles visualizations, we can see why the pedestrian detection did not make enough responses to detect.

Adverse Environmental Condition: It should be considered that all autonomous vehicle systems can fail due to some common deficiency of the sensors. For example, the LiDAR network can completely fail on a rainy or snowy day as it suffers from interference from rain drops or snowflakes making multiple echoes. Therefore, an improved sensor system would consist of complementary sensors such as LiDAR and radar, combined with robust algorithms that could correctly function under conditions as various as rainy days, nights, total loss of GPS, and severely varying light conditions.

VII. Conclusion

In this paper, we have presented an autonomous vehicle system called EURECAR. Our system is based on Unified Map that represents the outputs from many detection algorithms as imaginary obstacles in addition to real obstacles. Owing to the simple representation of traffic environments in Unified Map, the path planner can easily generate a safe path without a complex algorithm to handle real obstacles and traffic constraints separately.

In addition, we have reviewed the overall system including the hardware configuration and the sensor network, including the modular architecture that collects the information from individual detectors. Although the performance of our system was limited by the adopted detectors, it can be easily extended by adding detectors suitable to specific environments due to the modular architecture and the sensor network.

The proposed methods have been successfully implemented and tested in a real vehicle. Our autonomous vehicle was evaluated on challenging real tracks with a set of traffic signals and showed good performance. In particular, at the AVC, the vehicle performed almost perfectly and won an award. If the system is made more robustly with multiple complementary sensors and more solid algorithms, it will function more reliably under a wider range of operating conditions.

References

[1] S. Thrun *et al.*, "Stanley: The robot that won the DARPA grand challenge," *J. Field Robot. (JFR)*, vol. 23, no. 9, pp. 661–692, Sep. 2006.
[2] U. Ozguner, C. Stiller, and K. Redmill, "Systems for safety and autonomous behavior in cars: The DARPA grand challenge experience," *Proc. IEEE*, vol. 95, no. 2, pp. 397–412, Feb. 2007.
[3] M. Buehler, K. Iagnemma, and S. Singh, *The 2005 DARPA Grand Challenge: The Great Robot Race*. Berlin, Germany: Springer-Verlag, 2007, vol. 36.
[4] M. Montemerlo *et al.*, "Junior: The Stanford entry in the urban challenge," *J. Field Robot. (JFR)*, vol. 25, no. 9, pp. 569–597, Sep. 2008.
[5] J. Leonard *et al.*, "A perception-driven autonomous urban vehicle," *J. Field Robot. (JFR)*, vol. 25, no. 10, pp. 727–774, Oct. 2008.
[6] B. J. Patz, Y. Papelis, R. Pillat, G. Stein, and D. Harper, "A practical approach to robotic design for the DARPA urban challenge," *J. Field Robot. (JFR)*, vol. 25, no. 8, pp. 528–566, Aug. 2008.
[7] C. Urmson *et al.*, "Autonomous driving in urban environments: Boss and the urban challenge," *J. Field Robot. (JFR)*, vol. 25, no. 8, pp. 425–466, Aug. 2008.
[8] J. Markoff, *Google Cars Drive Themselves, in Traffic*. New York, NY, USA: The New York Times, 2010, vol. 10, p. A1.
[9] M. Bertozzi, A. Broggi, A. Coati, and R. I. Fedriga, "A 13,000 km intercontinental trip with driverless vehicles: The VIAC experiment," *IEEE Intell. Transp. Syst. Mag.*, vol. 5, no. 1, pp. 28–41, Spring 2013.
[10] A. Broggi, L. Bombini, S. Cattani, P. Cerri, and R. I. Fedriga, "Sensing requirements for a 13,000 km intercontinental autonomous drive," in *Proc. IEEE IV*, 2010, pp. 500–505.

[11] W. Burgard and M. Hebert, *World Modeling*. Berlin, Germany: Springer-Verlag, 2008.
[12] P. Pfaff, R. Triebel, and W. Burgard, "An efficient extension to elevation maps for outdoor terrain mapping and loop closing," *Int. J. Robot. Res.*, vol. 26, no. 2, pp. 217–230, Feb. 2007.
[13] T. Stoyanov, M. Magnusson, H. Andreasson, and A. J. Lilienthal, "Path planning in 3-D environments using the normal distributions transform," in *Proc. IEEE/RSJ Int. Conf. IROS*, 2010, pp. 3263–3268.
[14] Y. Choe, I. Shim, and M. J. Chung, "Urban structure classification using the 3-D normal distribution transform for practical robot applications," *Adv. Robot.*, vol. 27, no. 5, pp. 351–371, Apr. 2013.
[15] S. Huh and D. H. Shim, "A vision-based landing system for small unmanned aerial vehicles using an airbag," *Control Eng. Pract.*, vol. 18, no. 7, pp. 812–823, Jul. 2010.
[16] G. C. Buttazzo, *Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications*. New York, NY, USA: Springer-Verlag, 2011, vol. 24.
[17] D. P. Bovet and M. Cesati, *Understanding the Linux kernel*. Sebastopol, CA, USA: O'Reilly Media, Inc., 2005.
[18] B. Chapman, G. Jost, and R. Van Der Pas, *Using OpenMP: Portable Shared Memory Parallel Programming*. Cambridge, MA, USA: MIT Press, 2008, vol. 10.
[19] Q. Zhang and R. Pless, "Extrinsic calibration of a camera and laser range finder (improves camera calibration)," in *Proc. IEEE/RSJ Int. Conf. IROS*, 2004, vol. 3, pp. 2301–2306.
[20] D. Michie, D. J. Spiegelhalter, and C. C. Taylor, *Machine Learning, Neural and Statistical Classification*. New York, NY, USA: Ellis Horwood, 1994.
[21] J. Levinson, J. Askeland, J. Dolson, and S. Thrun, "Traffic light mapping, localization, and state detection for autonomous vehicles," in *Proc. IEEE ICRA*, 2011, pp. 5784–5791.
[22] P. Foucher, Y. Sebsadji, J.-P. Tarel, P. Charbonnier, and P. Nicolle, "Detection and recognition of urban road markings using images," in *Proc. IEEE Int. ITSC*, 2011, pp. 1747–1752.
[23] J. J. Moré, "The Levenberg–Marquardt algorithm: Implementation and theory," in *Numerical analysis*. Berlin, Germany: Springer-Verlag, 1978, pp. 105–116.
[24] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 11, pp. 1330–1334, Nov. 2000.
[25] M. Pedersoli, A. Vedaldi, and J. Gonzalez, "A coarse-to-fine approach for fast deformable object detection," in *Proc. IEEE Conf. CVPR*, 2011, pp. 1353–1360.
[26] R. Benenson, M. Mathias, R. Timofte, and L. Van Gool, "Pedestrian detection at 100 frames per second," in *Proc. IEEE Conf. CVPR*, 2012, pp. 2903–2910.
[27] T. D. Barfoot and C. M. Clark, "Motion planning for formations of mobile robots," *Robot. Autonom. Syst.*, vol. 46, no. 2, pp. 65–78, Feb. 2004.
[28] S. Thompson and S. Kagami, "Continuous curvature trajectory generation with obstacle avoidance for car-like robots," in *Proc. Int. Conf. Comput. Intell. Modell., Control Autom. Int. Intell. Agents, Web Technol. Internet Commerce*, vol. 1, 2005, pp. 863–870.
[29] B. Nagy and A. Kelly, "Trajectory generation for car-like robots using cubic curvature polynomials," in *Proc. Field Serv. Robots*, 2001, vol. 11, pp. 1–6.
[30] U. Lee, S. Yoon, D. H. Shim, P. Vasseur, and C. Demonceaux, "Local path planning in a complex environment for self-driving car," in *Proc. IEEE Int. Conf. Cyber Technol. Autom., Control Intell. Syst.*, 2014, pp. 445–450.
[31] H. Choset and J. Burdick, "Sensor-based exploration: The hierarchical generalized Voronoi graph," *Int. J. Robot. Res.*, vol. 19, no. 2, pp. 96–125, 2000.
[32] E.-H. Shin and N. El-Sheimy, "Accuracy improvement of low cost INS/GPS for land applications," M.S. thesis, Dept. Geomatics Eng., Univ. Calgary, Calgary, AB, Canada, 2001.
[33] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Conf. CVPR*, vol. 1, 2005, pp. 886–893.
[34] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. IEEE Conf. CVPR*, 2012, pp. 3354–3361.
[35] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Int. Conf. Comput. Vis. Pattern Recogn.*, C. Schmid, S. Soatto, and C. Tomasi, Jun. 2005, vol. 2, pp. 886–893. [Online]. Available: http://lear.inrialpes.fr/pubs/2005/DT05
[36] C. Vondrick, A. Khosla, T. Malisiewicz, and A. Torralba, "HOGgles: Visualizing Object Detection Features," *Proc. IEEE ICCV*, 2013, pp. 1– 8.
[37] I. Shim, J.-Y. Lee, and I. S. Kweon, "Auto-adjusting camera exposure for outdoor robotics using gradient information," in *Proc. IEEE/RSJ Int. Conf. IROS*, 2014, pp. 1011– 1017.
[38] S. Hrabar, P. Corke, and M. Bosse, "High dynamic range stereo vision for outdoor mobile robotics," in *Proc. IEEE ICRA*, 2009, pp. 430–435.
[39] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: Part I," *IEEE Robot. Autom. Mag.*, vol. 13, no. 2, pp. 99–110, Jun. 2006.
[40] Y. Jeong, D. Nister, D. Steedly, R. Szeliski, and I.-S. Kweon, "Pushing the envelope of modern methods for bundle adjustment," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 8, pp. 1605–1617, Aug. 2012.
[41] N. Snavely, S. M. Seitz, and R. Szeliski, "Photo tourism: exploring photo collections in 3D," *ACM Trans. Graphics*, vol. 25, no. 3, pp. 835–846, Jul. 2006.
[42] Y. Bok, Y. Jeong, D.-G. Choi, and I. S. Kweon, "Capturing village-level heritages with a hand-held camera-laser fusion sensor," *Int. J. Comput. Vis.*, vol. 94, no. 1, pp. 36–53, Aug. 2011.

Inwook Shim (S'12) received the B.S. degree in computer science from Hanyang University, Seoul, Korea, in 2009 and the M.S. degree in robotics program from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 2011. He is currently working toward the Ph.D. degree in the Division of Future Vehicle, Robotics and Computer Laboratory, Department of Electrical Engineering, College of Information Science and Technology, KAIST.

His research interests include real-time map representation and software architecture for autonomous systems.

Mr. Shim was the recipient of the Qualcomm Innovation Award in 2013.

Jongwon Choi (S'14) received the B.S. and M.S. degrees in electronic and electrical engineering from Korea Advanced Institute of Science and Technology, Daejeon, Korea, in 2012 and 2014, respectively. He is currently working toward the Ph.D. degree in electrical and computer engineering at Seoul National University, Seoul, Korea.

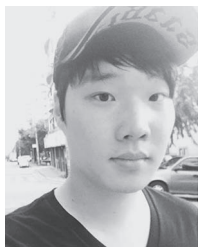
His research interests include optical flow and traffic environments/objects.

Mr. Choi was the recipient of the Qualcomm Innovation Award in 2013.

Seunghak Shin (S'14) received the B.S. degree in electrical engineering from Yonsei University, Seoul, Korea, in 2009 and the M.S. degree in robotics program from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 2011. He is currently working toward the Ph.D. degree in electrical engineering in the Robotics and Computer Vision Laboratory, Department of Electrical Engineering, College of Information Science and Technology, KAIST.

His research interests include the simultaneous localization and mapping problem and scene matching.

Mr. Shin was the recipient of the Qualcomm Innovation Award in 2013.



Tae-Hyun Oh (S'13) received the B.E. degree (*summa cum laude*) in computer engineering from Kwangwoon University, Seoul, Korea, in 2010 and the M.S. degree in electrical engineering from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 2012. He is currently working toward the Ph.D. degree in the Robotics and Computer Vision Laboratory, Department of Electrical Engineering, College of Information Science and Technology, KAIST.

In 2014, he was a Visiting Student with the Visual Computing Group, Microsoft Research Asia. His research interests include robust computer vision and machine learning.

Mr. Oh was a recipient of the Qualcomm Innovation Award.

Unghui Lee received the B.S. degree in aerospace engineering from Korea Aerospace University, Goyang, Korea, in 2009 and the M.S. degree in aerospace engineering from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 2011. He is currently working toward the Ph.D. degree in aerospace engineering in the Flight Dynamics and Control Laboratory, Department of Aerospace Engineering, School of Mechanical Aerospace and Systems Engineering, College of Engineering, KAIST.

His research interests include path planning and navigation for autonomous systems.

Byungtae Ahn (S'13) received the B.S. degree in electronic engineering from Kumoh National Institute of Technology, Gumi, Korea, in 2007 and the M.S. degree in biomechatronics from Sungkyunkwan University, Seoul, Korea, in 2011. He is currently working toward the Ph.D. degree in robotics program in the Robotics and Computer Vision Laboratory, Department of Electrical Engineering, College of Information Science and Technology, KAIST.

His research interests include advanced driver assistance systems and gaze estimation.

Mr. Ahn was the recipient of the Qualcomm Innovation Award in 2013.

Dong-Geol Choi (S'12) received the B.S. and M.S. degrees in electric engineering and computer science from Hanyang University, Seoul, Korea, in 2005 and 2007, respectively. He is currently working toward the Ph.D. degree in robotics program in the Robotics and Computer Vision Laboratory, Department of Electrical Engineering, College of Information Science and Technology, Korea Advanced Institute of Science and Technology, Daejeon, Korea.

His research interests include mobile navigation and sensor calibration issues.

Mr. Choi was the recipient of the Qualcomm Innovation Award in 2013.

David Hyunchul Shim received the B.S. and M.S. degrees in mechanical design and production engineering from Seoul National University, Seoul, Korea, in 1991 and 1993, respectively, and the Ph.D. degree in mechanical engineering from the University of California Berkeley, Berkeley, CA, USA, in 2000.

From 1993 to 1994, he was with Hyundai Motor Company, Young-in, Korea, as a Transmission Design Engineer. From 2001 to 2005, he was with Maxtor Corporation, Milpitas, CA, USA, as a Staff Engineer working on advanced control systems for hard disk drives. From 2005 to 2007, he was with the University of California Berkeley as a Principal Engineer in charge of the Berkeley Aerobot Team. In 2007, he joined the Department of Aerospace Engineering, School of Mechanical Aerospace and Systems Engineering, College of Engineering, Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, as an Assistant Professor. He is currently an Associate Professor with and the Director of the Center of Field Robotics for Innovation, Exploration, and Defense, KAIST. His research interests include dynamics and control, unmanned vehicle systems, and field robotics.

In-So Kweon (M'94) received the B.S. and M.S. degrees in mechanical design and production engineering from Seoul National University, Seoul, Korea, in 1981 and 1983, respectively, and the Ph.D. degree in robotics from Carnegie Mellon University, Pittsburgh, PA, USA, in 1990.

He worked with Toshiba Research and Development Center, Kawasaki, Japan, and in 1992, he joined the Department of Automation and Design Engineering, Korea Advanced Institute of Science and Technology, Daejeon, Korea, where he is now a Professor with the Department of Electrical Engineering, College of Information Science and Technology. His research interests include camera and 3-D sensor fusion, color modeling and analysis, visual tracking, and visual simultaneous localization and mapping.

Dr. Kweon is a member of the Korea Robotics Society (KROS). He was the Program Cochair for the 2007 Asian Conference on Computer Vision (ACCV) and was the General Chair for the 2012 ACCV. He is on the editorial board of the *International Journal of Computer Vision*. He was the recipient of the Best Student Paper Runner-Up Award at the 2009 IEEE Conference on Computer Vision and Pattern Recognition.