A Review of Intelligent Self-Driving Vehicle Software Research

Jeongwhan Gwak[1], Juho Jung[1], RyumDuck Oh[1], Manbok Park[2], Mukhammad Abdu Kayumbek Rakhimov[1] and Junho Ahn[1]*

[1] Computer Information Technology, Korea National University of Transportation, Chungju, South Korea
[e-mail: jgwak@ut.ac.kr (Jeonghwan Gwak), jjs1005k@ut.ac.kr (Juho Jung), rdoh@ut.ac.kr (RyumDuck Oh), rakhimov@ut.ac.kr (Mukhammad Abdu Kayumbek Rakhimov), jhahn@ut.ac.kr (Junho Ahn)]

[2] Department of Electronic Engineering, Korea National University of Transportation, Chungju, South Korea
[e-mail: ohnnuri@ut.ac.kr (Manbok Park)]

*Corresponding author: Junho Ahn

Received June 21, 2019; revised August 23, 2019; accepted October 6, 2019; published November 30, 2019

Abstract

Interest in self-driving vehicle research has been rapidly increasing, and related research has been continuously conducted. In such a fast-paced self-driving vehicle research area, the development of advanced technology for better convenience safety, and efficiency in road and transportation systems is expected. Here, we investigate research in self-driving vehicles and analyze the main technologies of driverless car software, including: technical aspects of autonomous vehicles, traffic infrastructure and its communications, research techniques with vision recognition, deep leaning algorithms, localization methods, existing problems, and future development directions. First, we introduce intelligent self-driving car and road infrastructure algorithms such as machine learning, image processing methods, and localizations. Second, we examine the intelligent technologies used in self-driving car projects, autonomous vehicles equipped with multiple sensors, and interactions with transport infrastructure. Finally, we highlight the future direction and challenges of self-driving vehicle transportation systems.

Keywords: Self-driving vehicle, Road infrastructure, Deep learning, Machine learning, Localization

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT : Ministry of Science and ICT) (No. 2017R1C1B5017847). This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT : Ministry of Science and ICT) (No. 2017R1C1B5017847). This research was supported by a grant(code 19PQOW-B152473-01) from R&D Program funded by Ministry of Land, Infrastructure and Transport of Korean government. And this work was supported by the Technology Innovation Program (or Industrial Strategic Technology Development Program-Automobile Industrial Core Technology Development Project) (No.K_G012000307002, Development of rear automatic braking system for NCAP) funded By the Ministry of Trade, Industry & Energy(MOTIE, Korea)

1. Introduction

According to a National Highway Traffic Safety Administration (NHTSA) report [1] in 2016, 94 percent of all automobile accidents occur due to driver errors and its associated factors. The main causes of vehicle accidents by drivers include recognition and decision errors, performance, and non-performance errors. Thus, self-driving vehicle research targets those factors, aiming at reducing the number of errors and offering solutions.

Self-driving cars continuously sense the surrounding area with vision, localization, mapping, and other techniques using camera, GPS (Global Positioning System), IMU(Inertial Measurement Unit), wheel odometry, LIDAR (Light Detection And Ranging) sensors, etc., thus reducing possible injuries and saving lives in accidents. Autonomous vehicles lead to improved convenience, safety, and efficiency of road and transportation systems [2]. Self-driving cars can collect traffic information, be connected by vehicular cloud computing [3], and find efficient travel paths in real time, further reducing air pollution caused by stopping. These vehicles can share their driving information with other systems to manage traffic congestion. Distracted, impaired, and drunk driving has been a major cause of fatalities on the road [4], and the new technology can help reduce these human errors, improving driver safety.

Here, we analyze commercial companies [5-8] such as Google, Uber, General Motors, and Tesla that are building self-driving systems. Self-driving cars in commercial companies are generally equipped with vision, LIDAR, radar, and other supplemental sensor systems to detect the surrounding area in autonomous driving. Google and Uber focus on self-driving software for precise and accurate performance, and are working with other car manufacturers to build driving system hardware. GM and Tesla are developing both the hardware and software of the systems. We examine the driverless vehicle technology of these four popular companies and compare the detailed techniques in the section below.

Here, we investigate the recent autonomous vehicle technologies of intelligence, automation, and connectivity to achieve efficient and safe driving. Innovations in intelligent technology (artificial intelligence) are accelerating the development of self-driving vehicles using computer vision (image recognition) and localization methods. In particular, camera-based vision techniques using deep learning algorithms are applied to recognize traffic information, such as lane lines, cars, signs, humans, etc., on the road with high accuracy to build intelligent traffic systems. LIDAR-based techniques [64] provide precise localization, achieving centimeter-level accuracy performance in real environments. Vehicle-to-everything (V2X) techniques allow for efficient communication between all traffic-related elements, such as vehicles, traffic infrastructure, and cloud servers, to reduce accidents and assist driving. We introduce these advanced technologies for self-driving cars, and review the recent algorithms to improve their performance.

Section 2 provides a description of the algorithms for precise computer vision, localization, and connectivity. We introduce vehicles equipped with intelligent sensors to identify the surrounding area. Section 3 describes traffic infrastructure technologies to provide information for improved safety and efficiency. In Section 4, we describe the recent commercial research in autonomous vehicles at Uber, General Motors, Tesla and Google, and compare these technologies. Section 5 summarizes and concludes the investigation, discussing future endeavors in this field.

2. Autonomous Vehicle Algorithms

2.1 Lane Tracking Algorithms for Autonomous Vehicles

Vision algorithms for developing self-driving technology have been actively studied. Early vision algorithms had low accuracy and thus had to be used in combination with other sensors. Furthermore, the existing vision algorithms relied on image processing technology. During the process, since the input image may be distorted after being received as an input, the calibration operation is performed to extract the correct result. Information on color and gradient are extracted from the calibrated image. HLS(Hue Luminance Saturation) and R information of RGB are extracted and combined to generate the color information, and x, y, the magnitude of the gradient, and direction of the gradient of Sobel are extracted and combined to generate the gradient information. Next, the region of interest (ROI) is specified to reduce noise and extract the lane area only. As the perspective is applied to an image, it is difficult to extract accurate information about the lane. Therefore, the image needs to be transformed as if it were captured from above. Using the perspective transform method [60] of OpenCV, the image is transformed accordingly, and the resulting image is usually called the bird's eye view. To extract the lane information from the transformed image, a lane is searched by using a sliding window search technique. The lane is divided into a left lane and a right lane, and a perspective transform is performed after the divided lanes are connected to be converted back to the original image. The process using image-processing technology for the lane detection is shown in **Fig. 1**.

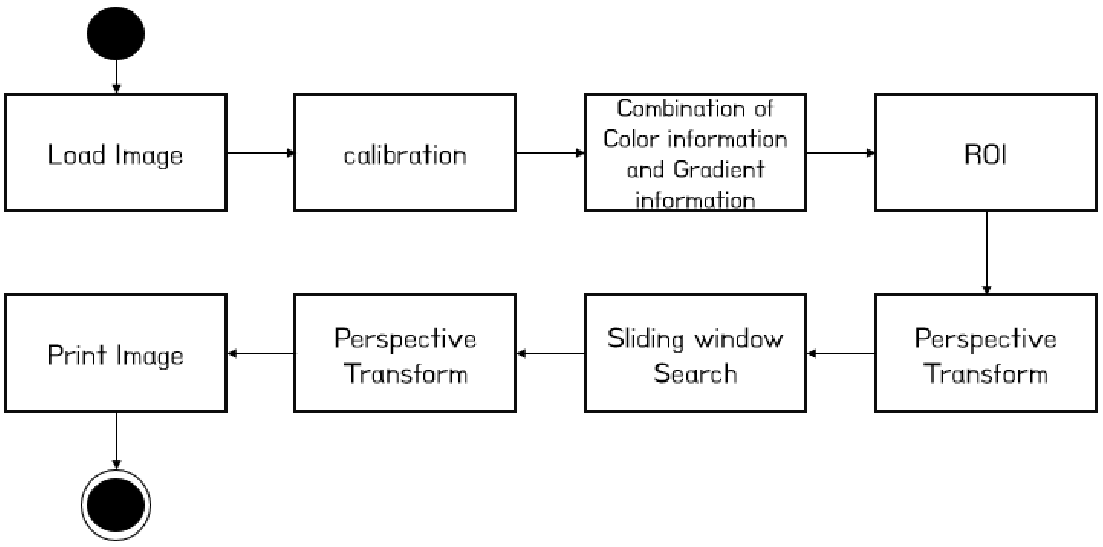

Fig. 1. Flow chart of a vision algorithm based on image processing for lane detections

Fig. 2 shows an example of a lane detection result using image-processing methods. While the lane has been detected, it is very vulnerable to surrounding noise because the system depends on image processing. Therefore, the self-driving technology is not accurate enough to detect lanes or perform object detection using vision algorithms alone. [61]

However, with the development of deep learning technology, the data necessary for self-driving can be extracted using vision technology, and the accuracy is improved correspondingly. Some studies [9–12] also used deep learning technology to detect vehicles and lanes on highways and to collect data necessary for self-driving. Computer vision plays an important role not only in lane detection, but also for detecting objects within a radius. Conventional classic computer vision algorithms were not precise or accurate enough for the human eye. However, large storage and computing capabilities over the past few years have smoothed learning-based deep-run operations.

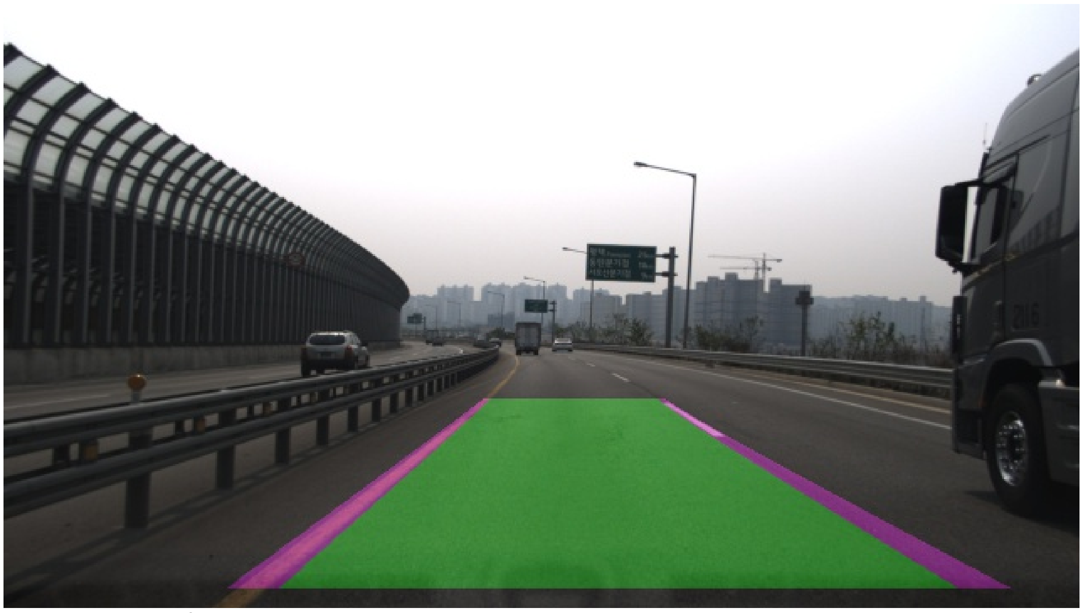

Fig. 2. Example of a lane detection result based on three image-processing methods: Sobel, perspective transform, and sliding window search.

Using a large dataset, a neural network can be trained to perform real-time detection with small-capacity models. As the development of convolutional neural networks (CNNs), a famous CNN model, AlexNet [12], has been proposed, and various algorithms have been developed based on it. Existing multiple CNNs require 5–6 s per frame to take many crops of images and it leads to reduce the processing performance. This research [9] has conducted a bounding box regression which combines both a sliding window detector and a mask detector on CNN-based detectors to overcome this drawback. As a result, it achieved to decrease the number of bounding boxes and to increase the performance speed. For the lane detection feature, lanes can be detected by adding classes to the CNN used for vehicle detection. Furthermore, it has been suggested that self-driving may be possible using vision technology alone (i.e., without having to attach various sensors to the vehicle) to predict and draw lanes even if the lanes are covered by an object.

2.2 Image Recognition Algorithms for Self-driving Object Detection

Object detection algorithms [13–19] for detecting objects, such as different types of vehicles and traffic lights in a self-driving environment, are widely used. Object detection algorithms are divided into two methods: 2-stage detection and unified detection. A typical algorithm for 2-stage detection is the region-based convolutional network (R-CNN) algorithm. Among them, the faster R-CNN algorithm has been applied to the object detection field. Thereafter, the faster R-CNN algorithm was developed.

R-CNN is used to perform object detection by connecting the CNN, which performs the image classification, and the region proposal algorithm, which suggests the area in which the object may exist in the image. In the process of implementing the R-CNN, approximately 2,000 region proposals are first extracted from the images received as inputs, and the selective search algorithm is used. Next, each region proposal area is cropped and warped from the images, and the CNN is used to extract the feature. Finally, classification is performed for each region proposal feature. The above process provides high algorithm performance, but the localization performance is low. As the CNN has some positional invariance characteristics and predicts a high classification score, it is difficult to find the exact location of an object even if it is not located at the center of the region proposal. To overcome this drawback, the bounding box regression, which can correct the location, is used. However, since the R-CNN requires as many CNN operations as the number of region proposals per image, the inference speed is very slow and the learning process must go through several stages. The fast R-CNN algorithm is an improvement over the R-CNN in terms of improved “speed,” In the case of the R-CNN, if 2,000 regions exist in the image, it is cropped and a CNN operation is conducted in each region. As a result, a total of 2,000 CNN operations are performed. To improve the inefficiency of conducting 2,000 operations, the cropping process can be performed at the level of the feature map rather than at the level of an image to reduce the number of operations to one. However, to perform the classification, the size of the feature should be the same for all examples. In the R-CNN, warping is performed to convert regions of different sizes into images of the same size to perform the CNN operation. However, warping is not a suitable method because information loss can occur. To compensate for the disadvantage, spatial pyramid pooling (SPP) is used. SPP divides an image into a certain number of regions and applies bag of words (BoW) to each region to maintain some local information. In the fast R-CNN, only a single-level pyramid of the SPP layer is used, and thus, it is called the ROI pooling layer. The learning process and testing process become relatively faster by more than a factor of 10 and 20, respectively. However, it takes a relatively long time to extract the region proposal, and the faster R-CNN was proposed to overcome this disadvantage and inefficiency.

The faster R-CNN algorithm is a combination of the region proposal network (RPN) and the fast R-CNN. The faster R-CNN algorithm generates a feature map by processing an image through a pre-trained model. Thereafter, the RPN, which is implemented using convolution, is applied, and the input values of the RPN are the feature maps extracted from the previous base network. An n × n spatial window is slid onto the feature maps to generate the region proposals. Multiple region proposals are predicted for each point determined by the sliding window. The highest number of region proposals is denoted by k, which is called the anchor. Typically, nine anchors exist per point in each sliding window, and three aspect ratios and three scales are combined to have the same central point (X_a, Y_a). The depth of the feature map from the sliding window takes a lower dimension (e.g., 512 depth ≥ 256 depth). Subsequent output values are split into two convolutional layers with a 1 × 1 kernel. In the classification layer, two prediction values are derived per anchor, and this value is a probability value determining whether the object is an object or not. The regression layer (or bounding box adjustment layer) derives four values—ΔX_{center}, ΔY_{center}, Δwidth, and Δheight—for each anchor, and these values are applied to anchors to obtain the final proposals. The training data for training the classifier are anchors and ground truth boxes (data of boxes that a person drew) obtained from the RPN. All the anchors should be classified as either foreground or background, and the criteria for the classification are as follows: an anchor is a foreground anchor if it overlaps with a ground truth box by a large area, and it is a background anchor if the area overlapping with a ground truth box is small. Given that the P value distinguishes the foreground from the background at each anchor, it can be expressed as follows:

$$P = \begin{cases} \text{if IoU} > 0.7 & 1 \\ \text{if IoU} < 0.3 & -1 \\ \text{if otherwise} & 0 \end{cases}$$

where Intersection over Union (IOU), which denotes intersection over union, is defined as follows:

$$\text{IoU} = \frac{A \cap B}{A \cup B}$$

A:anchor, B:ground truth box

The bounding box regression uses four coordinate values and receives values of the proposed regions with different sizes after the Region Proposal Network (RPN) is implemented as the outputs. In other words, the feature maps produced from the CNN as outputs have different sizes. If the maps differ in size, it becomes more difficult to classify them later. To address this issue, a technique called ROI pooling can be used. Using the ROI, feature maps with different sizes can be converted to the same size. The logic of the ROI divides each region proposal into equal-sized sections. The size of the section is the same as the output size of the ROI pooling. It finds the maximum value for each section and produces the maximum value that was found for each section as an output.

The faster R-CNN algorithm can be combined with various feature extractors to increase accuracy. The Inception [20–22] and Resnet models [23, 24] are the most popular feature extractors. The Inception V1 model sparsely connects the CONV (convolutions) layer and densely processes the matrix operation to increase efficiency. However, the change from the existing Inception V1 model to the V2 version involves a heavy convolution operation, which needs to be improved. After the convolution and pooling operations are performed in parallel, they are concated to reduce the computational complexity, and the CONV layer is used to reduce the representational bottleneck. Thereafter, the Inception V3 model, a slight modification of the Inception V2 model, was proposed, and the Inception V4 and Inception-Resnet models were developed by adding features and improving the convergence speed by introducing Resnet. We conducted a simple test using the faster R-CNN Inception V2 algorithm. **Figs. 3** and **4** show the detection results for the stop line and the center line using the faster R-CNN Inception V2, respectively.

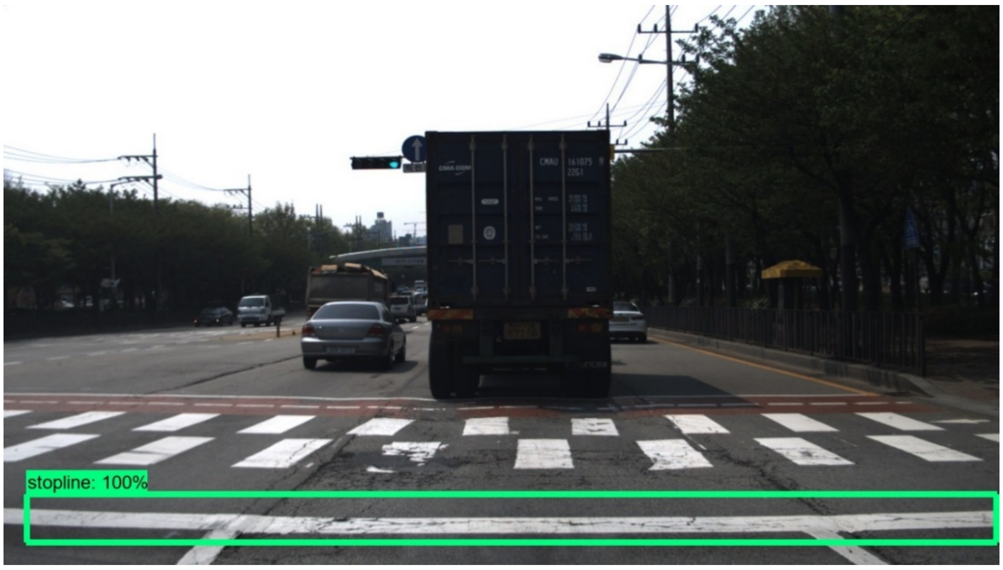

Fig. 3. Example of stop line detections using faster R-CNN inception V2

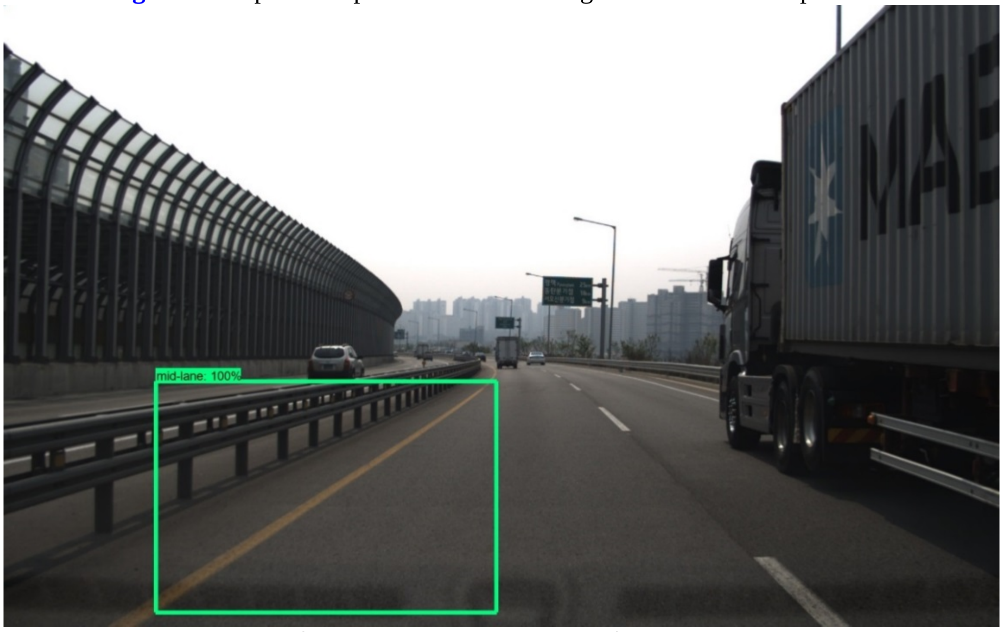

Fig. 4. Example of center line detections using faster R-CNN Inception V2

The Resnet model presents the residual learning framework to help the system easily learn and optimize a deeper and larger network, and to provide higher accuracy even if the depth increases. Resnet is based on the plain network and includes a shortcut connection. Various experiments were conducted as the number of layers was increased, and it was found that the deeper network showed a better performance. **Fig. 5** shows the results of road line detection using the faster R-CNN Resnet 101 algorithm, which is somewhat slower than the faster R-CNN Inception V2 but has higher accuracy.

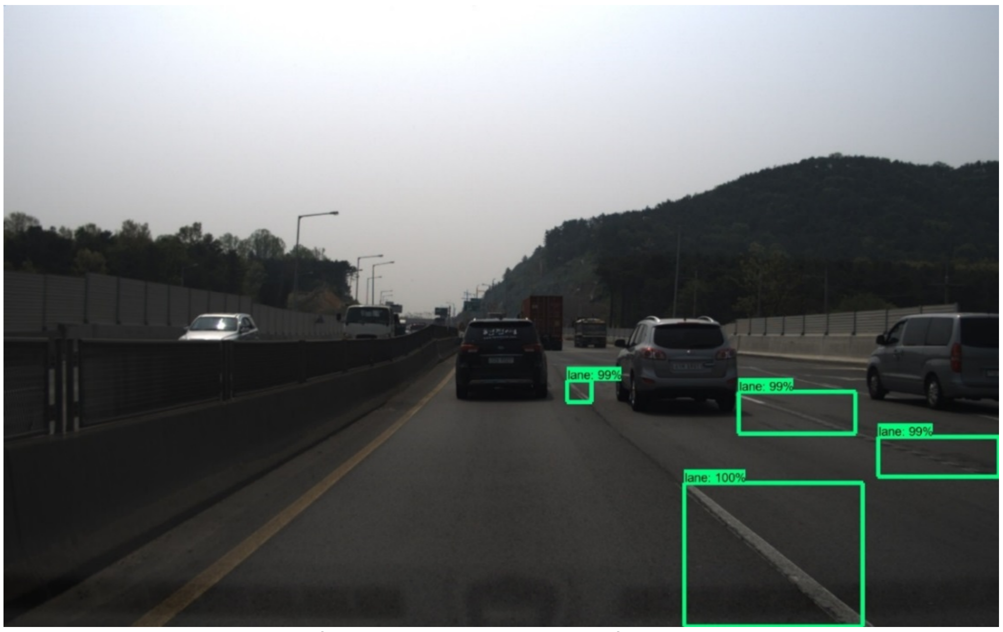

Fig. 5. Example of road line detections using faster R-CNN Resnet 101

It is possible to detect various objects based on the object detection algorithm. Using a Tensorflow detection model, zoo group [25], a pre-trained model can be applied using COCO (Common Object in Context), KITTI (Karlsruhe Institute of Technology and Toyota Technological Institute at Chicago), and open images datasets. These models are combined with algorithms, including single shot multibox detector (SSD), SSDlite, faster R-CNN, region-based fully convolutional networks, and mask R-CNN algorithms, and feature extractor models, including MobileNet, Resnet 50, Inception V2, Resnet 101, and Inception Resnet V2, to provide a feature to detect pedestrians, vehicles, and stop signs.

A typical algorithm of the unified detection algorithm is You Only Look Once (YOLO) [17]. The first version of YOLO was released in 2015, and it considers the bounding box and class probabilities in the image to be a single regression problem to estimate the type and location of an object by looking at the image just once. When an image is received as an input, it is divided into an S × S grid, and each grid cell has B bounding boxes and a confidence score for each bounding box. If there is no object in a cell, the confidence score is 0, and each grid cell has C conditional class probabilities. Each bounding box consists of x, y, w, h, and confidence. x and y indicate the center points of the bounding box, and values relative to the range of the grid cell are entered. Relative values for the width and height of the entire image are entered for w and h. The class-specific confidence score is obtained in the test time by multiplying the conditional class probability by the confidence score of the bounding box. YOLO's network architecture for the image classification model is based on GoogLeNet. Since it consists of 24 convolutional layers and 2 fully connected layers, and has a simple structure, it is much faster than the 2-stage detection algorithm. In addition, the background error is low due to the high contextual understanding of the class as it views the whole image at once. Given that it learns the generalized characteristics of objects, if the artwork is used to test the system after a learning process on the system is performed using a natural image, it provides higher performance than other detection algorithms. However, it shows relatively low accuracy for small objects.

YOLO V2 [18] is an improved version of its predecessor in terms of speed and performance as the first version of YOLO suffered from disadvantages such as low recall and localization errors. To overcome these aspects, batch normalization, high-resolution classifier, convolution with anchor boxes, dimension cluster, direct location prediction, fine-graded features, and multi-scale training processes are performed. Batch normalization is used to alleviate the necessity of regularization and improve convergence. Furthermore, as removing the dropout from the model does not cause overfitting, the dropout can be removed.

The high-resolution classifier increases the size of the existing 224 × 224 inputs to 448 × 448 inputs, and fine-tunes the classification network with 10 epochs using ImageNet to allow it to operate normally at high resolutions. The learned network is fine-tuned at the detection stage and in the convolution with anchor boxes, the location of YOLO's bounding box is directly estimated using the fully connected layers of the feature vector extracted from the fully convolutional feature extractor. However, in the faster R-CNN, the offsets and confidences are estimated based on each anchor box using human-selected hand-picked priors. As the last layer is convolutional, the box location information at every location is expressed as offsets, which simplifies the problem and makes learning easier. The existing class is separated from regional locations, and the class and objectness are separately estimated for every anchor box. Using anchor boxes lowers the accuracy but allows more boxes to be estimated. Dimension clusters are designed to automatically determine the appropriate size of a box by applying k-means clustering to a training set, instead of a person manually determining the size by applying an anchor. In this process, instead of using the Euclidean distance for the distance value, the following equation was used.

$$d(A, B) = 1 - IOU(A, B)$$

A:box, B:centroid

The k-value of k-means is set to 5. These automatically produced boxes are different from the manually produced counterparts and they include low-height and wide boxes or thin and long boxes. The direct location prediction estimates the relative values with respect to the grid cell in YOLO V2. The range of possible values is between 0 and 1, and logistic activation is used for this purpose. The network estimates five bounding boxes for each cell, and each cell estimates five elements of location information: t_x, t_y, t_w, t_h, and t_o. If the location of the cell is $(c_{x,} c_y)$ from the upper left corner of the image and the size of the bounding box is P_w and P_h, the size of the estimated box can be expressed as follows. t_x, t_y, t_w, and t_h are predictions made by YOLO. c_x and c_y are the top left corner of the grid cell of the anchor. p_w and p_h are the width and height of the anchor. b_x, b_y, b_w, b_h are the predicted boundary box. σ(E) is the box confidence score.

$$b_x = \sigma(A) + c_x$$
$$b_y = \sigma(B) + c_y$$
$$b_w = P_w * e^C$$
$$b_h = P_h * e^D$$
$$Pr(object) * IOU(b, object) = \sigma(E)$$

$A{:}t_x$, $B{:}t_y$, $C{:}t_w$, $D{:}t_h$, $E{:}t_o$

The location information is parametrized to make the system easier to learn for stable network use. For fine-graded features, YOLO's 13 × 13 feature map is sufficient to find large objects, but it requires more granular features to find smaller objects. Therefore, another method is used to obtain the features of the 26 × 26 feature map. The original input size of the multi-scale training is 448 but it is changed to 416 to use the anchor box. However, since the model itself consists of convolution and pooling, the size of the model can be changed as much as required. Rather than using a fixed size input, the input image size is randomly changed every 10 batches to continue the learning process. As the input is downsampled to a multiple of 32, the random size should be a multiple of 32. Furthermore, to improve speed, a new model called Darknet-19, which has 19 convolutional layers and 5 maxpooling layers, was proposed in YOLO V2.

In YOLO V3 [19], various minor changes were effected to improve the performance, and the system was changed to use logistic regression for the objectivity score. If one of the bounding boxes overlaps more than the other bounding boxes, the value becomes 1. The boxes that overlap more than a certain threshold are ignored in the estimation process, the value becomes 0.5. The ignored boxes are regarded as having zero loss in the estimation processes for the location and class information, and only the objectness loss is calculated. Each box uses a multi-label classification to estimate the classes in the box. Unlike the previous method, as it can perform well without using Softmax, it uses an independent logistic classifier for each label. Binary cross-entropy loss is used for learning. YOLO V3 estimates the box on three different scales. Features are extracted from three scales in a fashion similar to the feature pyramid network. First, several convolutional layers are added to the existing feature extractor to estimate the 3-D tensor, which contains information about box, objectness, and class. The feature extractor of YOLO V3 is a new network that merges the residual network with YOLO V2. Darknet-53, which has 53 convolutional layers on account of having repeating 3×3 and 1×1 convolutional layers and shortcut connections, was proposed.

Some researchers [26, 27] have extracted information from the self-driving environment to assist self-driving based on the 2-stage detection algorithm or unified detection algorithm. Ref. [26] involved detecting traffic lights in a self-driving environment by combining a feature extractor with the faster R-CNN algorithm. The feature extractor uses various models such as Resnet-101, Inception V2, Inception V3, and MobileNet. There is a tradeoff between the architecture and the feature extractor in terms of speed and accuracy. Therefore, it is also important to select the appropriate architecture and feature extractor according to the application needs, and the Inception V2 model was used in [26]. Based on the collected data sets and the pre-trained model, transfer learning was performed to conduct relearning. Transfer learning is a machine learning technique in which learning is improved for new tasks by transferring knowledge based on previously learned tasks. When the relearning is performed and the application is executed, object detection proceeds based on the relearned data. Ref. [27] claimed that it is difficult to use the 2-stage detection algorithm in a real-time detection environment due to the slow response time, and thus, object detection is performed using the unified detection algorithm, YOLO, to improve the speed. The YOLO V3 algorithm is used to detect traffic participants, including cars, trucks, pedestrians, traffic signs, and traffic lights. For an input video of 1920×1080 pixels, the system detects 23 frames per second on average.

2.3 Lidar Algorithms

Some of the autonomous driving approaches utilized simultaneous localization and mapping (SLAM) methods [28-32] based on a graphical representation in the map. The SLAM methods predicted the relative positions of a robot and infers the map of the environment at the same time. An approach learning globally consistent maps [28] measured relaxation algorithm to remain geometric consistency in a map. The robot repeatedly drove until some arbitrary stopping threshold was satisfied to maintain geometric consistency in the map. A method utilizing the Atlas framework [29] performed SLAM in large areas, and consists of a graph of coordinate frames of limited size with each vertex and edge between adjacent frames. The method decided a transition to a neighbor map-frame and predicts the location in a wide range. The approach built a map with the robot position and local environment. A robust SLAM approach [30] utilized energy nodes to interact with state nodes, for pose or feature measurements. It calculated relative positions between energy and state nodes, using gaussian measurement models, as well as features with uncertainty relative to robot position transitions. GraphSLAM [32] is a mapping algorithm using sparse constraint graphs, that estimates consistent maps and paths using standard optimization techniques with nonlinear data. It incorporated GPS information to localize the moving vehicles and control features. However, the SLAM approach generated estimation errors, and the accumulated errors lead to continuous prediction errors during autonomous driving. It is still limited to solve the problem of building consistent environment maps, as well as localizing the moving vehicles in real-time and in different environments.

One of the main challenges in driverless car technology is to precisely (and robustly) locate the vehicle in real-time. A LIDAR sensor is able to improve the localization performance for self-driving cars, as illustrated in **Fig. 6**. It shows to sense the surrounding area where red lines were extracted using Lidar in Las Data, using the difference in intensity between the floor and the lane. Centimeter-level localization in high definition maps, is important for perception, safe driving, and prediction. A map-based localization approach [33] generates high-resolution environment maps by combining GPS, IMU, wheel odometry, and LIDAR data obtained from sensors installed in vehicles in urban environments. The GPS, IMU, and wheel odometry are utilized to measure autonomous vehicle velocity. LIDAR is a remote sensing instrument which utilizes an infrared laser to map the environment using light detection and ranging. LIDAR identifies road surfaces by filtering normalized brightness and standard deviation using infrared reflectivity. A robust vehicle localization algorithm [34] extends the map-based localization approach, using the ability to learn and improve maps (with probability distributions) over time, in urban environments. This algorithm utilized a probabilistic grid, and each cell contained a gaussian distribution as a spatial grid of infrared remittance values.

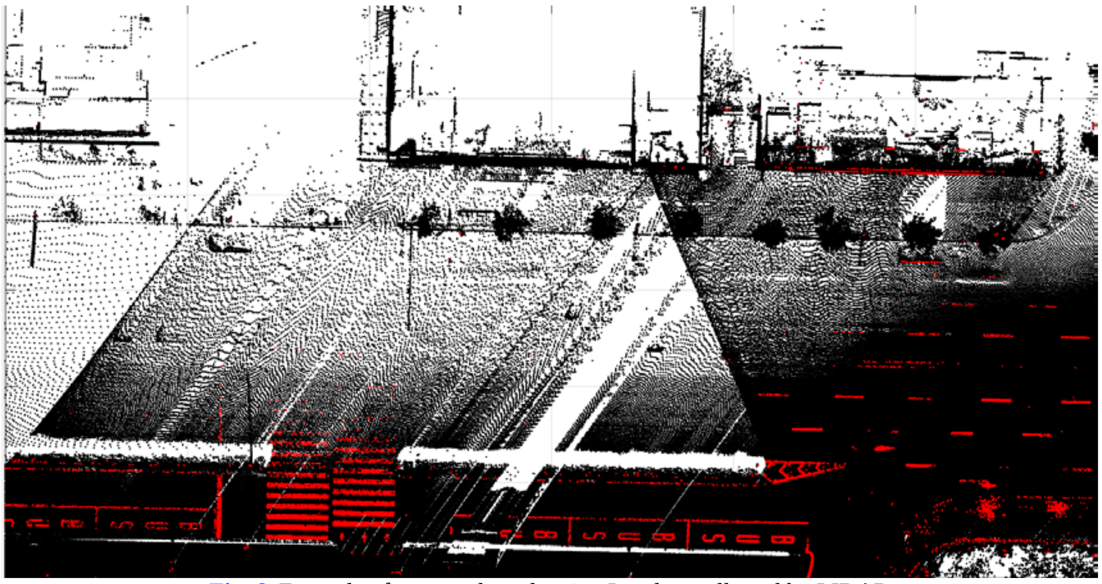

Fig. 6. Example of extracted result using Las data collected by LIDAR

The algorithm generated a grid-cell representation to build a map, and each cell includes the average and variance of infrared reflectivity observed at that location. These algorithms are suitable for localizing autonomous vehicles in the 10-centimeter range, but the performances of the LIDAR sensors are irregular, according to different beam measurements and manufacturers, as well as dynamic environments. These are necessary to handle fine-tuning of each sensor in the vehicles to achieve highly accurate measurements.

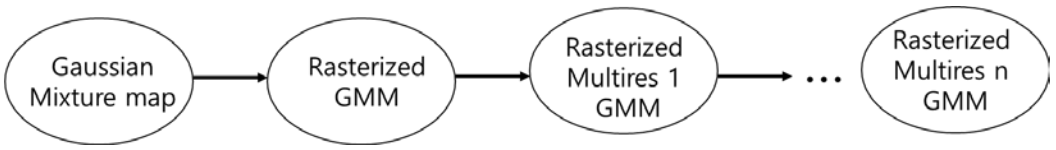

Fig. 7. Rasterization process performed on gaussian mixture map.

A fast LIDAR localization approach [35], with a multi-resolution gaussian mixture algorithm, improved building a traffic map for driverless vehicles in urban environments. A recent road surface reflectivity approach, utilized LIDAR range scanning and three-dimensional light detection to build a precise map. However, the classification results of existing approaches were sensitive to weather conditions when identifying traffic objects on the road. This research made a grid and measured gaussian mixture on z-score distribution in each cell integrating with GPS, odometry, GPS, and pose-graph scan matching constraints. Each grid cell was computed by expectation maximization with a gaussian mixture model. A multi-resolution map was constructed by creating many overlapping coarse blocks to calculate upper-bounds, as illustrated in **Fig. 7**. The upper bounds could be precisely measured by taking each discretization from an observed point cloud. The use of multi-resolution rasterized maps was suitable for robust online localization measurements of self-driving vehicles in different weather conditions. However, this approach is limited to the inference speed which was not satisfied with real-time requirements for the autonomous driving localization.

A real-time localization system [36] with online LIDAR sweeps and intensity map, using a deep learning algorithm, achieved centimeter-level accuracy in different environments. The system utilized deep neural networks that utilize both pre-built LIDAR intensity maps for the localization, and online LIDAR sweeps of the same area. The intensity maps were built from multiple passes, using point cloud data collected from the return strength of a laser pulse, and deep neural networks. The online LIDAR sweeps generated the orthographic bird's eye view intensity images of the ground.

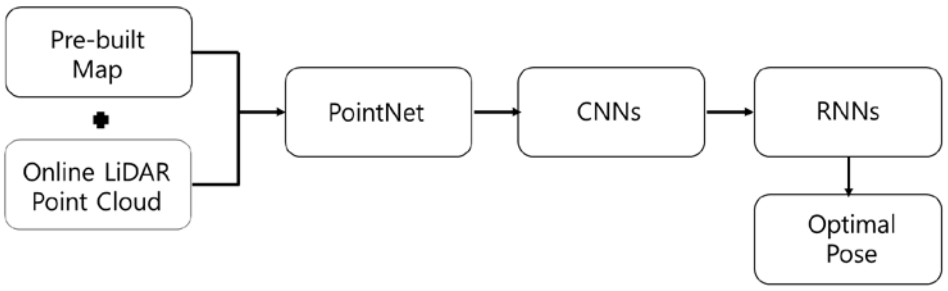

Fig. 8. A Process of a LIDAR localization framework by 3D convolutions.

A Recent research by the Baidu company [66] has developed a LIDAR localization framework [64] using deep learning algorithms in a self-driving environment as shown in **Fig. 8**. It designed and developed a deep neural network architecture using LIDAR scans by 3D convolutions with the eigenvalue of the neighbors of a 3D point. It selected a set of key points extracted by linearities and scattering which are defined as the eigenvalues of its surrounding 3D point. Feature descriptors are extracted by a single mini-PointNet method and regularized by 3D CNNs. It measured the matching probabilities of all the dimensions and then determined optimal results filtering by deep Recurrent Neural Networks (RNNs). It calculated an estimated offset in the process and could make a loss function with both the offset and its ground truth offset as shown in the below equation. The research achieves to boost centimeter-level localization accuracy performance with the measured loss. The estimated offset are $\Delta\hat{x}$, $\Delta\hat{y}$, and $\Delta\hat{\psi}$ in 3D dimension, and the ground truth offset are $\Delta x*$, $\Delta y*$, $\Delta\psi*$, and α is the balancing factor.

$$\text{Loss} = \alpha \cdot (\| A - A' \|^2 + \| B - B' \|^2) + \| C - C' \|^2$$
$$A: \Delta\hat{x},\ A': \Delta x^*,\ B: \Delta\hat{y},\ B': \Delta y^*,\ C: \Delta\hat{\psi},\ C': \Delta\psi^*$$

Current approaches are still limited to delivering precise and accuracy performances for online localization of self-driving vehicles, in real time and in practical environments. Although some of the research for online localization showed desirable accuracy, they need to be fully evaluated and verified in different environments.

2.4 Road Infrastructure Algorithms

In an efficient (self-driving) transportation system, smart traffic light control is important for reducing traffic congestion. Traditional traffic light control [37, 38] provides pre-timed signal control with an assigned (fixed) time, without considering situations in real-time. Intelligent control systems can be dynamically utilized in real-time with an effective deep reinforcement learning model [39-42]. The system can be intelligently adjusted to traffic congestion by observing the traffic. Existing reinforcement learning methods are limited to representing different environments and to building a decision model with the environments. Some recent research [40, 43] utilized deep learning methods to intelligently control traffic lights, combining traffic conditions and light phases to build prediction models. This recent research in traffic light control is still being developed to be applied in adaptive, realistic traffic environments, and they are required to be fully evaluated and verified in various realistic traffic settings.

Pedestrian protection research [44-49] for crossing a road is essential in improving the safety of people who are walking, as well as those inside self-driving vehicles. Crosswalks are one of the dangerous locations on a road and people are often injured in crosswalk accidents. There are several studies for classifying pedestrian flows at traffic lights [44-48]. One study [47] analyzed distances between pedestrian crossings to determine the flow estimations. Another mobile-based study [48] classified users' daily behaviors and predicted the activities requiring them to cross roads. A smart pedestrian crossing management study [49] utilized a fuzzy logic method to dynamically manage traffic light phases with real-time decision making in signalized pedestrian crossing.

A recent cooperative study [50] in intelligent transportation systems, investigated efficiently (and safely) fuse road infrastructure for autonomous vehicles. Vehicle driving can be adjusted depending on environmental conditions, such as weather or road status, and these adjustments help improve safety. This technology aims to manage this transition period efficiently, integrating with sensor techniques, communications and intelligent information management.

3. Infrastructure

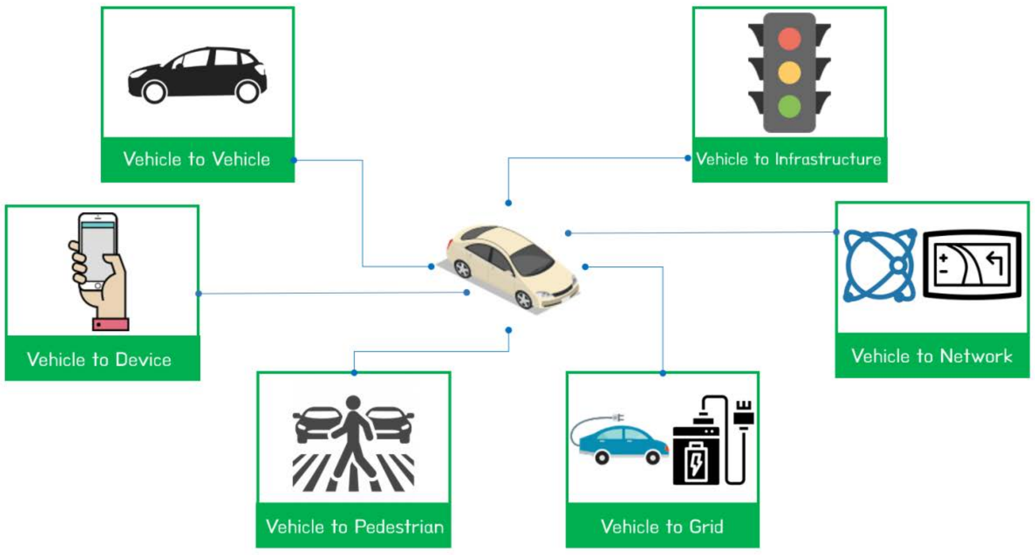

Fig. 9. Infrastructure of V2X

Vehicle-to-everything (V2X) [51] is a wireless communication technology that interacts with information between a vehicle and its surroundings for traffic safety, pollution reduction, economic benefits, time saving, and convenience. V2X can improve traffic efficiency by communicating useful information (e.g. road information, geography, traffic congestion and accidents) with any entity for pollution and accident rate reduction, as well as for safety. V2X technologies are connected to infrastructure, networks, pedestrians, grids, devices, and vehicles, as illustrated in **Fig. 9**. Vehicle-to-infrastructure (V2I) is a wireless exchange model that allows vehicles to communicate information with road infrastructure. V2I technology wirelessly shares and receives information with infrastructure such as cameras, traffic lights, road signs, lane markings, and parking meters. The Vehicle-to-network (V2N) technique typically utilizes broadcast and unicast communications to inform the network of accidents, congestion or other traffic information. This technology can provide on road information to mobile devices, tablets, and navigation systems. Vehicle-to-vehicle (V2V) technology wirelessly exchanges information such as speed, direction, location, loss of stability, and braking between vehicles to avoid accidents. This information can be retransmitted to other cars within several hops on the WiFi network. Vehicle-to-pedestrian (V2P) technology is aimed at improving pedestrian safety (from accidents and fatalities on the road), using wireless communication between vehicles and nearby pedestrians. V2P technology allows vehicles within close proximity to detect and locate pedestrians via their smartphones, or other communication enabled personal devices. Vehicle-to-device (V2D) communication is a wireless connection between vehicles and is for a particular or any electronic devices to exchange information for keyless cars. Vehicle-to-grid (V2G) technology uses and produces the electrical energy stored in vehicle batteries by communication with the electricity grids. The transition of V2G is suitable for vehicle and power grid electricity use to be improved by efficient collaborative strategies.

Self-driving technologies are developing to safely and efficiently drive vehicles to their destinations. The development of associated technologies can be accelerated by improving investment resources and overall road infrastructure. Commercial companies are preparing these road enhancements for the transition to automated vehicles. These companies aim to adapt, upgrade, design and test both physical and digital aspects of the road infrastructure for efficient and safe transportation.

TrafficVision [52] is one of the road infrastructure companies that makes V2I products to share information with components. They develop cameras, traffic lights, street meters, road signs, etc. This helps create an intelligent transport system with V2I sensors that can collect road data to provide road conditions, accidents, traffic congestion, construction areas and so on. SWARCO [53] develops an intelligent road marking product which can communicate with autonomous motors through V2I. They provide road infrastructure for a cooperative system by offering dynamic speed limit and friction coefficient factor functionalities. A ConVeX project [54] builds a V2X integration system into LET and 5G network infrastructure, via both the telecommunication and automotive industries in Germany for traffic safety and efficiency using their mobile solutions. Nissan [55] operated a pilot test using V2V and V2I technologies for autonomous driving. The V2V technology shared speed, location, heading, throttle status, braking, lane change, wipers and gear position information. It also utilized V2I traffic signal phase, stop and intersection sign, and temperature information. A V2P application by Savari [56] can detect a pedestrian’s location and provide alert information through users’ smartphones by using an integration sensor connected to traffic signal systems and receiving a pedestrian signal. It calculates pedestrian locations using a measurement between the pedestrian’s phone and the nearest traffic signal box. BMW [57] provides a mobile application and smart key fob using the V2D technology. The application can control certain vehicle functions remotely, and the smart key fob monitors vehicle information and statistics. AutoCrypt [58] offers a V2G system that can use a plug and charge service and shares information exchanged between the EV and EVSE with a secured authentication service.

Recently, C-V2X [62, 63], also known as Cellular vehicle-to-everything, have been developing to support wide area communication over a cellular network. KT Corporation [65], South Korea's largest telephone company, and Essys company have teamed up with a Qualcomm global technology company and they succeeded developing a V2V and V2V2V(Vehicle to Network to Vehicle) service in a world first using C-V2X and 5G NR(New Radio) networks. The service provides driving videos, stop information, warning messages, etc. in a real time for achieving efficient and safe driving.

There has been much enthusiasm in automated driving functions, and many attempts to improve and advance car technology in the automobile industry. Automotive companies mainly compete to offer reliable and robust automated driving functions. Most of these studies emphasized improvement of road infrastructure for the industry to prepare for the gradual deployment of automated vehicles. Advancement of road infrastructure is required for a highly cost-intensive and time-consuming process. As well as, it needs to develop physical or digital infrastructure factors to be flexible, fast, and cost-effective.

4. Commercial Autonomous Driving Research

4. 1 Commercial Self-driving Vehicle Technologies

A remarkable number of companies manufacture self-driving vehicles or develop self-driving technologies. Each company develops self-driving vehicles and technologies using its own systems and algorithms. **Fig. 9** shows an example of the test platform for self-driving vehicles. Two light detection and ranging (LIDAR) instruments, a communication antenna, and a geographical positioning system (GPS) antenna are attached to the outside of a car. A camera detecting the front is placed inside the car. Table 1 shows a comparative chart of self-driving technologies developed by various companies based on the sensors used.

Table 1. Comparison of commercial self-driving cars by companies

Component	Tesla	Google	Uber	GM
Manufacture their own cars	O	X	X	O
Ultrasonics	O	X	O	X
Cameras	O	O	O	O
Radar	O	O	O	O
Lidar	X	O	O	O
Audio	X	O	X	X
GPS	X	O	O	X
Telematics	X	X	O	O
VIM	X	X	O	X

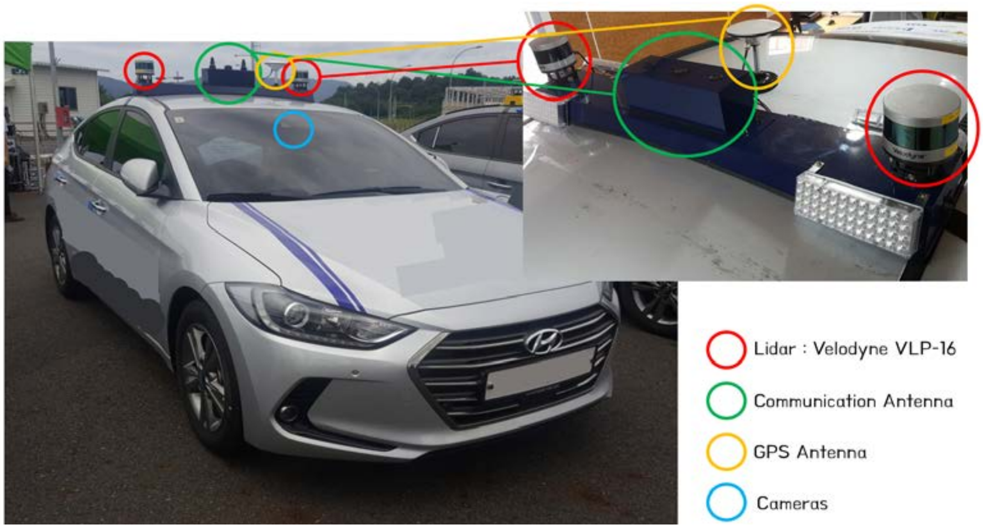

Fig. 9. An example of the test platform for a self-driving car

Tesla [5] develops vehicles and technologies independently. Its self-driving technology is called "Autopilot," A Tesla vehicle uses 8 cameras, 12 ultrasonics sensors, and a radar sensor. Furthermore, the vehicle is equipped with a self-developed onboard computer to process vision, sonar, and radar software. A camera detects the front, sides, and rear of the vehicle, and can detect for distances up to 250 m in the front and up to 100 m at the rear. The combination of various sensors mounted on the vehicle helps it to automatically steer, accelerate, and brake in a lane.

Google launched Waymo, a self-driving vehicle project [6], in 2009. It has developed self-driving technology by collaborating with various vehicle manufacturers but does not manufacture vehicles of its own. The sensors mounted on the vehicle consist of a vision system, lidar system, radar system, and supplemental sensors. The vision system is designed with multiple sets of high-resolution cameras so that the system can obtain a view of 360° instead of 120°, that is the complete visual field of the human eye. The radar uses wavelengths to sense vehicular motion with respect to objects, and the wavelengths can be effectively used in various climates. A radar is placed on the front, rear, and both sides of the vehicle, which makes it easy to track the speed of road users. Lidar emits millions of laser pulses per second for 360° and measures the time that a reflected laser from an object surface returns to the vehicle. The lidar mounted on Waymo consists of short-, medium-, and long-distance lidars. Supplemental sensors include various sensors such as an audio detection system, which can detect sirens, and a GPS.

Uber has also developed self-driving technology [7] but does not manufacture its own cars. Each Uber self-driving vehicle uses lidar, cameras, radar, GPS, a self-driving computer, telematics, ultrasonic sensors, and a vehicle interface module. Lidar can identify objects within 100 m. The camera can identify objects in near, medium, and long ranges for 360°. It also recognizes people and objects within 5 m from the vehicle and helps with parking, lane changes, and other tasks. The GPS helps to collect vehicle location data and map data. A self-driving computer is equipped to run a variety of software. Telematics supports the carrier network. Ultrasonic sensors recognize objects and people within 5 m around the vehicle and help to stop, depart, change lanes, and park. The vehicle interface module is a gateway to communicate with various vehicle control systems.

General Motors (GM) owns several automobile brands and develops self-driving technology [8]. In 2017, Volt EV vehicles were equipped with a self-driving system. The vehicle is equipped with multiple cameras, lidar sensors, a radar sensor, and 4G LTE Connected. A total of 10 cameras were installed to allow a 360° view outside the vehicle. Each camera installed in the self-driving car captures approximately 10 frames per second.

4.2 Precision Road Map

The existing map helps self-driving vehicles to detect the location of the vehicle using GPS and to operate within the range of the road. However, because it is difficult to update according to the changes in the map (e.g., road extension and installation of facilities), the self-driving vehicle may not be able to detect real-time location changes. Furthermore, if the self-driving vehicle cannot detect such changes by using sensors, an accident may occur. Therefore, the map in the self-driving vehicle needs to be updated consistently. Since the environmental changes cause errors when the location of the vehicle is detected by using the Global Navigation Satellite System (GNSS) only, a precision road map was developed to overcome this problem.

Precision map [59] is an electronic map that expresses various kinds of road information necessary for self-driving in three dimensions, and it is accurate within 25 cm. The precision map collects data by mounting the mobile mapping system (MMS) on the vehicle. The MMS is equipped with sensors such as GNSS, inertial navigation system (INS), distance measuring instrument (DMI), lidar, and cameras to collect location and visual information of various topographic objects around the vehicle. Based on the collected data, standard data are produced, and the parts that need to be modified are checked and edited manually. The map comprises various detailed items such as lane markings, road facilities, marking facilities, self-driving facilities, and self-driving prohibited areas. The geographic information system (GIS) software for precision maps consists of five files, among which the file with the extension Shp contains vectors with geometric location information. The file with the extension Shx contains geometric index information, whereas spatial index information is present in the files with the extensions Sbx and Sbn. The file with the extension Dbf contains the database of attribute information. Projections that refer to projection and coordinate system information appear in the file with the extension Prj.

5. Conclusion

Autonomous vehicle technologies are increasingly being developed and are offering innovative solutions for road safety, convenience, and economic and environmental benefits. Vehicle users have to spend the majority of their time paying attention to the road while in their cars; autonomous driving will, however, free up time for activities such as playing mobile games, sleeping, and other work. Self-driving cars can help reduce concerns related to accidents caused by driver, recognition, decision, performance, and non-performance errors. Autonomous vehicles are anticipated to impact our society in terms of its direct implications to our society's behavior and our environment, thus saving on costs, time and effort. Researchers in the field of autonomous vehicles are working to develop advanced technologies to achieve precise and accurate performance, as well as efficiency. They have been improving the self-driving car technologies of intelligence, automation, and connectivity using computer vision, localization, and intelligent communication methods. These studies investigate achieving efficient and safe driving by advancing V2X communication, camera-based vision using deep learning algorithms, and LIDAR-based localization techniques.

Several challenges are however presented. The difficulty of creating maps, demanding driving requirements in many complex social interactions, harsh weather environments, regulations, security, etc., are all challenges the industry faces in increasing precision and accuracy of self-driving vehicle performance (in real-time), and in practical environments. Self-driving vehicles are required to be fully evaluated and verified in different environments. We expect that fully autonomous vehicles will be achieved through more innovation efforts from researchers in the near future.

References

[1] USDOT Releases 2016 Fatal Traffic Crash Data, Article (CrossRef Link)

[2] Howard Daniel, Dai Danielle, "Public Perceptions of Self-driving Cars: The Case of Berkeley, California," *Transportation Research Board 93rd Annual Meeting*, 2013. Article (CrossRef Link)

[3] Iftikhar Ahmad, Rafidah Md Noor, Ihsan Ali, Muhammad Imran, Athanasios Vasilakos, "Characterizing the role of vehicular cloud computing in road traffic management," *International Journal of Distributed Sensor Networks*, May 12, 2017.

[4] KPMG, Center for Automotive Research, "Self-driving cars: The next revolution," New York, 2012. Article (CrossRef Link)

[5] Tesla. Article (CrossRef Link)

[6] Waymo. Article (CrossRef Link)

[7] Uber. Article (CrossRef Link)

[8] General Motors. Article (CrossRef Link)

[9] Brody Huval, Tao Wang, Sameep Tandon, Jeff Kiske, Will Song, Joel Pazhayampallil, Mykhaylo. Andriluka, Pranav Rajpurkar, Toki Migimatsu, Royce Cheng-Yue, Fernando Mujica, Adam Coates, Andrew Y. Ng, "An Empirical Evaluation of Deep Learning on Highway Driving," *Computer Vision and Pattern Recognition*, 17 Apr 2015. Article (CrossRef Link)

[10] Sermanet, Pierre, et al., "Overfeat: Integrated recognition, localization and detection using convolutional networks," *Computer Vision and Pattern Recognition*, 2014. Article (CrossRef Link)

[11] Szegedy, Christian, Alexander Toshev, and Dumitru Erhan, “Deep neural networks for object detection,” *Advances in Neural Information Processing Systems*, 2013. Article (CrossRef Link)
[12] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Proc. of NIPS'12 Proceedings of the 25th International Conference on Neural Information Processing Systems*, Volume 1, Pages 1097-1105, 2012. Article (CrossRef Link)
[13] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” *Advances in neural information processing systems*, pages 91–99, 2015. Article (CrossRef Link)
[14] Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, Kevin Murphy, “Speed/accuracy trade-offs for modern convolutional object detectors,” *Computer Vision and Pattern Recognition*, 25 Apr 2017. Article (CrossRef Link)
[15] Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation Tech report (v5),” *Computer Vision and Pattern Recognition*, 22 Oct 2014. Article (CrossRef Link)
[16] Ross Girshick, “Fast R-CNN,” *Computer Vision and Pattern Recognition*, 8 Sep 2015. Article (CrossRef Link)
[17] Joseph Redmon, Santosh Divvala, Ross Girshick , Ali Farhadi, “You Only Look Once: Unified, Real-Time Object Detection,” *Computer Vision and Pattern Recognition*, 8 Jun 2015. Article (CrossRef Link)
[18] Joseph Redmon, Ali Farhadi, “YOLO 9000: Better, Faster, Stronger,” *Computer Vision and Pattern Recognition*, 25 Dec 2016. Article (CrossRef Link)
[19] Joseph Redmon, Ali Farhadi, “YOLOv3: An Incremental Improvement,” *Computer Vision and Pattern Recognition*, 8 Apr 2018. Article (CrossRef Link)
[20] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, “Going Deeper with Convolutions,” *Computer Vision and Pattern Recognition*, 17 Sep 2014. Article (CrossRef Link)
[21] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, Zbigniew Wojna, “Rethinking the Inception Architecture for Computer Vision,” *Computer Vision and Pattern Recognition*, 11 Dec 2015. Article (CrossRef Link)
[22] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, Alex Alemi, “Inception-v4, Inception-Resnet and the Impact of Residual Connections on Learning,” *Computer Vision and Pattern Recognition*, 23 Aug 2016. Article (CrossRef Link)
[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, “Deep Residual Learning for Image Recognition,” *Computer Vision and Pattern Recognition*, 10 Dec 2015. Article (CrossRef Link)
[24] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, “Identity Mappings in Deep Residual Networks,” *Computer Vision and Pattern Recognition*, 25 Jul 2016. Article (CrossRef Link)
[25] Tensorflow Object detection model zoo. Article (CrossRef Link)
[26] Kulkarni, R., Dhavalikar, S., & Bangar, S, “Traffic Light Detection and Recognition for Self Driving Cars Using Deep Learning,” in *Proc. of 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*, 2018. Article (CrossRef Link)
[27] Corovic, A., Ilic, V., Duric, S., Marijan, M., & Pavkovic, B, “The Real-Time Detection of Traffic Participants Using YOLO Algorithm,” in *Proc. of 2018 26th Telecommunications Forum (TELFOR)*, 2018. Article (CrossRef Link)
[28] T. Duckett, S. Marsland, and J. Shapiro, “Learning globally consistent maps by relaxation,” in *Proc. of 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, 2000. Article (CrossRef Link)
[29] M. Bosse, P. Newman, J. Leonard, M. Soika, W. Feiten, and S. Teller, “Simultaneous localization and map building in large-scale cyclic environments using the atlas framework,” *The International Journal of Robotics Research*, 23(12), 2004. Article (CrossRef Link)
[30] J. Folkesson and H. I. Christensen, “Robust SLAM,” IFAC Proceedings Volumes, 37(8), 722-727, 2004. Article (CrossRef Link).

[31] U. Frese, P. Larsson, and T. Duckett, "A multilevel relaxation algorithm for simultaneous localization and mapping," *IEEE TRANSACTIONS ON ROBOTICS*, VOL. 21, NO. 2, APRIL 2005. Article (CrossRef Link)

[32] S. Thrun and M. Montemerlo, "The GraphSLAM algorithm with applications to large-scale mapping of urban structures," *IJRR*, 25(5/6), 403-429, 2005. Article (CrossRef Link)

[33] J. Levinson, M. Montemerlo, and S. Thrun, "Map-based precision vehicle localization in urban environments," *In Robotics: Science and Systems III*, 2007. Article (CrossRef Link)

[34] J. Levinson and S. Thrun, "Robust vehicle localization in urban environments using probabilistic maps," in *Proc. of 2010 IEEE International Conference on Robotics and Automation*, 2010. Article (CrossRef Link)

[35] Ryan W. Wolcott and Ryan M. Eustice, "Fast LIDAR Localization using Multiresolution Gaussian Mixture Maps," in *Proc. of 2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015. Article (CrossRef Link)

[36] Ioan Andrei Barsan, Shenlong Wang, Andrei Pokrovsky, Raquel Urtasun, "Learning to Localize Using a LiDAR Intensity Map," in *Proc. of Proceedings of the 2nd Conference on Robot Learning, PMLR*, 87, 605-616, 2018. Article (CrossRef Link)

[37] Francois Dion, Hesham Rakha, and Youn-Soo Kang, "Comparison of delay estimates at under-saturated and over-saturated pre-timed signalized intersections," *Transportation Research Part B: Methodological*, Volume 38, Issue 2, Pages 99-122, February 2004. Article (CrossRef Link).

[38] Alan J Miller, "Settings for fixed-cycle traffic signals," *Journal of the Operational Research Society*, Volume 14, Issue 4, pp 373–386, December 1963. Article (CrossRef Link).

[39] Lior Kuyer, Shimon Whiteson, Bram Bakker, and Nikos Vlassis, "Multiagent reinforcement learning for urban traffic control using coordination graphs," *ECML PKDD 2008: Machine Learning and Knowledge Discovery in Databases*, pp 656-671, 2008. Article (CrossRef Link)

[40] Elise van der Pol and Frans A. Oliehoek, "Coordinated Deep Reinforcement Learners for Traffic Light Control," in *Proc. of 30th Conference on Neural Information Processing Systems (NIPS 2016)*, 2016. Article (CrossRef Link)

[41] Marco Wiering, "Multi-agent reinforcement learning for traffic light control," in *Proc. of ICML '00 Proceedings of the Seventeenth International Conference on Machine Learning*, Pages 1151-1158, 2000. Article (CrossRef Link)

[42] Hua Wei, Guanjie Zheng, Huaxiu Yao, and Zhenhui Li, "IntelliLight: A Reinforcement Learning Approach for Intelligent Traffic Light Control," in *Proc. of KDD '18: The 24th ACM SIGKDD*, 2018. Article (CrossRef Link)

[43] Li Li, Yisheng Lv, and Fei-Yue Wang, "Traffic signal timing via deep reinforcement learning," *IEEE/CAA Journal of Automatica Sinica*, 3(3), p247–254, 2016. Article (CrossRef Link)

[44] Ho, T.J, Chung, M.J, "Information-Aided Smart Schemes for Vehicle Flow Detection Enhancements of Traffic Microwave Radar Detectors," *Appl. Sci.*, 6(7), 196, 2016. Article (CrossRef Link).

[45] Zhang, Z, Jia, L, Qin, Y, "Level-of-Service Based Hierarchical Feedback Control Method of Network-Wide Pedestrian Flow," *Mathematical Problems in Engineering*, Volume 2016, Article ID 9617890, 14 pages, 2016. Article (CrossRef Link)

[46] Salvo, G, Caruso, L, Scordo, A, Guido, G, Vitale, A, "Traffic data acquirement by unmanned aerial vehicle," *Eur. J. Remote Sens.*, 50(1), 343–351, 2017. Article (CrossRef Link)

[47] Knoop, V. Daganzo, C, "The Effect of Pedestrian Crossings on Traffic Flow," *EJTIR*, 18(2), 145-157, 2018. Article (CrossRef Link)

[48] Zhang, H, Zhang, C, Wei, Y, Chen, F, "The effects of mobile phone use on pedestrian crossing behavior and safety at unsignalized intersections," in *Proc. of the 4th International Conference on Transportation Information and Safety (ICTIS), Banff, AB, Canada*, 8–10 August 2017. Article (CrossRef Link)

[49] Giovanni Pau, Tiziana Campisi , Antonino Canale, Alessandro Severino, Mario Collotta and Giovanni Tesoriere, "Smart Pedestrian Crossing Management at Traffic Light Junctions through a Fuzzy-Based Approach," *Future Internet*, 10(2), 15, 2018. Article (CrossRef Link)

[50] Meng Lu, Robbin Blokpoel, Julian Schindler, Sven Maerivoet, Evangelos Mintsis, "ICT Infrastructure for Cooperative, Connected and Automated Transport in Transition Areas," in *Proc. of 7th Transport Research Arena TRA 2018, Vienna, Austria*, April 16-19 2018. Article (CrossRef Link)
[51] Jian Wang, Yameng Shao, Yuming Ge and Rundong Yu, "A Survey of Vehicle to Everything (V2X) Testing," *Sensors (Basel)*, 19(2), 334, Jan 2019. Article (CrossRef Link)
[52] TrafficVision. Article (CrossRef Link)
[53] swarco. Article (CrossRef Link)
[54] convex. Article (CrossRef Link)
[55] Nissan Is Rolling Out Cars That Talk to Each Other. Article (CrossRef Link)
[56] savari. Article (CrossRef Link)
[57] Cars: Connecting People. Article (CrossRef Link)
[58] AutoCrypt V2G : Vehicle to Grid. Article (CrossRef Link)
[59] Precision Road Map. Article (CrossRef Link)
[60] Richard Szelisk, "Computer Vision Algorithms and Applications," *Springer*, 2011. Article (CrossRef Link)
[61] Jun Jo, Yukito Tsunoda, Bela Stantic, Alan Wee-Chung Liew, "A Likelihood-Based Data Fusion Model for the Integration of Multiple Sensor Data: A Case Study with Vision and Lidar Sensors," *Robot Intelligence Technology and Applications 4*, pp 489-500, 2017. Article (CrossRef Link)
[62] Qualcomm, 2019, Article (CrossRef Link)
[63] Essys, 2019, Article (CrossRef Link)
[64] Weixin Lu, Yao Zhou, Guowei Wan, Shenhua Hou, Shiyu Song, "L3-Net: Towards Learning Based LiDAR Localization for Autonomous Driving," in *Proc. of The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6389-6398, 2019, Article (CrossRef Link)
[65] KT. Article (CrossRef Link)
[66] Baidu Autonomous Driving Unit. Article (CrossRef Link)

Jeonghwan Gwak received the Ph.D. degree in machine learning and artificial intelligence from Gwangju Institute of Science and Technology, Gwangju, Korea in 2014. From 2002 to 2007, he had worked for several companies and research institutes as a researcher and a chief technician. From 2014 to 2016, he worked as a postdoctoral researcher in GIST and from 2016 to 2017 as a research professor. From 2017 to 2019, he was a research professor in Bio-medical Research Institute & Department of Radiology at Seoul National University Hospital, Seoul, Korea. From 2019, he joined Korea National University of Transporta-tion (KNUT) and he is the director of Applied Machine Intelligence laboratory. Since 2018, he is Associate Editor of IEEE Access, Guest Editor of IJCVR and served as Chairman and PC/TPC members in many artificial intelli-gence, machine learning, and computer vision conferences. His current research interests include deep learning, com-puter vision, image and video processing, evolutionary computation, optimization, and relevant applications of medical and visual surveillance systems.

Ju-Ho Jung received the B.S. degrees in Computer Science from Korea National University of Transportation, Korea, in 2019. Juho Jung is currently a M.S. in the Department of Computer Science, Korea National University of Transportation. Juho Jung is interested in Computer Vision, Machine Learning, Deep Learning, IoT, and Self-driving.

RyumDuck Oh is a Professor in the Computer Information Technology at Korea National University of Transportation. He received the Ph.D. degree in Computer Science at Hongik University Seoul, Korea, in 1993. He received the B.S. and M.S. degrees in Computer Science from Hongik University in 1986 and 1988, respectively. His research interest includes the data sensing and analysis in the IoT platform.

MANBOK PARK received the B.S. degrees in mechanical engineering from Inha university, Korea, and the M.S. degrees in mechanical engineering from Korea Advanced Institute of Science and Technology (KAIST), Korea in 2002, and Ph.D. degrees in graduate school of convergence science and technology from Seoul National University, Korea in 2014. He had been worked for 15 years at Mando corporation, automotive part company. He is currently an assistant professor of Electrical Engineering at Korea national university of transportation. His research interests are autonomous vehicle, SLAM, precision map, risk assessment and artificial intelligence.

Mukhammad Abdu Kayumbek Rakhimov received the B.S. degrees in Computer engineering from Tashkent University of Information Technologies, Uzbekistan, in 2016. Mukhammad Abdu Kayumbek Rakhimov is currently a M.S. in the Department of Computer Science, Korea National University of Transportation. Mukhammad Abdu Kayumbek Rakhimov is interested in Database, IOT, and Big data.

Jun-Ho Ahn is an Assistant Professor in the Computer Information Technology at Korea National University of Transportation. Junho Ahn received a Ph.D. degree in Computer Science at University of Colorado at Boulder in 2013. Junho Ahn is interested in intelligent extensive knowledge of vision, artificial intelligence algorithms, self-driving car systems, mobile systems, embedded systems, sensor networks, and the prospects for uniting these areas. Much of his research involved intelligent mobile and self-driving car application systems, in which he designed to intelligent fuse multi-modal mobile sensor data.