Computing Systems for Autonomous Driving: State of the Art and Challenges

Liangkai Liu, *Graduate Student Member, IEEE*, Sidi Lu, Ren Zhong, Baofu Wu, Yongtao Yao, Qingyang Zhang, and Weisong Shi, *Fellow, IEEE*

(Invited Paper)

***Abstract*—The recent proliferation of computing technologies (e.g., sensors, computer vision, machine learning, and hardware acceleration) and the broad deployment of communication mechanisms (e.g., dedicated short-range communication, cellular vehicle-to-everything, 5G) have pushed the horizon of autonomous driving, which automates the decision and control of vehicles by leveraging the perception results based on multiple sensors. The key to the success of these autonomous systems is making a reliable decision in real-time fashion. However, accidents and fatalities caused by early deployed autonomous vehicles arise from time to time. The real traffic environment is too complicated for current autonomous driving computing systems to understand and handle. In this article, we present state-of-the-art computing systems for autonomous driving, including seven performance metrics and nine key technologies, followed by 12 challenges to realize autonomous driving. We hope this article will gain attention from both the computing and automotive communities and inspire more research in this direction.**

***Index Terms*—Autonomous driving, challenges, computing systems.**

I. Introduction

RECENTLY, with the vast improvements in computing technologies, e.g., sensors, computer vision, machine learning, and hardware acceleration, and the wide deployment of communication mechanisms, e.g., dedicated short-range communications (DSRCs), cellular vehicle-to-everything (C-V2X), and 5G, autonomous driving techniques have attracted massive attention from both the academic and automotive communities [1], [2]. According to [3], the global autonomous driving market expects to grow up to $173.15B by 2030. Many automotive companies have made enormous investments in this domain, including ArgoAI, Audi, Baidu, Cruise, Mercedes-Benz, Tesla, Uber, and Waymo, to name a few [4]–[7]. Several fleets of society of automotive engineers (SAEs) L4 vehicles have been deployed around the world [8], [9].

To achieve autonomous driving, determining how to make the vehicle understand the environment correctly and make safe controls in real time is the essential task. Rich sensors, including camera, light detection and ranging (LiDAR), radar, inertial measurement unit (IMU), global navigation satellite system (GNSS), and sonar, as well as powerful computation devices, are installed on the vehicle [10]–[14]. This design makes autonomous driving a real powerful "computer on wheels." In addition to hardware, the rapid development of deep learning algorithms in object/lane detection, simultaneous localization and mapping (SLAM), and vehicle control also promotes the real deployment and prototyping of autonomous vehicles (AVs) [15]–[18]. The AV's computing systems are defined to cover everything (excluding the vehicle's mechanical parts), including sensors, computation, communication, storage, power management, and full-stack software. Plenty of algorithms and systems are designed to process sensor data and make a reliable decision in real time.

However, news of fatalities caused by early developed AVs arises from time to time. Until August 2020, five self-driving car fatalities happened for level-2 autonomous driving: four of them from Tesla and one from Uber [19]. Table I summarizes the date, place, company, and reasons for these five fatalities. The first two fatalities attributed to Tesla happened in 2016 with the first accident occurring because neither the Autopilot system nor the driver failed to recognize the truck under thick haze. The vehicle in the second incident mistook the truck for open space. In the 2018 incident involving Tesla, the autopilot failed to recognize the highway divider and crashed into it. The most recent fatality from Tesla happened in 2019 because the vehicle failed to recognize a semitrailer. The fatality from Uber happened because the autonomous driving system failed to recognize that pedestrians jaywalk.

In summary, all four incidents associated with Tesla are due to perception failure, while Uber's incident happened because of the failure to predict human behavior. Another fact to pay attention to is that currently, the field testing of level 2 autonomous driving vehicles mostly happens in places with good weather and light traffic conditions, such as Arizona and Florida. The real traffic environment is too complicated for the current autonomous driving systems to understand and handle easily. The objectives of level 4 and level 5 autonomous driving require colossal improvement of the computing systems for AVs.

This article presents state-of-the-art computing systems for autonomous driving, including seven performance metrics and nine key technologies, followed by eleven challenges and

Manuscript received September 5, 2020; revised November 19, 2020; accepted November 24, 2020. Date of publication December 9, 2020; date of current version April 7, 2021. *(Corresponding author: Weisong Shi.)*

Liangkai Liu, Sidi Lu, Ren Zhong, Baofu Wu, Yongtao Yao, and Weisong Shi are with the Department of Computer Science, Wayne State University, Detroit, MI 48202 USA (e-mail: liangkai@wayne.edu; lu.sidi@wayne.edu; ren.zhong@wayne.edu; baofu.wu@wayne.edu; yongtaoyao@wayne.edu; weisong@wayne.edu).

Qingyang Zhang is with the School of Computer Science and Technology, Anhui University, Hefei 230601, China (e-mail: qingyang@thecarlab.org).

Digital Object Identifier 10.1109/JIOT.2020.3043716

TABLE I
LIST OF FATALITIES CAUSED BY LEVEL 2 AUTONOMOUS DRIVING VEHICLES

Date	Place	Company	Reason
20 Jan. 2016	Handan, Hebei China	Tesla	fail to recognize truck under a thick haze
07 May 2016	Williston, Florida USA	Tesla	mistook the truck for open sky
18 Mar. 2018	Tempe, Arizona USA	Uber	fail to recognize pedestrians jaywalk at night
23 Mar. 2018	Mountain View, California USA	Tesla	fail to recognize the highway divider
1 Mar. 2019	Delray Beach, Florida USA	Tesla	fail to recognize semi-trailer

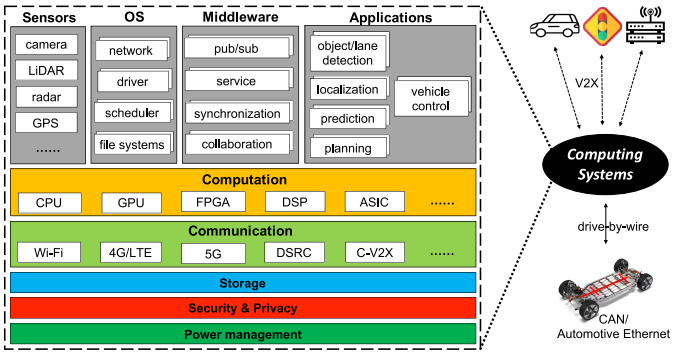

Fig. 1. Representative reference architecture of the computing system for autonomous driving.

TABLE II
END-TO-END APPROACHES FOR AUTONOMOUS DRIVING

Work	Methods	Characteristics
[24]	supervised DNN	raw image to steering angles for off-road obstacle avoidance on mobile robots
[25]	supervised DNN	map an input image to a small number of key perception indicators
[26]	supervised DNN	CNN to map raw pixels from a camera directly to steering commands
[27]	supervised DNN	FCN-LSTM network to predict multi-modal discrete and continuous driving behaviors
[28]	DQN	automated driving framework in simulator environment
[29]	DQN	lane following in a countryside road without traffic using a monocular image as input

opportunities to realize autonomous driving. The remainder of this article are organized as follows. Section II discusses the reference architecture of the computing systems for autonomous driving. In Section III, we show some metrics used in the evaluation of the computing system. Section IV discusses the key technologies for autonomous driving. Section V presents possible challenges. Finally, this article concludes in Section VI.

II. REFERENCE ARCHITECTURE

As an essential part of the whole autonomous driving vehicle, the computing system plays a significant role in the whole pipeline of driving autonomously. There are two types of designs for computing systems on AVs: 1) modular based and 2) end-to-end based.

Modular design decouples the localization, perception, control, etc., as separate modules and makes it possible for people with different backgrounds to work together [20]. The DARPA challenges are a milestone for the prototyping of autonomous driving vehicles, including Boss from CMU [21], Junior from Stanford [22], TerraMax and BRAiVE from University of Parma [23], etc. Their designs are all based on modules, including perception, mission planning, motion planning, and vehicle controls. Similarly, the survey fleet vehicles developed by Google and Uber are also modular based [4], [7]. The main differences for these AV prototypes are the software and the configuration of sensors, such as camera, LiDAR, radar, etc.

In contrast, the end-to-end-based design is largely motivated by the development of artificial intelligence. Compared with modular design, end-to-end system purely relies on machine learning techniques to process the sensor data and generate control commands to the vehicle [24]–[29]. Table II shows a detailed description of these end-to-end designs. Four of them are based on supervised DNNs to learn driving patterns and behaviors from human drivers. The remaining two are based on deep Q-network (DQN), which learns to find the optimum driving by itself. Although the end-to-end-based approach promises to decrease the modular design's error propagation and computation complexity, there is no real deployment and testing of it [30].

As most prototypes are still modular based, we choose it as the basis for the computing system reference architecture. Fig. 1 shows a representative reference architecture of the computing system on AVs. Generally, the computing system for autonomous driving vehicles can be divided into computation, communication, storage, security and privacy, and power management. Each part covers four layers with sensors, operating system (OS), middleware, and applications. The following paragraphs will discuss the corresponding components.

For safety, one of the essential tasks is to enable the "computer" to understand the road environment and send correct control messages to the vehicle. The whole pipeline starts with the sensors. Plenty of sensors can be found on an autonomous driving vehicle: camera, LiDAR, radar, GPS/GNSS, ultrasonic, IMU, etc. These sensors capture real-time environment information for the computing system, such as the eyes of human beings. OS plays a vital role between hardware devices (sensors, computation, communication) and applications. Within the OS, drivers are bridges between the software and hardware devices; the network module provides the abstraction communication interface; the scheduler manages the competition to all the resources; and the file system provides the abstraction to all the resources. For safety-critical scenarios, the OS must satisfy real-time requirements.

As the middle layer between applications and OSs [31], middleware provides usability and programmability to develop and improve systems more effectively. Generally, middleware supports publish/subscriber, remote procedure call (RPC) or

service, time synchronization, and multisensor collaboration. A typical example of the middleware system is the robot OS (ROS) [32]. On top of the OS and middleware system, several applications, including object/lane detection, SLAM, prediction, planning, and vehicle control, are implemented to generate control commands and send them to the vehicle's drive-by-wire system. Inside the vehicle, several electronic control units (ECUs) are used to control the brake, steering, etc., which are connected via the controller area network (CAN bus) or automotive Ethernet [33]. In addition to processing the data from onboard sensors, the autonomous driving vehicle is also supposed to communicate with other vehicles, traffic infrastructures, pedestrians, etc., as complementary.

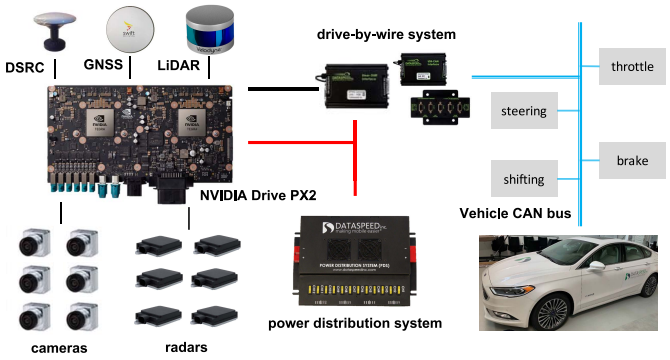

Fig. 2. Typical example of a computing system for autonomous driving.

III. Metrics for Computing System

According to the report about autonomous driving technology from the National Science and Technology Council (NSTC) and the United States Department of Transportation (USDOT) [34], ten technology principles are designed to foster research, development, and integration of AVs and guide consistent policy across the U.S. Government. These principles cover safety, security, cyber security, privacy, data security, mobility, accessibility, etc. Corresponding to the autonomous driving principles, we define several metrics to evaluate the computing system's effectiveness.

Accuracy: Accuracy is defined to evaluate the difference between the detected/processed results with the ground truth. Take object detection and lane detection, for example, the intersection over union (IOU) and mean average precision (mAP) are used to calculate the exact difference between the detected bounding box of objects/lanes and the real positions [35], [36]. For vehicle controls, the accuracy would be the difference between the expected controls in break/steering with the vehicle's real controls.

Timeliness: Safety is always the highest priority. Autonomous driving vehicles should be able to control themselves autonomously in real time. According to [20], if the vehicle is self-driving at 40 km per hour in an urban area and wants the control effective every 1 m, then the whole pipeline's desired response time should be less than 90 ms. To satisfy the desired response time, we need each module in the computing system to finish before the deadline.

Power: Since the onboard battery powers the whole computing system, the computing system's power dissipation can be a big issue. For electrical vehicles, the computing system's power dissipation for autonomous driving reduces the vehicle's mileage with up to 30% [37]. In addition to mileage, heat dissipation is another issue caused by high power usage. Currently, the NVIDIA Drive PX Pegasus provides 320 INT8 TOPS of AI computational power with a 500-W budget [38]. With the power budget of sensors, communication devices, etc., the total power dissipation will be higher than 1000 watts. The power budget is supposed to be a significant obstacle for producing the real autonomous driving vehicle.

Cost: Cost is one of the essential factors that affect the board deployment of AVs. According to [39] and [40], the cost of a level 4 autonomous driving vehicle attains $300 000, in which the sensors, computing device, and communication device cost almost 4200 000. In addition to the hardware cost, the operator training and vehicle maintenance cost of AVs (such as insurance, parking, and repair) are also more expensive than traditional vehicles.

Reliability: To guarantee the safety of the vehicle, reliability is a big concern [41]. On one hand, the worst case execution time is supposed to be longer than the deadline. Interruptions or emergency stops should be applied in such cases. On the other hand, failures happen in sensors, computing/communication devices, algorithms, and systems' integration [42]. How to handle these potential failures is also an essential part of the design of the computing system.

Privacy: As the vehicle captures a massive amount of sensor data from the environment, vehicle data privacy becomes a big issue. For example, the pedestrian's face and the license plate captured by the vehicle's camera should be masked as soon as possible [43]. Furthermore, who owns the driving data is also an important issue, which requires the system's support for data access, storage, and communication [44].

Security: The secureness of the onboard computing system is essential to the success of autonomous driving since, ultimately, the computing system is responsible for the driving process. Cyber attacks can be launched quickly to any part of the computing system [44], [45]. We divide the security into four aspects: 1) sensing security; 2) communication security; 3) data security; and 4) control security [46], [47]. We envision that the onboard computing system will have to pass a certain security test level before deploying it into real products.

IV. Key Technologies

An AV involves multiple subjects, including computing systems, machine learning, communication, robotics, mechanical engineering, and systems engineering, to integrate different technologies and innovations. Fig. 2 shows a typical example of autonomous driving vehicles called *Hydra*, which is developed by The CAR lab at Wayne State University [48]. An NVIDIA Drive PX2 is used as the vehicle computation unit (VCU). Multiple sensors, including six cameras, six radars, one LiDAR, one GNSS antenna, and one DSRC antenna, are installed for sensing and connected with VCU. The CAN bus is used to transmit messages between different ECUs controlling steering, throttle, shifting, brake, etc. Between the NVIDIA Drive PX2 and the vehicle's CAN bus, a drive-by-wire system

is deployed as an actuator of the vehicle control commands from the computing system. Additionally, a power distribution system is used to provide extra power for the computing system. It is worth noting that the computing system's power distribution is nonnegligible in modern AVs [49]. In this section, we summarize several key technologies and discuss their state of the art.

A. Sensors

1) Cameras: In terms of usability and cost, cameras are the most popular sensors on autonomous driving vehicles. The camera image gives straightforward 2-D information, making it useful in some tasks, such as object classification and lane tracking. Also, the range of the camera can vary from several centimeters to near 100 m. The relatively low cost and commercialization production also contribute to the complete deployment in the real autonomous driving vehicle. However, based on lights, the camera's image can be affected by low lighting or bad weather conditions. The usability of the camera decreases significantly under heavy fog, raining, and snowing. Futhermore, the data from the camera are also a big problem. On average, every second, one camera can produce 20–40 MB of data.

2) Radar: The radar's full name is radio detection and ranging, which means to detect and get the distance using radio. The radar technique measures the time of flight (TOF) and calculates the distance and speed. Generally, the working frequency of the vehicle radar system is 24 or 77 GHz. Compared with 24 GHz, 77 GHz shows higher accuracy in distance and speed detection. Besides, 77 GHz has a smaller antenna size, and it has less interference than 24 GHz. For 24-GHz radar, the maximum detection range is 70 m, while the maximum range increases to 200 m for 77-GHz radar. According to [10], the price for continental's long-range radar can be around $3000, which is higher than the camera's price. However, compared with a camera, radar is less affected by the weather and low lighting environment, making it very useful in some applications like object detection and distance estimation. The data size is also smaller than the camera. Each radar produces 10–100 KB per second.

3) LiDAR: Similar to Radar, LiDAR's distance information is also calculated based on the TOF. The difference is that LiDAR uses the laser for scanning, while radar uses electromagnetic waves. LiDAR consists of a laser generator and a high accuracy laser receiver. LiDAR generates a 3-D image of objects, so it is widely used to detect static objects and moving objects. LiDAR shows good performance with a range from several centimeters to 200 m, and the accuracy of distance goes to centimeter level. LiDAR is widely used in object detection, distance estimation, edge detection, SLAM [17], [50], and high-definition (HD) map generation [18], [51]–[53]. Compared with the camera, LiDAR shows larger sensing range and its performance is less affected by bad weather and low lighting. However, in terms of the cost, LiDAR seems less competitive than camera and radar. According to [11], 16 lines Velodyne LiDAR costs almost $8000, while the Velodyne VLS-128E costs over $100 000. High costs restrict the wide deployment of LiDAR on AVs, contributing to the AV's high cost. LiDAR can generate almost 10–70 MB data per second, a huge amount of data for the computing platform to process in real time.

4) Ultrasonic Sensor: Ultrasonic sensor is based on ultrasound to detect the distance. Ultrasound is a particular sound that has a frequency higher than 20 kHz. The distance is also detected by measuring TOF. The ultrasonic sensor's data size is close to the radar's, which is 10–100 KB per second. Besides, the ultrasonic sensor shows good performance in bad weather and low lighting environment. The ultrasonic sensor is much cheaper than the camera and radar. The price of the ultrasonic sensor is always less than $100. The shortcoming of ultrasonic sensors is the maximum range of only 20 m, limiting its application to short-range detection like parking assistance.

5) GPS/GNSS/IMU: Except for sensing and perception of the surrounding environment, localization is also a significant task running on top of the autonomous driving system. In the localization system of the AV, GPS, GNSS, and IMU are widely deployed. GNSS is the name for all the satellite navigation systems, including GPS developed by the U.S., Galileo from Europe, and BeiDou Navigation Satellite System (BDS) [12] from China. The accuracy of GPS can vary from several centimeters to several meters when different observation values and different processing algorithms are applied [13]. The strengths of GPS are low costs, and the nonaccumulation of error over time. The drawback of GPS is that the GPS deployed on current vehicles only has accuracy within 1 m, and the GPS requires an unobstructed view in the sky, so it does not work in environments such as tunnels, for example. Besides, the GPS sensing data update every 100 ms, which is not enough for the vehicle's real-time localization.

IMU consists of gyroscopes and accelerometers. Gyroscopes are used to measure the axes' angular speed to calculate the carrier's position. In comparison, the accelerometer measures the object's three axes' linear acceleration and can be used to calculate the carrier's speed and position. The strength of IMU is that it does not require an unobstructed view from the sky. The drawback is that the accuracy is low, and the error is accumulated with time. IMU can be a complementary sensor to the GPS because it has an updated value every 5 ms, and it works appropriately in environments such as tunnels. Usually, a Kalman filter is applied to combine the sensing data from GPS and IMU to get fast and accurate localization results [14].

Table III shows a comparison of sensors, including camera, radar, LiDAR, and ultrasonic sensors with human beings. From the comparison, we can easily conclude that although humans have strength in the sensing range and show more advantaged application scenarios than any sensor, the combination of all the sensors can do a better job than human beings, especially in bad weather and low lighting conditions.

B. Data Source

1) Data Characteristics: As we listed before, various sensors, such as GPS, IMU, camera, LiDAR, and radar, are equipped in AVs, and they will generate hundreds of megabytes of data per second, fed to different autonomous

TABLE III
COMPARISONS OF CAMERA, RADAR, LIDAR, AND ULTRASONIC SENSOR

Metrics	Human	Camera	Radar	LiDAR	Ultrasonic
Techniques	-	Lights	Electromagnetic	Laser Reflection	Ultrasound
Sensing Range	0-200m	0-100m	1cm-200m (77GHz) 1cm-70m (24GHz)	0.7-200m	0-20m
Cost	-	∼$500	∼$3,000	$5,000 - $100,000	∼$100
Data per second	-	20-40MB	10-100KB	10-70MB	10-100KB
Bad weather functionality	Fair	Poor	Good	Fair	Good
Low lighting functionality	Poor	Fair	Good	Good	Good
Application Scenarios	Object Detection Object Classification Edge Detection Lane Tracking	Object Classification Edge Detection Lane Tracking	Object Detection Distance Estimation	Object Detection Distance Estimation Edge Detection	Object Detection Distance Estimation

driving algorithms. The data in AVs could be classified into two categories: 1) real-time data and 2) historical data. Typically, the former is transmitted by a messaging system with the Pub/Sub pattern in most AVs solutions, enabling different applications to access one data simultaneously. Historical data include application data. The data persisted from real-time data, where structured data, i.e., GPS, are stored into a database, and unstructured data, i.e., video, are stored as files.

2) Dataset and Benchmark: Autonomous driving dataset is collected by survey fleet vehicles driving on the road, which provides the training data for research in machine learning, computer vision, and vehicle control. Several popular datasets provide benchmarks, which are rather useful in autonomous driving systems and algorithms design. A few popular datasets are as follows.

1) *KITTI:* As one of the most famous autonomous driving dataset, the KITTI [54] dataset covers stereo, optical flow, visual odometry, 3-D object detection, and 3-D tracking. It provides several benchmarks, such as stereo, flow, scene, optical flow, depth, odometry, object tracking [55], road, and semantics [56].
2) *Cityscapes:* For the semantic understanding of urban street scenes, the Cityscapes [57] dataset includes 2-D semantic segmentation on pixel-level, instance-level, and panoptic semantic labeling, and provides corresponding benchmarks on them.
3) *BDD100K:* As a large scale and diverse driving video database, BDD100K [58] consists of 100 000 videos and covers different weather conditions and times of the day.
4) *DDD17:* As the first end-to-end dynamic and active-pixel vision sensors (DAVISs) driving dataset, DDD17 [59] has more than 12 h of DAVIS sensor data under different scenarios and different weather conditions, as well as vehicle control information, such as steering, throttle, and brake.

3) Labeling: Data labeling is an essential step in a supervised machine learning task, and the quality of the training data determines the quality of the model. A few different types of annotation methods are as follows.

1) *Bounding Boxes:* The most commonly used annotation method (rectangular boxes) in object detection tasks is to define the location of the target object, which can be determined by x and y-axis coordinates in the upper-left corner and the lower-right corner of the rectangle.
2) *Polygonal Segmentation:* Since objects are not always rectangular, polygonal segmentation is another annotation approach where complex polygons are used to define the object's shape and location in a considerably precise way.
3) *Semantic Segmentation:* It is a pixelwise annotation, where every pixel in an image is assigned to a class. It is primarily used in cases where environmental context is essential.
4) *3-D Cuboids* They provide 3-D representations of the objects, allowing models to distinguish features, such as volume and position in a 3-D space.
5) *Key-Point and Landmark:* They are used to detect small objects and shape variations by creating dots across the image. As to the annotation software, MakeSense.AI [60], LabelImg [61], VGG image nnotator [62], LabelMe [63], Scalable [64], and RectLabel [65] are popular image annotation tools.

C. Autonomous Driving Applications

Plenty of algorithms are deployed in the computing system for sensing, perception, localization, prediction, and control. In this part, we present the state-of-the-art works for algorithms, including object detection, lane detection, localization and mapping, prediction and planning, and vehicle control.

1) Object Detection: Accurate object detection under challenging scenarios is essential for real-world deep learning applications for AVs [66]. In general, it is widely accepted that the development of object detection algorithms has gone through two typical phases: 1) conventional object detection phase and 2) deep learning supported object detection phase [67]. Viola Jones Detectors [68], histogram of oriented gradients (HOGs) feature descriptor [69], and deformable part-based model (DPM) [70] are all the typical traditional object detection algorithms. Although today's most advanced approaches have far exceeded the accuracy of traditional methods, many dominant algorithms are still deeply affected by their valuable insights, such as hybrid models, bounding box regression, etc. As to the deep learning-based object detection approaches, the state-of-the-art methods include the regions with CNN features (RCNN) series [36], [71]–[73], single shot

multibox detector (SSD) series [74], [75], and you only look once (YOLO) series [15], [76], [77]. Girshick *et al.* [36], [78] first introduced deep learning into the object detection field by proposing RCNN in 2014. Later on, Fast RCNN [71] and Faster RCNN [72] were developed to accelerate detection speed. In 2015, the first one-stage object detector, i.e., YOLO, was proposed [76]. Since then, the YOLO series algorithms have been continuously proposed and improved, for example, YOLOv3 [15] is one of the most popular approaches, and YOLOv4 [79] is the latest version of the YOLO series. To solve the tradeoff problem between speed and accuracy, Liu *et al.* proposed SSD [74] in 2015, which introduces the regression technologies for object detection. Then, RetinaNet was proposed in 2017 [80] to further improve detection accuracy by introducing a new loss function to reshape the standard cross-entropy loss.

2) Lane Detection: Performing accurate lane detection in real time is a crucial function of advanced driver-assistance systems (ADAS) [16], since it enables AVs to drive themselves within the road lanes correctly to avoid collisions, and it supports the subsequent trajectory planning decision and lane departure.

Traditional lane detection approaches (e.g., [81]–[86]) aim to detect lane segments based on diverse handcrafted cues, such as color-based features [87], the structure tensor [88], the bar filter [89], and ridge features [90]. This information is usually combined with a Hough transform [91], [92] and particle or Kalman filters [89], [93], [94] to detect lane markings. Then, postprocessing methods are leveraged to filter out misdetections and classify lane points to output the final lane detection results [95]. However, in general, they are prone to effectiveness issues due to road scene variations, e.g., changing from city scene to highway scene and hard to achieve reasonable accuracy under challenging scenarios without a visual clue.

Recently, deep learning-based segmentation approaches have dominated the lane detection field with more accurate performance [96]. For instance, VPGNet [97] proposes a multitask network for lane marking detection. To better utilize more visual information of lane markings, SCNN [98] applies a novel convolution operation that aggregates diverse dimension information via processing sliced features and then adds them together. In order to accelerate the detection speed, lightweight DNNs have been proposed for real-time applications. For example, self-attention distillation (SAD) [99] adopts an attention distillation mechanism. Besides, other methods, such as sequential prediction and clustering, are also introduced. In [100], a long short-term memory (LSTM) network is presented to face the lane's long line structure issue. Similarly, Fast-Draw [101] predicts the lane's direction at the pixelwise level. In [102], the problem of lane detection is defined as a binary clustering problem. The method proposed in [103] also uses a clustering approach for lane detection. Subsequently, a 3-D form of lane detection [104] is introduced to face the nonflatten ground issue.

3) Localization and Mapping: Localization and mapping are fundamental to autonomous driving. Localization is responsible for finding ego position relative to a map [105]. The mapping constructs multilayer HD maps [106] for path planning. Therefore, the accuracy of localization and mapping affects the feasibility and safety of path planning. Currently, GPS-IMU-based localization methods have been widely utilized in navigation software such as Google maps. However, the accuracy required for urban automated driving cannot be fulfilled by GPS-IMU systems [107].

Currently, systems that use a prebuild HD map are more practical and accurate. There are three main types of HD maps: 1) landmark based; 2) point cloud based; and 3) vision based. Landmarks, such as poles, curbs, signs, and road markers, can be detected with LiDAR [108] or camera [109]. Landmark searching consumes less computation than the point cloud-based approach but fails in scenarios where landmarks are insufficient. The point cloud contains detailed information about the environment with thousands of points from LiDAR [110] or camera [111]. Iterative closest point (ICP) [112] and normal distributions transform (NDT) [113] are two algorithms used in point cloud-based HD map generation. They utilize numerical optimization algorithms to calculate the best match. ICP iteratively selects the closest point to calculate the best match. On the other side, NDT represents the map as a combination of the normal distribution, then uses the maximum-likelihood estimation equation to search match. NDT's computation complexity is less than ICP [114], but it is not as robust as ICP. Vision-based HD maps are another direction recently becoming more and more popular. The computational overhead limits its application in real systems. Several methods for matching maps with the 2-D camera as well as matching 2-D image to the 3-D image are proposed for mapping [115]–[117].

In contrast, SLAM [118] is proposed to build the map and localize the vehicle simultaneously. SLAM can be divided into LiDAR-based SLAM and camera-based SLAM. Among LiDAR-based SLAM algorithms, LOAM [119] can be finished in real time. IMLS-SLAM [120] focuses on reducing accumulated drift by utilizing a scan-to-model matching method. Cartographer [121], a SLAM package from Google, improves performance by using submap and loop closure while supporting both 2-D and 3-D LiDAR. Compared with LiDAR-based SLAM, camera-based SLAM approaches use frame-to-frame matching. There are two types of matching methods: 1) feature based and 2) direct matching. Feature-based methods [122]–[124] extract features and track them to calculate the motion of the camera. Since features are sparse in the image, feature-based methods are also called sparse visual SLAM. Direct matching [125]–[127] is called dense visual SLAM, which adopts original information for matching that is dense in the image, such as color and depth from an RGB-D camera. The inherent properties of feature-based methods lead to its faster speed but tend to fail in textureless environments as well. The dense SLAM solves the issues of the sparse SLAM with higher computation complexity. For situations that lack computation resources, semiDense [128], [129] SLAM methods that only use direct methods are proposed. Besides the above methods, deep learning methods are also utilized in solving feature extraction [130], motion estimation [131], and long-term localization [132].

TABLE IV
COMPARISON OF DIFFERENT COMPUTING HARDWARE FOR AUTONOMOUS DRIVING

Boards	Architecture	Performance	Power Consumption	Cost[1]
NVIDIA DRIVE PX2	GPU	30 TOPS	60W	$15,000
NVIDIA DRIVE AGX	GPU	320 TOPS	300W	$30,000
Texas Instruments TDA3x	DSP	-	30mW in 30fps	$549
Zynq UltraScale+ MPSoC ZCU104	FPGA	14 images/sec/Watt	-	$1,295
Mobileye EyeQ5	ASIC	24 TOPS	10W	$750
Google TPU v3	ASIC	420 TFLOPS	40W	$8 per hour

4) Prediction and Planning: The prediction module evaluates the driving behaviors of the surrounding vehicles and pedestrians for risk assessment [30]. The hidden Markov model (HMM) has been used to predict the target vehicle's future behavior and detect unsafe lane change events [133], [134].

Planning means finding feasible routes on the map from the origin to destination. GPS navigation systems are known as global planners [135] to plan a feasible global route, but it does not guarantee safety. In this context, the local planner is developed [136], which can be divided into three groups: 1) graph-based planners that give the best path to the destination; 2) sampling-based planners that randomly scan the environments and only find a feasible path; and 3) interpolating curve planners that are proposed to smooth the path. A* [137] is a heuristic implementation of Dijkstra that always preferentially searches the path from the origin to the destination (without considering the vehicle's motion control), which causes the planning generated by A* to not always be executed by the vehicle. To remedy this problem, hybrid A* [138] generates a drivable curve between each node instead of a jerky line. Sampling-based planners [139] randomly select nodes for search in the graph, reducing the searching time. Among them, rapidly exploring random tree (RRT) [140] is the most commonly used method for automated vehicles. As an extension of RRT, RRT* [141], [142] tries to search the optimal paths satisfying real-time constraints. How to balance the sampling size and computation efficiency is a big challenge for sampling-based planners. Graph-based planners and sampling-based planners can achieve optimal or suboptimal with jerky paths that can be smoothed with interpolating curve planners.

5) Vehicle Control: Vehicle control connects autonomous driving computing systems and the drive-by-wire system. It adjusts the steering angle and maintains the desired speed to follow the planning module's trajectories. Typically, vehicle control is accomplished by using two controllers: 1) lateral controller and 2) longitudinal controller. Controllers must handle rough and curvy roads, and quickly varying types, such as gravel, loose sand, and mud puddles [143], are not considered by vehicle planners. The output commands are calculated from the vehicle state and the trajectory by control law. There are various control laws, such as fuzzy control [144], [145], PID control [146], [147], Stanley control [143], and model predictive control (MPC) [148]–[150]. PID control creates outputs based on proportional, integral, and derivative teams of inputs. Fuzzy control accepts continuous values between 0 and 1, instead of either 1 or 0, as inputs continuously respond. Stanley control is utilized to follow the reference path by minimizing the heading angle and cross-track error using a nonlinear control law. MPC performs a finite horizon optimization to identify the control command. Since it can handle various constraints and use past and current errors to predict more accurate solutions, MPC has been used to solve hard control problems such as following overtaking trajectories [151]. Controllers derive control laws depending on the vehicle model. Kinematic bicycle models and dynamic bicycle models are most commonly used. In [152], a comparison is present to determine which of these two models is more suitable for MPC in forecast error and computational overhead.

D. Computation Hardware

To support real-time data processing from various sensors, powerful computing hardware is essential to AVs' safety. Currently, plenty of computing hardware with different designs show up on the automobile and computing market. In this section, we will show several representative designs based on graphic processor unit (GPU), digital signal processor (DSP), field programmable gate arrays (FPGA), and application-specific integrated circuit (ASIC). The comparisons of GPU, DSP, FPGA, and ASIC in terms of architecture, performance, power consumption, and cost are shown in Table IV.

NVIDIA DRIVE AGX is the newest solution from NVIDIA unveiled at CES 2018 [38]. NVIDIA DRIVE AGX is the world's most powerful System on Chip (SoC), and it is ten times more powerful than the NVIDIA Drive PX2 platform. Each DRIVE AGX consists of two Xavier cores. Each Xavier has a custom 8-core CPU and a 512-core Volta GPU. DRIVE AGX is capable of 320 trillion operations per second (TOPS) of processing performance.

Zynq UltraScale+ MPSoC ZCU104 is an automotive-grade product from Xilinx [153]. It is an FPGA-based device designed for autonomous driving. It includes 64-b quad-core ARM Corte-A53 and dual-core ARM Cortex-R5. This scalable solution claims to deliver the right performance/watt with safety and security [154]. When running CNN tasks, it achieves 14 images/s/W, which outperforms the Tesla K40 GPU (4 images/s/W). Also, for object tracking tasks, it reaches 60 fps in a live 1080p video stream.

Texas Instruments' TDA provides a DSP-based solution for autonomous driving. A TDA3x SoC consists of two C66× floating-point VLIW DSP cores with vision AccelerationPac. Furthermore, each TDA3× SoC has dual Arm Cortex-M4

image processors. The vision accelerator is designed to accelerate the process functions on images. Compared with an ARM Cortex-15 CPU, TDA3× SoC provides an eightfold acceleration on computer vision tasks with less power consumption [155].

MobileEye EyeQ5 is the leading ASIC-based solution to support fully autonomous (level 5) vehicles [156]. EyeQ5 is designed based on 7-nm FinFET semiconductor technology, and it provides 24Tops computation capability with 10-W power budget. TPU is Google's AI accelerator ASIC mainly for neural network and machine learning [157]. TPU v3 is the newest release, which provides 420 TFLOPS computation for a single board.

E. Storage

The data captured by an AV are proliferating, typically generating between 20 and 40 TB per day, per vehicle [158]. The data include cameras (20–40 MB), as well as sonar (10–100 KB), radar (10–100 KB), and LiDAR (10–70 MB) [159], [160]. Storing data securely and efficiently can accelerate overall system performance. Take object detection for example: the history data could contribute to the improvement of detection precision using machine learning algorithms. Map generation can also benefit from the stored data in updating traffic and road conditions appropriately. Additionally, the sensor data can be utilized to ensure public safety and predict and prevent crime. The biggest challenge is to ensure that sensors collect the right data, and it is processed immediately, stored securely, and transferred to other technologies in the chain, such as the roadside unit (RSU), cloud data center, and even third-party users [161]. More importantly, creating hierarchical storage and workflow that enables smooth data accessing and computing is still an open question for the future development of AVs.

In [162], a computational storage system called HydraSpace is proposed to tackle the storage issue for autonomous driving vehicles. HydraSpace is designed with multilayered storage architecture and practical compression algorithms to manage the sensor pipe data. OpenVDAP is a full-stack edge-based data analytic platform for connected and AVs (CAVs) [161]. It envisions for the future four types of CAVs applications, including autonomous driving, in-vehicle infotainment, real-time diagnostics, and third-party applications, such as traffic information collector and SafeShareRide [163]. The hierarchical design of the storage system called driving data integrator (DDI) is proposed in OpenVDAP to provide sensor-aware and application-aware data storage and processing [161].

F. Real-Time Operating Systems

According to the automation level definitions from the SAE [164], the automation of vehicles increases from level 2 to level 5, and the level 5 requires full automation of the vehicle, which means the vehicle can drive under any environment without the help from the human. To make the vehicle run in a safe mode, how to precept the environment and make decisions in real time becomes a big challenge. That is why real-time OSs become a hot topic in the design and implementation of autonomous driving systems.

RTOS is widely used in the embedded system of ECUs to control the vehicle's throttle, brake, etc. *QNX* and *VxWorks* are two representative commercialized RTOS widely used in the automotive industry. The *QNX* kernel contains only CPU scheduling, interprocess communication, interrupt redirection, and timers. Everything else runs as a user process, including a unique process known as "proc," which performs process creation and memory management by operating in conjunction with the microkernel [165]. *VxWorks* is designed for embedded systems requiring real time, deterministic performance and, in many cases, safety and security certification [166]. *VxWorks* supports multiple architectures, including Intel, POWER, and ARM. *VxWorks* also uses real-time kernels for mission-critical applications subject to real-time constraints, which guarantees a response within predefined time constraints.

RTLinux is a microkernel-based OS that supports hard real time [167]. The scheduler of *RTLinux* allows full preemption. Compared with using a low-preempt patch in *Linux*, *RTLinux* allows preemption for the whole Linux system. *RTLinux* makes it possible to run real-time critical tasks and interprets them together with the *Linux* [168].

NVIDIA DRIVE OS is a foundational software stack from NVIDIA that consists of an embedded RTOS, hypervisor, NVIDIA CUDA libraries, NVIDIA Tensor RT, etc., which is needed for the acceleration of machine learning algorithms [169].

G. Middleware Systems

Robotic systems, such as AV systems, often involve multiple services, with many dependencies. Middleware is required to facilitate communications between different autonomous driving services.

Most existing autonomous driving solutions utilize the *ROS* [32]. Specifically, *ROS* is a communication middleware that facilitates communications between different modules of an AV system. *ROS* supports four communication methods: 1) topic; 2) service; 3) action; and 4) parameter. *ROS2* is a promising type of middleware developed to make communications more efficient, reliable, and secure [170]. However, most of the packages and tools for sensor data process are still currently based on *ROS*.

The *Autoware Foundation* is a nonprofit organization supporting open-source projects enabling self-driving mobility [171]. *Autoware.AI* is developed based on *ROS*, and it is the world's first "all-in-one" open-source software for autonomous driving technology. *Apollo Cyber* [172] is another open-source middleware developed by Baidu. *Apollo* aims to accelerate the development, testing, and deployment of AVs. *Apollo Cyber* is a high-performance runtime framework that is greatly optimized for high concurrency, low latency, and high throughput in autonomous driving.

In the traditional automobile society, the runtime environment layer in Automotive Open System Architecture (*AutoSAR*) [173] can be seen as middleware. Many companies develop their middleware to support *AutoSAR*. However, there are few independent open-source middlewares nowadays because it is a commercial vehicle company's core technology.

Auto companies prefer to provide middleware as a component of a complete set of autonomous driving solutions.

H. Vehicular Communication

In addition to obtaining information from the onboard sensors, the recent proliferation in communication mechanisms, e.g., DSRC, C-V2X, and 5G, has enabled autonomous driving vehicles to obtain information from other vehicles, infrastructures such as traffic lights and RSU, as well as pedestrians.

1) LTE/4G/5G: Long-term evolution (LTE) is a transitional product in the transition from 3G to 4G [174], which provides downlink peak rates of 300 Mb/s and uplink peak rates of 75 Mb/s. The fourth-generation communications (4G) comply with 1 Gb/s for stationary reception and 100 Mb/s for mobile. As the next-generation mobile communication, U.S. users that experienced the fastest average 5G download speed reached 494.7 Mb/s on Verizon, 17.7 times faster than that of 4G. From Verizon's early report, the latency of 5G is less than 30 ms, 23 ms faster than average 4G metrics. However, we cannot deny that 5G still has the following challenges: complex system, high costs, and poor obstacle avoidance capabilities.

2) DSRC: DSRC [1] is a type of V2X communication protocol, which is specially designed for connected vehicles. DSRC is based on the IEEE 802.11p standard, and its working frequency is 5.9 GHz. Fifteen message types are defined in the SAE J2735 standard [175], which covers information, such as the vehicle's position, map information, emergence warning, etc., [1]. Limited by the available bandwidth, DSRC messages have small size and low frequency. However, DSRC provides reliable communication, even when the vehicle is driving 120 miles per hour.

3) C-V2X: C-V2X combines the traditional V2X network with the cellular network, which delivers mature network assistance and commercial services of 4G/5G into autonomous driving. Like DSRC, the working frequency of C-V2X is also the primary common spectrum, 5.9 GHz [2]. Different from the CSMA-CA in DSRC, C-V2X has no contention overheads by using semipersistent transmission with relative energy-based selection. Besides, the performance of C-V2X can be seamlessly improved with the upgrade of the cellular network. Generally, C-V2X is more suitable for V2X scenarios where cellular networks are widely deployed.

I. Security and Privacy

With the increasing degree of vehicle electronification and the reliance on a wide variety of technologies, such as sensing and machine learning, the security of AVs has risen from the hardware damage of traditional vehicles to comprehensive security with multidomain knowledge. Here, we introduce several security problems strongly associated with AVs with the current attacking methods and standard coping methods. In addition to the security and privacy issues mentioned as follows, AVs systems should also take care of many other security issues in other domains, such as patching vulnerabilities of hardware or software systems and detecting intrusions [176].

1) Sensing Security: As the eye of AVs, the security of sensors is nearly essential. Typically, jamming attacks and spoofing attacks are two primary attacks for various sensors [44], [45]. For example, the spoofing attack generates an interference signal, resulting in a fake obstacle captured by the vehicle [46]. Besides, GPS also encounters spoofed attacks [47]. Therefore, protection mechanisms are expected for sensor security. Randomized signals and redundant sensors are usually used by these signal-reflection sensors [177], [178], including LiDAR and radar. The GPS can check signal characteristics [179] and authenticate data sources [180] to prevent attacks. Also, sensing data fusion is an effective mechanism.

2) Communication Security: Communication security includes two aspects: 1) internal communication and 2) outside communication. Currently, internal communication, such as CAN, LIN, and FlexRay, has faced severe security threats [181]–[183]. The cryptography is a frequently used technology to keep the transmitted data confidential, integrated, and authenticated [184]. However, the usage of cryptography is limited by the high computational cost for these resource-constrained ECUs. Therefore, another attempt is to use the gateway to prevent unallowed access [185]. The outside communication has been studied in VANETs with V2V, V2R, and V2X communications [186]–[188]. Cryptography is the primary tool. A trusted key distribution and management is built in most approaches, and vehicles use assigned keys to authenticate vehicles and data.

3) Data Security: Data security refers to preventing data leakage from the perspectives of transmission and storage. The former has been discussed in communication security, where various cryptography approaches are proposed to protect data in different scenarios [189], [190]. The cryptography is also a significant technology of securing data storage, such as an encrypted database [191] and file system [192]. Besides, access control technology [193] protects stored data from another view, widely used in modern OSs. An access control framework [194] has been proposed for AVs to protect in-vehicle data in real-time data and historical data, with different access control models.

4) Control Security: With vehicles' electronification, users could open the door through an electronic key and control their vehicles through an application or voice. However, this also leads to new attack surfaces with various attack methods, such as jamming attacks, replay attacks, relay attacks, etc., [44]. For example, attackers could capture the communication between key and door and replay it to open the door [195]. Also, for those voice control supported vehicles, the attackers could successfully control the vehicle by using voices that humans cannot hear [196]. Parts of these attacks could be classified into sensing security, communication security, or data security, which can be addressed by corresponding protection mechanisms.

5) Privacy: AVs heavily rely on the data of the surrounding environment, which typically contains user privacy. For example, by recognizing buildings in cameras, attackers can learn the vehicle location [197], or an attacker can obtain the location directly from GPS data. Thus, the most straightforward

but the most difficult solution is to prevent data from being obtained by an attacker, such as access control [193], [194] and data encryption [44]. However, AVs will inevitably utilize location-based services. Except for the leak of current location, attackers could learn the home address from the vehicle trajectory [198]. Thus, data desensitization is necessary to protect privacy, including anonymization and differential privacy [199].

V. Challenges and Discussion

From the review of the current key technologies of the computing system for autonomous driving, we find that there are still many challenges and open issues for the research and development of L4 or L5 autonomous driving vehicles. In this section, we summarize 12 remaining challenges and discuss the challenges with our visions for autonomous driving.

A. Artificial Intelligence for AVs

Most of the AV's services (e.g., environmental perception and path planning) are carried out by artificial intelligence-based approaches. As the focus of the automotive industry gradually shifts to series production, the main challenge is how to apply machine learning algorithms to mass-produced AVs for real-world applications. Here, we list three main challenges in artificial intelligence (AI) for AVs.

1) Standardization of Safety Issue: One of the main challenges is that machine learning algorithms are unstable in terms of performance. For example, even a small change to camera images (such as cropping and variations in lighting conditions) may cause the ADAS system to fail in object detection and segmentation [27], [98], [200]. However, the automotive safety standard of ISO 26262 [201] was defined without taking deep learning into consideration because the ISO 26262 was published before the boom of AI, leading to the absence of proper ways to standardize the safety issue when incorporating AI for AVs [202].

2) Infeasibility of Scalable Training: To achieve high performance, machine learning models used on AVs need to be trained on representative datasets under all application scenarios, which bring challenges in training time-sensitive models based on Petabytes of data. In this case, collaborative training [203], model compression technologies [204]–[209], and lightweight machine learning algorithms [210]–[212] were proposed in recent years. Besides, getting accurate annotations of every pedestrian, vehicle, lane, and other objects are necessary for model training using supervised learning approaches, which becomes a significant bottleneck [213].

3) Infeasibility of Complete Testing: It is infeasible to test machine learning models used on AVs thoroughly. One reason is that machine learning learns from large amounts of data and stores the model in a complex set of weighted feature combinations, which is not intuitive or difficult to conduct thorough testing [214]. In addition, previous work pointed out that to verify the catastrophic failure rate, around 10^9 hours (billion hours) of vehicle operation test should be carried out [215] and the test needs be repeated many times to achieve statistical significance [202].

B. Multisensors Data Synchronization

Data on the autonomous driving vehicle have various sources: its sensors, other vehicle sensors, RSU, and even social media. One big challenge to handle a variety of data sources is how to synchronize them.

For example, a camera usually produces 30–60 frames/s, while LiDAR's point cloud data frequency is 10 Hz. For applications such as 3-D object detection, which requires camera frames and point cloud at the same time, should the storage system do synchronization beforehand or let the application developer do it? This issue becomes more challenging, considering that the timestamp's accuracy from different sensors falls into different granularities. For example, considering the vehicles that use a network time protocol (NTP) for time synchronization, the timestamp difference can be as long as 100 ms [216], [217]. For some sensors with a built-in GNSS antenna, the time accuracy goes to the nanosecond level. In contrast, other sensors get a timestamp from the host machine's system time when accuracy is at the millisecond level. Since the accuracy of time synchronization is expected to affect the vehicle control's safety, handling the sensor data with different frequency and timestamp accuracy is still an open question.

C. Failure Detection and Diagnostics

Today's AVs are equipped with multiple sensors, including LiDARs, radars, and GPS [46]. Although we can take advantage of these sensors in terms of providing a robust and complete description of the surrounding area, some open problems related to the failure detection are waiting to be solved. Here, we list and discuss four failure detection challenges.

1) *Definition of Sensor Failure:* There is no standard, agreed-upon universal definition or standards, to define the scenario of sensor failures [41]. However, we must propose and categorize the standard of sensor failures to support failure detection by applying proper methods.
2) *Sensor Failure:* More importantly, there is no comprehensive and reliable study on sensor failure detection, which is extremely dangerous since most of the self-driving applications are relying on the data produced by these sensors [42]. If some sensors encountered a failure, collisions and environmental catastrophes might happen.
3) *Sensor Data Failure:* In the real application scenario, even when the sensors themselves are working correctly, the generated data may still not reflect the actual scenario and report the wrong information to people [218]. For instance, the camera is blocked by unknown objects, such as leaves or mud, or the radar deviates from its original fixed position due to wind force. In this context, sensor data failure detection is very challenging.
4) *Algorithm Failure:* In challenging scenarios with severe occlusion and extreme lighting conditions, such as night, rainy days, and snowy days, deploying and executing state-of-the-art algorithms cannot guarantee output the ideal results [219]. For example, lane markings usually fail to be detected at night by algorithms that find it difficult to explicitly utilize prior information, such

as rigidity and smoothness of lanes [220]. However, humans can easily infer their positions and fill in the occluded part of the context. Therefore, how to develop advanced algorithms to further improve detection accuracy is still a big challenge.

For a complex system with rich sensors and hardware devices, failures could happen everywhere. How to tackle the failure and diagnose the issue becomes a big issue. One example is the diagnose of lane controller systems from Google [221]. The idea is to determine the root cause of malfunctions based on comparing the actual steering corrections applied to those predicted by the virtual dynamics module.

D. How to Deal With Normal–Abnormal?

Normal–abnormal represents normal scenarios in daily life that are abnormal in the autonomous driving dataset. Typically, there are three cases of normal–abnormal: 1) adverse weather; 2) emergency maneuvers; and 3) work zones.

1) Adverse Weather: One of the most critical issues in the development of AVs is the poor performance under adverse weather conditions, such as rain, snow, fog, and hail, because the equipped sensors (e.g., LiDAR, radar, camera, and GPS) might be significantly affected by the extreme weather. The work of [222] characterized the effect of rainfall on millimeter-wave (mm-wave) radar and proved that under heavy rainfall conditions, the detection range of the millimeter-wave radar can be reduced by as much as 45%. Filgueira *et al.* [223] pointed out that as the rain intensity increases, the detected LiDAR intensity will attenuate. At the same time, Bernardin *et al.* [224] proposed a methodology to quantitatively estimate the loss of visual performance due to rainfall. Most importantly, experimental results show that compared to training in narrow cases and scenarios, using various data sets to train object detection networks may not necessarily improve the performance of these networks [225]. However, there is currently no research to provide a systematic and unified method to reduce the impact of weather on various sensors used in AVs. Therefore, there is an urgent need for novel deep learning networks that have sufficient capabilities to cope with safe autonomous driving under severe weather conditions.

2) Emergency Maneuvers: In emergency situations, such as a road collapse, braking failure, a tire blowout, or suddenly seeing a previously "invisible" pedestrian, the maneuvering of the AVs may need to reach its operating limit to avoid collisions. However, these collision avoidance actions usually conflict with stabilization actions aimed at preventing the vehicle from losing control, and in the end, they may cause collision accidents. In this context, some research has been done to guarantee safe driving for AVs in emergent situations. For example, Hilgert *et al.* [226] proposed a path planning method for emergency maneuvers based on elastic bands. Funke *et al.* [227] is proposed to determine the minimum distance at which obstacles cannot be avoided at a given speed. Guo *et al.* [228] discussed dynamic control design for automated driving, with particular emphasis on coordinated steering and braking control in emergency avoidance. Nevertheless, how an AV safely responds to different classes of emergencies with onboard sensors is still an open problem.

3) Work Zone: Work zone recognition is another challenge for an autonomous driving system to overcome. For most drivers, the work zone means congestion and delay of the driving plan. Many projects have been launched to reduce and eliminate work zone injuries and deaths for construction workers and motorists. "Workzonesafety.org" summarizes recent years of work zone crashes and supplies training programs to increase public awareness of the importance of work-zone safety. Seo *et al.* [229] proposed a machine learning-based method to improve the recognition of work zone signs. Developers from Kratos Defense and Security Solutions [230] present an autonomous truck which safely passes a work zone. Their system relied on V2V communications to connect the self-driving vehicle with a leader vehicle. The self-driving vehicle accepted navigation data from the leader vehicle to travel along its route while keeping a predefined distance. Until now, the work zone is still a threat to drivers and workers' safety but has not attracted too much attention to autonomous driving researchers. There are still significant gaps in this research field, waiting for researchers to explore and tackle critical problems.

E. Cyberattack Protection

Attacks and defenses are always opposites, and absolute security does not exist. The emerging CAVs face many security challenges, such as reply attacks to simulate a vehicle's electronic key and spoof attacks to make vehicle detour [44], [195]. With the integration of new sensors, devices, technologies, infrastructures, and applications, the attack surface of CAVs is further expanded.

Many attacks focus on one part of the CAVs system and could be protected by the method of fusing several other views. For example, a cheated roadblock detected by radars could be corrected by camera data. Thus, how to build such a system to protect CAVs, systematically, is the first challenge for the CAVs system. The protection system is expected to detect potential attacks, evaluate the system security status, and recover from attacks.

Besides, some novel attack methods should be attended. Recently, some attacks have been proposed to trick these algorithms [231]. For example, a photograph instead of a human to pass the face recognition or a note-sized photograph posted on the forehead makes machine learning algorithms fail to detect faces [232]. Thus, how to defend the attacks on machine learning algorithms is a challenge for CAVs systems.

Furthermore, some new technologies could be used to enhance the security of the CAVs system. With the development of quantum computing technology, the existing cryptography standards cannot ensure protected data, communication, and systems. Thus, designing postquantum cryptography [233] and architecture is a promising topic for CAVs and infrastructure in ITS.

Also, we noticed that the hardware-assistant trusted execution environment [234] could improve the system security, which provides an isolated and trusted execution environment (TEE) for applications. However, it has limited physical

memory size, and execution performance will drop sharply as the total memory usage increases. Therefore, how to split the system components and make critical parts in the TEE with high security is still a challenge in design and implementation.

F. Vehicle Operating System

The vehicle OS is expected to abstract the hardware resources for higher layer middleware and autonomous driving applications. In the vehicle OS development, one of the biggest challenges is the compatibility with the vehicle's embedded system. Take autoware as an example: although it is a full-stack solution for the vehicle OS that provides a rich set of self-driving modules composed of sensing, computing, and actuation capabilities, the usage of it is still limited to several commercial vehicles with a small set of supportable sensors [235]. On a modern automobile, as many as 70 ECUs are installed for various subsystems, and they are communicated via CAN bus. For the sake of system security and commercial interests, most of the vehicles' CAN protocol is not open sourced, which is the main obstacle for developing a unified vehicle OS.

AUTOSAR is a standardization initiative of leading automotive manufacturers and suppliers founded in the autumn of 2003 [173]. AUTOSAR is promising in narrowing the gap for developing an open-source vehicle OS. However, most automobile companies are relatively conservative to open source their vehicle OSs, restricting the availability of AUTOSAR to the general research and education community. There is still a strong demand for a robust, open-source vehicle OS for AVs.

G. Energy Consumption

With rich sensors and powerful computing devices implemented on the vehicle, energy consumption becomes a big issue. Take the NVIDIA Drive PX Pegasus as an example: it consumes 320 INT8 TOPS of AI computational power with a 500-W budget. If we added external devices, such as sensors, communication antennas, storage, battery, etc., the total energy consumption would be larger than 1000 W [38]. Besides, if a duplicate system is installed for the autonomous driving applications' reliability, the total power dissipation could go up to almost 2000 W.

How to handle such a tremendous amount of power dissipation is not only a problem for the battery management system but it is also a problem for the heat dissipation system. What makes this issue more severe is the size limitation and autogrid requirements from the vehicle's perspective. How to make the computing system of the autonomous driving vehicle become energy efficient is still an open challenge. E2M tackles this problem by proposing as an energy-efficient middleware for the management and scheduling deep learning applications to save energy for the computing device [49]. However, according to the profiling results, most of the energy is consumed by vehicles' motors. Energy-efficient autonomous driving requires the co-design in battery cells, energy management systems, and AV computing systems.

H. Cost

In the United States, the average cost to build a traditional nonluxury vehicle is roughly $30 000, and for an AV, the total cost is around $250 000 [236]. AVs need an abundance of hardware equipment to support their normal functions. Additional hardware equipments required for AVs include, but are not limited to, the communication device, computing equipment, drive-by-wire system, extra power supply, various sensors, cameras, LiDAR, and radar. In addition, to ensure AV's reliability and safety, a backup of these hardware devices may be necessary [237]. For example, if the main battery fails, the vehicle should have a backup power source to support computing systems to move the vehicle.

The cost of building an AV is already very high, not to mention the maintenance cost of an AV, e.g., diagnostics and repair. High maintenance costs lead to declining consumer demand and undesirable profitability for the vehicle manufacturers. Companies, such as Ford and GM, have already cut their low-profit production lines to save costs [238], [239].

Indeed, the cost of computing systems for AVs currently in the research and development stage is very high. However, we hope that with the maturity of the technologies and the emergence of some alternative solutions, the price will ultimately drop to a level that individuals can afford. Take battery packs of electric vehicles (EVs) as an example: when the first mass-market EVs were introduced in 2010, their battery packs were estimated at $1000 per kilowatt-hour (kWh). However, Tesla's Model 3 battery pack costs $190 kWh, and General Motors' 2017 Chevrolet Bolt battery pack is estimated to cost $205 kWh. In six years, the price per kilowatt-hour has dropped by more than 70% [240]. Also, Waymo claims to have successfully reduced the experimental version of high-end LiDAR to approximately $7500. Besides, Tesla, which uses only radar instead of LiDAR, says its AV equipment is around $8000 [236]. In addition to the reduction of hardware costs, we believe that the optimization of computing software in an AV can also help reduce the cost to a great extent.

I. How to Benefit From Smart Infrastructure?

Smart infrastructure combines sensors, computing platforms, and communication devices with the physical traffic infrastructure [241]. It is expected to enable the AVs to achieve more efficient and reliable perception and decision making. Typically, AVs could benefit from smart infrastructure in three aspects as follows.

1) *Service Provider:* It is struggling for an AV to find a parking space in the parking lot. By deploying sensors such as RFID on the smart infrastructure, parking services can be handled quickly [242]. As the infrastructure becomes a provider for parking service, it is possible to schedule service requests to achieve the maximum usage. Meanwhile, AVs can reduce the time and computation for searching services.
2) *Traffic Information Sharing:* Traffic information is essential for safe driving. The lack of traffic information causes traffic congestion or even accidents. Roadside units (RSUs) are implemented to provide traffic information to passing vehicles through V2X communications. Besides, RSUs are also used to surveil road situations using various onboard sensors, such as cameras and LiDARs [243]. The collected data are used

for various tasks, including weather warning, map updating, road events detection, and making up blind spots of AVs.

3) *Task Offloading:* Various algorithms are running on vehicles for safe driving. Handling all workloads in real time requires a tremendous amount of computation and power, infeasible on a battery-powered vehicle [244]. Therefore, offloading heavy computation workloads to the infrastructure is proposed to accelerate the computation and save energy. However, to perform feasible offloading, the offloading framework must offload computations to the infrastructure while ensuring timing predictability [245]. Therefore, how to schedule the order of offloading workloads is still a challenge to benefit from the smart infrastructure.

J. Dealing With Human Drivers

According to NHTSA data collected from all 50 states and the District of Columbia, 37 461 lives were lost on U.S. roads in 2016, and 94% of crashes were associated with "a human choice or error" [246]. Although autonomous driving is proposed to replace human drivers with computers/machines for safety purposes, human driving vehicles will never disappear. How to enable computers/machines in AVs to interact with a human driver becomes a big challenge [247].

Compared with a human driver, machines are generally more suited for tasks, such as vehicle control and multisensor data processing. In contrast, the human driver maintains an advantage in perception and sensing the environment [248]. One of the fundamental reasons is that the machine cannot think like a human. The current machine learning-based approaches cannot handle situations that are not captured in the training data set. For example, in driving automation from SAE, one of the critical differences between level 2 and level 3/4/5 is whether the vehicle can make decisions, such as overtaking or lane changing by itself [249]. In some instances, interacting with other human drivers becomes a big challenge because human drivers can make mistakes or violate traffic rules.

Many works focus on getting a more accurate speed and control predictions of the surrounding vehicles to handle the machine–human interaction [133], [250]. Deep reinforcement learning shows promising performance in complex scenarios requiring interaction with other vehicles [28], [100]. However, they are either simulation based or demonstration in limited scenarios. Another promising direction to tackle machine–human interaction is through V2X communications. Compared with predicting other vehicles' behavior, it is more accurate to communicate safety information [251].

K. Experimental Platform

The deployment of autonomous driving algorithms or prototypes requires complex tests and evaluations in a real environment, which makes the experimental platform become one of the fundamental parts of conducting research and development. However, building and maintaining an autonomous driving vehicle are enormous: the cost of a real autonomous driving vehicle could attain $250 000; and maintaining the vehicle requires parking, insurance, and automaintenance. Let alone the laws and regulations to consider for field testing.

Given these limitations and problems, lots of autonomous driving simulators and open-source prototypes are proposed for research and development purposes. dSPACE provides an end-to-end simulation environment for sensor data processing and scenario-based testing with RTMaps and VEOS [252]. The automated driving toolbox is Mathwork's software, which provides algorithms and tools for designing, simulating, and testing ADAS and autonomous driving systems [253]. AVL DriveCube is a hardware-in-the-loop driving simulator designed for real vehicles with simulated environments [254]. In addition to these commercialized products, there are also open-source projects, such as CARLA and Gezabo, for urban driving or robotics simulations [255], [256].

Another promising direction is to develop affordable research and development of autonomous driving platforms. Several experiment platforms are quite successful for indoor or low-speed scenarios. HydraOne is an open-source experimental platform for indoor autonomous driving, and it provides full-stack programmability for autonomous driving algorithms developers and system developers [257]. DragonFly is another example that supports self-driving with a speed of fewer than 40 miles per hour and a price of less than $40 000 [258].

L. Physical Worlds Coupling

Autonomous driving is a typical cyber–physical system [259], where the computing systems and the physical world have to work closely and smoothly. With a human driver, the feeling of a driver is easily coupled with the vehicle control actions. For example, if the driver does not like the abrupt stop, he or she can step on the brake gradually. In autonomous driving, the control algorithm will determine the speed of braking and accelerating. We envision that different human feelings, coupled with complex traffic environment, bring an unprecedented challenge to the vehicle control in autonomous driving. Take the turning left as an example: how fast should the drive-by-wire system turn 90°? An ideal vehicle control algorithm of turning left should consider many factors, such as the friction of road surface, vehicle's current speed, weather conditions, and the movement range, as well as human comfortableness, if possible. Cross-layer design and optimization among perception, control, vehicle dynamics, and drive-by-wire systems might be a promising direction [260].

VI. Conclusion

The recent proliferation of computing and communication technologies, such as machine learning, hardware acceleration, DSRC, C-V2X, and 5G has dramatically promoted autonomous driving vehicles. Complex computing systems are designed to leverage the sensors and computation devices to understand the traffic environments correctly in real time. However, the early developed AVs' fatalities arise from time to time, which reveals the big gap between the current computing system and the expected robust system for level-4/level-5

full autonomous driving. In this article, we presented the state-of-the-art computing systems for autonomous driving, including seven performance metrics, nine key technologies, and 12 challenges and opportunities to realize the vision of autonomous driving. We hope this article will bring these challenges to the attention of both the computing and automotive communities.

REFERENCES

[1] J. B. Kenney, "Dedicated short-range communications (DSRC) standards in the United States," *Proc. IEEE*, vol. 99, no. 7, pp. 1162–1182, Jul. 2011.

[2] "The case for cellular V2X for safety and cooperative driving," GA Assoc., Atlanta, GA, USA, White Paper, Nov. 2016.

[3] (2018). *Global Autonomous Driving Market Outlook*. [Online]. Available: https://www.prnewswire.com/news-releases/global-autonomous-driving-market-outlook-2018-300624588.html

[4] M. Birdsall, "Google and ITE: The road ahead for self-driving cars," *Inst. Transp. Eng. J.*, vol. 84, no. 5, p. 36, 2014.

[5] (2020). *Autonomous Driving*. [Online]. Available: https://mbrdna.com/teams/autonomous-driving/

[6] (2020). *Mobility Goes Smart and Individual: Audi at CES 2020*. [Online]. Available: https://www.audi-mediacenter.com/en/press-releases/mobility-goes-smart-and-individualaudi-at-ces-2020-12455

[7] H. Somerville, P. Lienert, and A. Sage, *Uber's Use of Fewer Safety Sensors Prompts Questions After Arizona Crash*, Reuters, London, U.K., 2018.

[8] (2020). *5 Top Autonomous Vehicle Companies to Watch in 2020*. [Online]. Available: https://www.mes-insights.com/5-top-autonomous-vehicle-companies-to-watch-in-2020-a-910825/

[9] *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*. SAE Int., Warrendale, PA, USA, 2018.

[10] (2017). *Continental ARS4-A 77GHz Radar*. [Online]. Available: https://www.systemplus.fr/reverse-costing-reports/continental-ars4-a-77ghz-radar/

[11] (2019). *Velodyne LiDAR Products*. [Online]. Available: https://velodynelidar.com/products.html

[12] (2019). *BeiDou Navigation Satellite System*. [Online]. Available: http://en.beidou.gov.cn/

[13] (2019). *GPS: The Global Positioning System*. [Online]. Available: https://www.gps.gov/

[14] S. Liu, L. Li, J. Tang, S. Wu, and J.-L. Gaudiot, "Creating autonomous vehicle systems," *Synth. Lectures Comput. Sci.*, vol. 6, no. 1, p. 186, 2017.

[15] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018. [Online]. Available: arXiv:1804.02767.

[16] D. Neven, B. De Brabandere, S. Georgoulis, M. Proesmans, and L. Van Gool, "Towards end-to-end lane detection: An instance segmentation approach," in *Proc. IEEE Intell. Veh. Symp. (IV)*, 2018, pp. 286–291.

[17] C. Stachniss, J. J. Leonard, and S. Thrun, "Simultaneous localization and mapping," in *Springer Handbook of Robotics*. Berlin, Germany: Springer, 2016, pp. 1153–1176.

[18] J. Ziegler *et al.*, "Making bertha drive—An autonomous journey on a historic route," *IEEE Intell. Transp. Syst. Mag.*, vol. 6, no. 2, pp. 8–20, Apr. 2014.

[19] (2020). *List of Self-Driving Car Fatalities*. [Online]. Available: https://en.wikipedia.org/wiki/List_of_self-driving_car_fatalities#cite_note-20

[20] S. Kato, E. Takeuchi, Y. Ishiguro, Y. Ninomiya, K. Takeda, and T. Hamada, "An open approach to autonomous vehicles," *IEEE Micro*, vol. 35, no. 6, pp. 60–68, Nov./Dec. 2015.

[21] C. Urmson *et al.*, "Autonomous driving in urban environments: Boss and the urban challenge," *J. Field Robot.*, vol. 25, no. 8, pp. 425–466, 2008.

[22] J. Levinson *et al.*, "Towards fully autonomous driving: Systems and algorithms," in *Proc. IEEE Intell. Veh. Symp. (IV)*, 2011, pp. 163–168.

[23] A. Broggi *et al.*, "Extensive tests of autonomous driving technologies," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 3, pp. 1403–1415, Sep. 2013.

[24] U. Muller, J. Ben, E. Cosatto, B. Flepp, and Y. L. Cun, "Off-road obstacle avoidance through end-to-end learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2006, pp. 739–746.

[25] C. Chen, A. Seff, A. Kornhauser, and J. Xiao, "DeepDriving: Learning affordance for direct perception in autonomous driving," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 2722–2730.

[26] M. Bojarski *et al.*, "End to end learning for self-driving cars," 2016. [Online]. Available: arXiv:1604.07316.

[27] H. Xu, Y. Gao, F. Yu, and T. Darrell, "End-to-end learning of driving models from large-scale video datasets," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2174–2182.

[28] A. E. Sallab, M. Abdou, E. Perot, and S. Yogamani, "Deep reinforcement learning framework for autonomous driving," *Electron. Imag.*, vol. 2017, no. 19, pp. 70–76, 2017.

[29] A. Kendall *et al.*, "Learning to drive in a day," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2019, pp. 8248–8254.

[30] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, "A survey of autonomous driving: Common practices and emerging technologies," *IEEE Access*, vol. 8, pp. 58443–58469, 2020.

[31] R. E. Schantz and D. C. Schmidt, "Middleware," in *Encyclopedia of Software Engineering*. Boca Raton, FL, USA: CRC Press, 2002.

[32] M. Quigley *et al.*, "ROS: An open-source robot operating system," in *Proc. ICRA Workshop Open Source Softw.*, vol. 3. Kobe, Japan, 2009, p. 5.

[33] P. Hank, S. Müller, O. Vermesan, and J. Van Den Keybus, "Automotive Ethernet: In-vehicle networking and smart mobility," in *Proc. Design Autom. Test Europe Conf. Exhibit. (DATE)*, 2013, pp. 1735–1739.

[34] (2020). *Ensuring American Leadership in Automated Vehicle Technologies: Automated Vehicles 4.0*. [Online]. Available: https://www.transportation.gov/av/4

[35] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, 2010.

[36] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 580–587.

[37] (2019). *Autonomous Cars—Big Problem: The Energy Consumption of Edge Processing Reduces a Car's Mileage With Up to 30%*. [Online]. Available: https://medium.com/@teraki/energy-consumption-required-by-edge-computing-reduces-a-autonomous-cars-mileage-with-up-to-30-46b6764ea1b7

[38] (2018). *Meet NVIDIA Xavier: A New Brain for Self-Driving, AI, and AR Cars*. [Online]. Available: https://www.slashgear.com/meet-nvidia-xavier-a-new-brain-for-self-driving-ai-and-ar-cars-07513987/

[39] Steve LeVine. (2017). *What It Really Costs to Turn a Car Into a Self-Driving Vehicle*. [Online]. Available: https://qz.com/924212/what-it-really-costs-to-turn-a-car-into-a-self-driving-vehicle/

[40] (2017). *Google's Waymo Invests in LiDAR Technology, Cuts Costs by 90 Percent*. [Online]. Available: https://arstechnica.com/cars/2017/01/googles-waymo-invests-in-lidar-technology-cuts-costs-by-90-percent/

[41] G. Sabaliauskaite, L. S. Liew, and J. Cui, "Integrating autonomous vehicle safety and security analysis using STPA method and the six-step model," *Int. J. Adv. Security*, vol. 11, nos. 1–2, pp. 160–169, 2018.

[42] A. Orrick, M. McDermott, D. M. Barnett, E. L. Nelson, and G. N. Williams, "Failure detection in an autonomous underwater vehicle," in *Proc. IEEE Symp. Auton. Underwater Veh. Technol. (AUV)*, 1994, pp. 377–382.

[43] S. Taghavi and W. Shi, "EdgeMask: An edge-based privacy preserving service for video data sharing," in *Proc. 3rd Workshop Security Privacy Edge Comput. (EdgeSP)*, 2020, pp. 1–6.

[44] K. Ren, Q. Wang, C. Wang, Z. Qin, and X. Lin, "The security of autonomous driving: Threats, defenses, and future directions," *Proc. IEEE*, vol. 108, no. 2, pp. 357–372, Feb. 2020.

[45] E. Yagdereli, C. Gemci, and A. Z. Aktas, "A study on cyber-security of autonomous and unmanned vehicles," *J. Defense Model. Simulat.*, vol. 12, no. 4, pp. 369–381, 2015. [Online]. Available: https://doi.org/10.1177/1548512915575803

[46] C. Yan, W. Xu, and J. Liu, "Can you trust autonomous vehicles: Contactless attacks against sensors of self-driving vehicle," presented at the DEF CON, vol. 24, 2016, p. 109.

[47] K. C. Zeng *et al.*, "All your GPS are belong to us: Towards stealthy manipulation of road navigation systems," in *Proc. 27th USENIX Security Symp. (USENIX Security)*, Aug. 2018, pp. 1527–1544. [Online]. Available: https://www.usenix.org/conference/usenixsecurity18/presentation/zeng

[48] (2020). *The CAR Lab*. [Online]. Available: https://www.thecarlab.org/

[49] L. Liu, J. Chen, M. Brocanelli, and W. Shi, "E2M: An energy-efficient middleware for computer vision applications on autonomous mobile robots," in *Proc. 4th ACM/IEEE Symp. Edge Comput.*, 2019, pp. 59–73.

[50] Z. Zhang, S. Liu, G. Tsai, H. Hu, C.-C. Chu, and F. Zheng, "PIRVS: An advanced visual-inertial SLAM system with flexible sensor fusion and

hardware co-design," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2018, pp. 1–7.
[51] J. Levinson, M. Montemerlo, and S. Thrun, "Map-based precision vehicle localization in urban environments," *Robot. Sci. Syst.*, vol. 4, pp. 121–128, Jun. 2008.
[52] K. Konolige, "Large-scale map-making," in *Proc. AAAI*, 2004, pp. 457–463.
[53] N. Fairfield and C. Urmson, "Traffic light mapping and detection," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2011, pp. 5421–5426.
[54] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, 2013.
[55] J. H. Yoon, C.-R. Lee, M.-H. Yang, and K.-J. Yoon, "Online multi-object tracking via structural constraint event aggregation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 1392–1400.
[56] A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez, and J. Garcia-Rodriguez, "A review on deep learning techniques applied to semantic segmentation," 2017. [Online]. Available: arXiv:1704.06857.
[57] A. Tampuu, M. Semikin, N. Muhammad, D. Fishman, and T. Matiisen, "A survey of end-to-end driving: Architectures and training methods," 2020. [Online]. Available: arXiv:2003.06404.
[58] F. Yu *et al.*, "BDD100K: A diverse driving dataset for heterogeneous multitask learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 2636–2645.
[59] J. Binas, D. Neil, S.-C. Liu, and T. Delbruck, "DDD17: End-to-end DAVIS driving dataset," 2017. [Online]. Available: arXiv:1711.01458.
[60] *Makesense.ai*. Accessed: Jan. 11, 2020. [Online]. Available: https://www.makesense.ai/
[61] *labelImg*. Accessed: Jan. 11, 2020. [Online]. Available: https://github.com/tzutalin/labelImg
[62] *VGG Image Annotator*. Accessed: Jan. 11, 2020. [Online]. Available: https://gitlab.com/vgg/via
[63] *LabelMe*. Accessed: Jan. 11, 2020. [Online]. Available: http://labelme.csail.mit.edu/Release3.0/
[64] *Scalable*. Accessed: Jan. 11, 2020. [Online]. Available: https://scalabel.ai/
[65] *RectLabel*. Accessed: Jan. 11, 2020. [Online]. Available: https://rectlabel.com/
[66] C. Michaelis *et al.*, "Benchmarking robustness in object detection: Autonomous driving when winter is coming," 2019. [Online]. Available: arXiv:1907.07484.
[67] Z. Zou, Z. Shi, Y. Guo, and J. Ye, "Object detection in 20 years: A survey," 2019. [Online]. Available: arXiv:1905.05055.
[68] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 1, 2001, pp. 511–518.
[69] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 1, 2005, pp. 886–893.
[70] P. Felzenszwalb, D. McAllester, and D. Ramanan, "A discriminatively trained, multiscale, deformable part model," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2008, pp. 1–8.
[71] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 1440–1448.
[72] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 91–99.
[73] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2961–2969.
[74] W. Liu *et al.*, "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 21–37.
[75] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg, "DSSD: Deconvolutional single shot detector," 2017. [Online]. Available: arXiv:1701.06659.
[76] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 779–788.
[77] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 7263–7271.
[78] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Region-based convolutional networks for accurate object detection and segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 1, pp. 142–158, Jan. 2016.
[79] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," 2020. [Online]. Available: arXiv:2004.10934.
[80] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2980–2988.
[81] A. Borkar, M. Hayes, and M. T. Smith, "A novel lane detection system with efficient ground truth generation," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 1, pp. 365–374, Mar. 2012.
[82] H. Deusch, J. Wiest, S. Reuter, M. Szczot, M. Konrad, and K. Dietmayer, "A random finite set approach to multiple lane detection," in *Proc. 15th Int. IEEE Conf. Intell. Transp. Syst.*, 2012, pp. 270–275.
[83] J. Hur, S.-N. Kang, and S.-W. Seo, "Multi-lane detection in urban driving environments using conditional random fields," in *Proc. IEEE Intell. Veh. Symp. (IV)*, 2013, pp. 1297–1302.
[84] H. Jung, J. Min, and J. Kim, "An efficient lane detection algorithm for lane departure detection," in *Proc. IEEE Intell. Veh. Symp. (IV)*, 2013, pp. 976–981.
[85] H. Tan, Y. Zhou, Y. Zhu, D. Yao, and K. Li, "A novel curve lane detection based on improved river flow and RANSA," in *Proc. 17th Int. IEEE Conf. Intell. Transp. Syst. (ITSC)*, 2014, pp. 133–138.
[86] P.-C. Wu, C.-Y. Chang, and C. H. Lin, "Lane-mark extraction for automobiles under complex conditions," *Pattern Recognit.*, vol. 47, no. 8, pp. 2756–2767, 2014.
[87] K.-Y. Chiu and S.-F. Lin, "Lane detection using color-based segmentation," in *Proc. IEEE Intell. Veh. Symp.*, 2005, pp. 706–711.
[88] H. Loose, U. Franke, and C. Stiller, "Kalman particle filter for lane recognition on rural roads," in *Proc. IEEE Intell. Veh. Symp.*, 2009, pp. 60–65.
[89] Z. Teng, J.-H. Kim, and D.-J. Kang, "Real-time lane detection by using multiple CUES," in *Proc. IEEE ICCAS*, 2010, pp. 2334–2337.
[90] A. López, J. Serrat, C. Canero, F. Lumbreras, and T. Graf, "Robust lane markings detection and road geometry computation," *Int. J. Autom. Technol.*, vol. 11, no. 3, pp. 395–407, 2010.
[91] G. Liu, F. Wörgötter, and I. Markelić, "Combining statistical hough transform and particle filter for robust lane detection and tracking," in *Proc. IEEE Intell. Veh. Symp.*, 2010, pp. 993–997.
[92] S. Zhou, Y. Jiang, J. Xi, J. Gong, G. Xiong, and H. Chen, "A novel lane detection based on geometrical model and Gabor filter," in *Proc. IEEE Intell. Veh. Symp.*, 2010, pp. 59–64.
[93] Z. Kim, "Robust lane detection and tracking in challenging scenarios," *IEEE Trans. Intell. Transp. Syst.*, vol. 9, no. 1, pp. 16–26, Mar. 2008.
[94] R. Danescu and S. Nedevschi, "Probabilistic lane tracking in difficult road scenarios using stereovision," *IEEE Trans. Intell. Transp. Syst.*, vol. 10, no. 2, pp. 272–282, Jun. 2009.
[95] A. B. Hillel, R. Lerner, D. Levi, and G. Raz, "Recent progress in road and lane detection: A survey," *Mach. Vis. Appl.*, vol. 25, no. 3, pp. 727–745, 2014.
[96] R. Gopalan, T. Hong, M. Shneier, and R. Chellappa, "A learning approach towards detection and tracking of lane markings," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 3, pp. 1088–1098, Sep. 2012.
[97] S. Lee *et al.*, "VPGNET: Vanishing point guided network for lane and road marking detection and recognition," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 1947–1955.
[98] X. Pan, J. Shi, P. Luo, X. Wang, and X. Tang, "Spatial as deep: Spatial CNN for traffic scene understanding," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 7276–7283.
[99] Y. Hou, Z. Ma, C. Liu, and C. C. Loy, "Learning lightweight lane detection CNNs by self attention distillation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 1013–1021.
[100] J. Li, X. Mei, D. Prokhorov, and D. Tao, "Deep neural network for structural prediction and lane detection in traffic scene," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 3, pp. 690–703, Mar. 2016.
[101] J. Philion, "FastDraw: Addressing the long tail of lane detection by adapting a sequential prediction network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 11582–11591.
[102] Y.-C. Hsu, Z. Xu, Z. Kira, and J. Huang, "Learning to cluster for proposal-free instance segmentation," in *Proc. IEEE Int. Joint Conf. Neural Netw. (IJCNN)*, 2018, pp. 1–8.
[103] Y. Hou, "Agnostic lane detection," 2019. [Online]. Available: arXiv:1905.03704.
[104] N. Garnett, R. Cohen, T. Pe'er, R. Lahav, and D. Levi, "3D-LaneNet: End-to-end 3D multiple lane detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 2921–2930.
[105] S. Kuutti, S. Fallah, K. Katsaros, M. Dianati, F. Mccullough, and A. Mouzakitis, "A survey of the state-of-the-art localisation techniques and their potentials for autonomous vehicle applications," *IEEE Internet Things J.*, vol. 5, no. 2, pp. 829–846, Apr. 2019.
[106] K. Jiang, D. Yang, C. Liu, T. Zhang, and Z. Xiao, "A flexible multi-layer map model designed for lane-level route planning in

autonomous vehicles," *Engineering*, vol. 5, no. 2, pp. 305–318, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2095809918300328

[107] C. Urmson *et al.*, "High speed navigation of unrehearsed terrain: Red team technology for grand challenge 2004," Robot. Inst., Carnegie Mellon Univ., Pittsburgh, PA, USA, Rep. CMU-RI-04-37, Jan. 2004.

[108] A. Hata and D. Wolf, "Road marking detection using LiDAR reflective intensity data and its application to vehicle localization," in *Proc. ITSC*, Oct. 2014, pp. 584–589.

[109] J. Suhr, J. Jang, D. Min, and H. Jung, "Sensor fusion-based low-cost vehicle localization system for complex urban environments," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 5, pp. 1078–1086, May 2017.

[110] S. M. I. Zolanvari *et al.*, "DublinCity: Annotated LiDAR point cloud and its applications," 2019. [Online]. Available: arXiv:1909.03613.

[111] B. Su, J. Ma, Y. Peng, and M. Sheng, "Algorithm for RGBD point cloud denoising and simplification based on *k*-means clustering," *J. Syst. Simulat.*, vol. 28, no. 10, pp. 2329–2334, Oct. 2016.

[112] P. J. Besl and H. D. McKay, "A method for registration of 3-D shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 239–256, Feb. 1992.

[113] P. Biber and W. Strasser, "The normal distributions transform: A new approach to laser scan matching," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, vol. 3, Nov. 2003, pp. 2743–2748.

[114] M. Magnusson, A. Nuchter, C. Lorken, A. Lilienthal, and J. Hertzberg, "Evaluation of 3D registration reliability and speed—A comparison of ICP and NDT," in *Proc. ICRA*, May 2009, pp. 3907–3912.

[115] R. Wolcott and R. Eustice, "Visual localization within LiDAR maps for automated urban driving," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, Oct. 2014, pp. 176–183.

[116] Q.-H. Pham, M. A. Uy, B.-S. Hua, D. T. Nguyen, G. Roig, and S.-K. Yeung, "LCD: Learned cross-domain descriptors for 2D-3D matching," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, Apr. 2020, pp. 11856–11864.

[117] C. Mcmanus, W. Churchill, A. Napier, B. Davis, and P. Newman, "Distraciton suppression for vision-based pose estimation at city scales," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2013, pp. 3762–3769.

[118] G. Bresson, Z. Alsayed, L. Yu, and S. Glaser, "Simultaneous localization and mapping: A survey of current trends in autonomous driving," *IEEE Trans. Intell. Veh.*, vol. 2, no. 3, pp. 194–220, Sep. 2017.

[119] J. Zhang and S. Singh, "LOAM: Lidar odometry and mapping in real-time," in *Proc. Robot. Sci. Syst.*, vol. 2, no. 9, 2014.

[120] J.-E. Deschaud, "IMLS-SLAM: Scan-to-model matching based on 3D data," in *Proc. ICRA*, May 2018, pp. 2480–2485.

[121] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2D LiDAR SLAM," in *Proc. ICRA*, May 2016, pp. 1271–1278.

[122] R. Mur-Artal and J. D. Tardos, "ORB-SLAM2: An open-source SLAM system for monocular, stereo and RGB-D cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017.

[123] S. Sumikura, M. Shibuya, and K. Sakurada, "OpenVSLAM: A versatile visual SLAM framework," in *Proc. ACM Multimedia*, Oct. 2019, pp. 2292–2295.

[124] D. Schlegel, M. Colosi, and G. Grisetti, "ProSLAM: Graph SLAM from a programmer's perspective," in *Proc. ICRA*, May 2018, pp. 1–9.

[125] R. Newcombe, S. Lovegrove, and A. Davison, "DTAM: Dense tracking and mapping in real-time," in *Proc. ICCV*, Nov. 2011, pp. 2320–2327.

[126] F. Endres, J. Hess, J. Sturm, D. Cremers, and W. Burgard, "3-D mapping with an RGB-D camera," *IEEE Trans. Robot.*, vol. 30, no. 1, pp. 177–187, Feb. 2014.

[127] M. Labbé and F. Michaud, "RTAB-map as an open-source LiDAR and visual simultaneous localization and mapping library for large-scale and long-term online operation," *J. Field Robot.*, vol. 36, no. 2, pp. 416–446, Oct. 2018.

[128] J. Engel, T. Schoeps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *Proc. ECCV*, vol. 8690, Sep. 2014, pp. 1–16.

[129] H. Rebecq, T. Horstschaefer, G. Gallego, and D. Scaramuzza, "EVO: A geometric approach to event-based 6-DOF parallel tracking and mapping in real-time," *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 593–600, Apr. 2017.

[130] S. Yang, Y. Song, M. Kaess, and S. Scherer, "Pop-up SLAM: Semantic monocular plane SLAM for low-texture environments," in *Proc. IROS*, Oct. 2016, pp. 1222–1229.

[131] K.-N. Lianos, J. L. Schönberger, M. Pollefeys, and T. Sattler, "VSO: Visual semantic odometry," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 234–250.

[132] A. Gawel, C. Don, R. Siegwart, J. Nieto, and C. Cadena, "X-View: Graph-based semantic multi-view localization," *IEEE Robot. Autom. Lett.*, vol. 3, no. 3, pp. 1687–1694, Jul. 2018.

[133] X. Geng, H. Liang, B. Yu, P. Zhao, L. He, and R. Huang, "A scenario-adaptive driving behavior prediction approach to urban autonomous driving," *Appl. Sci.*, vol. 7, no. 4, p. 426, 2017.

[134] S. Yamazaki *et al.*, "Integrating driving behavior and traffic context through signal symbolization," in *Proc. IEEE Intell. Veh. Symp. (IV)*, 2016, pp. 642–647.

[135] H. Bast *et al.*, "Route planning in transportation networks," Apr. 2015. [Online]. Available: arxiv.abs/1504.05140.

[136] D. G. Bautista, J. Pérez, V. Milanes, and F. Nashashibi, "A review of motion planning techniques for automated vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 4, pp. 1135–1145, Apr. 2016.

[137] P. Hart, N. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *Intell. SIGART Bull.*, vol. 37, pp. 28–29, Dec. 1972.

[138] M. Montemerlo *et al.*, "JUNIOR: The stanford entry in the urban challenge," *J. Field Robot.*, vol. 25, pp. 569–597, Sep. 2008.

[139] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. J. Robot. Res.*, vol. 30, pp. 846–894, Jun. 2011.

[140] S. LaValle and J. Kuffner, "Randomized kinodynamic planning," *Int. J. Robot. Res.*, vol. 20, no. 5, pp. 473–479, Jan. 1999.

[141] J.-H. Ryu, D. Ogay, S. Bulavintsev, H. Kim, and J.-S. Park, "Development and experiences of an autonomous vehicle for high-speed navigation and obstacle avoidance," in *Frontiers of Intelligent Autonomous Systems*. Heidelberg, Germany: Springer, Jan. 2013, pp. 105–116.

[142] S. Karaman, M. Walter, A. Perez, E. Frazzoli, and S. Teller, "Anytime motion planning using the RRT," in *Proc. ICRA*, Jun. 2011, pp. 1478–1483.

[143] G. Hoffmann, C. Tomlin, M. Montemerlo, and S. Thrun, "Autonomous automobile trajectory tracking for off-road driving: Controller design, experimental validation and racing," in *Proc. ACC*, Aug. 2007, pp. 2296–2301.

[144] S. Allou, Z. Youcef, and B. Aissa, "Fuzzy logic controller for autonomous vehicle path tracking," in *Proc. STA*, Dec. 2017, pp. 328–333.

[145] I. Emmanuel, "Fuzzy logic-based control for autonomous vehicle: A survey," *Int. J. Educ. Manag. Eng.*, vol. 7, pp. 41–49, Mar. 2017.

[146] A. Baskaran, A. Talebpour, and S. Bhattacharyya, "End-to-end drive by-wire PID lateral control of an autonomous vehicle," in *Proc. Future Technol. Conf.*, Jan. 2020, pp. 365–376.

[147] M. Prexl, N. Zunhammer, and U. Walter, "Motion prediction for tele-operating autonomous vehicles using a PID control model," in *Proc. ANZCC*, Nov. 2019, pp. 133–138.

[148] W. Choi, H.-S. Nam, B. Kim, and C. Ahn, "Model predictive control for evasive steering of autonomous vehicle," *Int. J. Autom. Technol.*, vol. 20, pp. 1252–1258, Feb. 2020.

[149] J. Yu, X. Guo, X. Pei, Z. Chen, M. Zhu, and B. Gong, "Robust model predictive control for path tracking of autonomous vehicle," SAE Technical Paper, SAE Int., Warrendale, PA, USA, Apr. 2019.

[150] H. Jafarzadeh and C. Fleming, "Learning model predictive control for connected autonomous vehicles," in *Proc. CDC*, Aug. 2019, pp. 2336–2343.

[151] S. Dixit *et al.*, "Trajectory planning for autonomous high-speed overtaking using MPC with terminal set constraints," in *Proc. ITSC*, Nov. 2018, pp. 1061–1068.

[152] J. Kong, M. Pfeiffer, G. Schildbach, and F. Borrelli, "Kinematic and dynamic vehicle models for autonomous driving control design," in *Proc. Intell. Veh. Symp.*, Jun. 2015, pp. 1094–1099.

[153] (2020). *Enabling Next Generation ADAS and AD Systems*. [Online]. Available: https://www.xilinx.com/products/silicon-devices/soc/xa-zynq-ultrascale-mpsoc.html

[154] (2020). *Zynq UltraScale + MPSoC ZCU104 Evaluation Kit*. [Online]. Available: https://www.xilinx.com/products/boards-and-kits/zcu104.html

[155] *Texas Instruments TDA*. Accessed: Dec. 28, 2018. [Online]. Available: http://www.ti.com/processors/automotive-processors/tdax-adas-socs/overview.html

[156] (2020). *The Evolution of EyeQ*. [Online]. Available: https://www.mobileye.com/our-technology/evolution-eyeq-chip/

[157] (2019). *An In-Depth Look at Google's First Tensor Processing Unit (TPU)*. [Online]. Available: https://cloud.google.com/blog/products/gcp/an-in-depth-look-at-googles-first-tensor-processing-unit-tpu

[158] *Flood of Data Will Get Generated in Autonomous Cars*. Accessed: Feb. 18, 2020. [Online]. Available: https://autotechreview.com/features/flood-of-data-will-get-generated-in-autonomous-cars

[159] *Data Storage Is the Key to Autonomous Vehicles—Future*. Accessed: Dec. 30, 2019. [Online]. Available: https://iotnowtransport.com/2019/02/12/71015-data-storage-key-autonomous-vehicles-future/

[160] *The Basics of LiDAR—Light Detection and Ranging—Remote Sensing*. Accessed: Feb. 18, 2020. [Online]. Available: https://www.neonscience.org/lidar-basics

[161] Q. Zhang *et al.*, "OpenVDAP: An open vehicular data analytics platform for CAVs," in *Proc. IEEE 38th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, 2018, pp. 1310–1320.

[162] L. Liu, R. Wang, and W. Shi, "HydraSpace: Computational data storage for autonomous vehicles," in *Proc. IEEE Collaborative Internet Comput. Vis. Track (CIC)*, Dec. 2020.

[163] L. Liu, X. Zhang, M. Qiao, and W. Shi, "SafeShareRide: Edge-based attack detection in ridesharing services," in *Proc. IEEE/ACM Symp. Edge Comput. (SEC)*, 2018, pp. 17–29.

[164] (2019). *Self-Driving Car*. [Online]. Available: https://en.wikipedia.org/wiki/Self-driving_car

[165] D. Hildebrand, "An architectural overview of QNX," in *Proc. USENIX Workshop Microkernels Kernel Architectures*, 1992, pp. 113–126.

[166] *VxWorks*. Accessed: Dec. 28, 2018. [Online]. Available: https://www.windriver.com/products/vxworks/

[167] V. Yodaiken *et al.*, "The RTLinux manifesto," in *Proc. 5th Linux Expo.*, 1999.

[168] H. Sato and T. Yakoh, "A real-time communication mechanism for RTLinux," in *Proc. 21st Century Technol. 26th Annu. Conf. IEEE Ind. Electron. Soc. (IECON) IEEE Int. Conf. Ind. Electron. Control Instrum.*, vol. 4, 2000, pp. 2437–2442.

[169] (2020). *NVIDIA DRIVE—Software*. [Online]. Available: https://developer.nvidia.com/drive/drive-software

[170] (2020). *ROS 2 Documentation*. [Online]. Available: https://index.ros.org/doc/ros2/

[171] (2020). *Welcome to the Autoware Foundation*. [Online]. Available: https://www.autoware.org/

[172] Baidu. *Apollo Cyber*. [Online]. Available: https://github.com/ApolloAuto/apollo/tree/master/cyber

[173] *AUTOSAR*. [Online]. Available: https://www.autosar.org/

[174] L. Liu, Y. Yao, R. Wang, B. Wu, and W. Shi, "EquiNox: A road-side edge computing experimental platform for CAVs," in *Proc. IEEE Int. Conf. Connected Auton. Driving (MetroCAD)*, 2020, pp. 41–42.

[175] *Dedicated Short Range Communications (DSRC) Message Set Dictionary*, SAE Int., Warrendale, PA, USA, Nov. 2009.

[176] C. Warrender, S. Forrest, and B. Pearlmutter, "Detecting intrusions using system calls: Alternative data models," in *Proc. IEEE Symp. Security Privacy*, 1999, pp. 133–145.

[177] H. Shin, D. Kim, Y. Kwon, and Y. Kim, "Illusion and dazzle: Adversarial optical channel exploits against lidars for automotive applications," in *Proc. Int. Conf. Cryptogr. Hardw. Embedded Syst.*, 2017, pp. 445–467.

[178] J. Petit, B. Stottelaar, M. Feiri, and F. Kargl, "Remote attacks on automated vehicles sensors: Experiments on camera and LiDAR," in *Proc. Black Hat Europe*, vol. 11, 2015, p. 2015.

[179] A. Konovaltsev, M. Cuntz, C. Hättich, and M. Meurer, "Autonomous spoofing detection and mitigation in a GNSS receiver with an adaptive antenna array," in *Proc. ION GNSS+*, Sep. 2013, pp. 2937–2948. [Online]. Available: https://elib.dlr.de/86230/

[180] B. W. O'Hanlon, M. L. Psiaki, J. A. Bhatti, D. P. Shepard, and T. E. Humphreys, "Real-time GPS spoofing detection via correlation of encrypted signals," *Navigation*, vol. 60, no. 4, pp. 267–278, 2013. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/navi.44

[181] K. Koscher *et al.*, "Experimental security analysis of a modern automobile," in *Proc. IEEE Symp. Security Privacy*, 2010, pp. 447–462.

[182] J. M. Ernst and A. J. Michaels, "Lin bus security analysis," in *Proc. 44th Annu. Conf. IEEE Ind. Electron. Soc. (IECON)*, 2018, pp. 2085–2090.

[183] D. K. Nilsson, U. E. Larson, F. Picasso, and E. Jonsson, "A first simulation of attacks in the automotive network communications protocol flexray," in *Proc. Int. Workshop Comput. Intell. Security Inf. Syst. (CISIS)*, 2009, pp. 84–91.

[184] D. R. Stinson and M. Paterson, *Cryptography: Theory and Practice*. Hoboken, NJ, USA: CRC Press, 2018.

[185] J. H. Kim, S. Seo, N. Hai, B. M. Cheon, Y. S. Lee, and J. W. Jeon, "Gateway framework for in-vehicle networks based on CAN, FlexRay, and Ethernet," *IEEE Trans. Veh. Technol.*, vol. 64, no. 10, pp. 4472–4486, Oct. 2015.

[186] A. Nanda, D. Puthal, J. J. P. C. Rodrigues, and S. A. Kozlov, "Internet of autonomous vehicles communications security: Overview, issues, and directions," *IEEE Wireless Commun.*, vol. 26, no. 4, pp. 60–65, Aug. 2019.

[187] F. Qu, Z. Wu, F.-Y. Wang, and W. Cho, "A security and privacy review of VANETs," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 6, pp. 2985–2996, Dec. 2015.

[188] I. Ali, A. Hassan, and F. Li, "Authentication and privacy schemes for vehicular ad hoc networks (VANETs): A survey," *Veh. Commun.*, vol. 16, pp. 45–61, Apr. 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S221420961830319X

[189] H. Zhong, L. Pan, Q. Zhang, and J. Cui, "A new message authentication scheme for multiple devices in intelligent connected vehicles based on edge computing," *IEEE Access*, vol. 7, pp. 108211–108222, 2019.

[190] S. Garg *et al.*, "Edge computing-based security framework for big data analytics in VANETs," *IEEE Netw.*, vol. 33, no. 2, pp. 72–81, Mar./Apr. 2019.

[191] R. A. Popa, C. M. S. Redfield, N. Zeldovich, and H. Balakrishnan, "CryptDB: Protecting confidentiality with encrypted query processing," in *Proc. 23rd ACM Symp. Oper. Syst. Principles (SOSP)*, 2011, pp. 85–100. [Online]. Available: https://doi.org/10.1145/2043556.2043566

[192] M. Blaze, "A cryptographic file system for UNIX," in *Proc. 1st ACM Conf. Comput. Commun. Security (CCS)*, 1993, pp. 9–16. [Online]. Available: https://doi.org/10.1145/168588.168590

[193] R. S. Sandhu and P. Samarati, "Access control: Principle and practice," *IEEE Commun. Mag.*, vol. 32, no. 9, pp. 40–48, Sep. 1994. [Online]. Available: https://doi.org/10.1109/35.312842

[194] Q. Zhang, H. Zhong, J. Cui, L. Ren, and W. Shi, "AC4AV: A flexible and dynamic access control framework for connected and autonomous vehicles," *IEEE Internet Things J.*, early access, Aug. 17, 2020, doi: 10.1109/JIOT.2020.3016961.

[195] S. Kamkar, "Drive it like you hacked it: New attacks and tools to wirelessly steal cars," presentation at the DEFCON, 2015.

[196] G. Zhang, C. Yan, X. Ji, T. Zhang, T. Zhang, and W. Xu, "DolphinAttack: Inaudible voice commands," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security (CCS)*, 2017, pp. 103–117. [Online]. Available: https://doi.org/10.1145/3133956.3134052

[197] Z. Xiong, W. Li, Q. Han, and Z. Cai, "Privacy-preserving auto-driving: A GAN-based approach to protect vehicular camera data," in *Proc. IEEE Int. Conf. Data Min. (ICDM)*, 2019, pp. 668–677.

[198] H. Li, D. Ma, B. Medjahed, Y. S. Kim, and P. Mitra, "Analyzing and preventing data privacy leakage in connected vehicle services," *SAE Int. J. Adv. Current Practices*, vol. 1, pp. 1035–1045, Apr. 2019. [Online]. Available: https://doi.org/10.4271/2019-01-0478

[199] F. Martinelli, F. Mercaldo, A. Orlando, V. Nardone, A. Santone, and A. K. Sangaiah, "Human behavior characterization for driving style recognition in vehicle system," *Comput. Elect. Eng.*, vol. 83, May 2020, Art. no. 102504. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0045790617329531

[200] M. Ghafoorian, C. Nugteren, N. Baka, O. Booij, and M. Hofmann, "EL-GAN: Embedding loss driven generative adversarial networks for lane detection," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 256–272.

[201] Q. V. E. Hommes, "Review and assessment of the ISO 26262 draft road vehicle-functional safety," SAE, Warrendale, PA, USA, Rep. 2012-01-0025, 2012.

[202] Q. Rao and J. Frtunikj, "Deep learning for self-driving cars: Chances and challenges," in *Proc. IEEE/ACM 1st Int. Workshop Softw. Eng. AI Auton. Syst. (SEFAIAS)*, 2018, pp. 35–38.

[203] S. Lu, Y. Yao, and W. Shi, "Collaborative learning on the edges: A case study on connected vehicles," in *Proc. 2nd USENIX Workshop Hot Topics Edge Comput. (HotEdge)*, Renton, WA, USA, Jul. 2019. [Online]. Available: https://www.usenix.org/conference/hotedge19/presentation/lu

[204] M. Courbariaux, Y. Bengio, and J.-P. B. David, "Training deep neural networks with binary weights during propagations," 2015. [Online]. Available: arXiv:1511.00363.

[205] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 1135–1143.

[206] E. L. Denton, W. Zaremba, J. Bruna, Y. LeCun, and R. Fergus, "Exploiting linear structure within convolutional networks for efficient evaluation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 1269–1277.

[207] M. Denil, B. Shakibi, L. Dinh, M. A. Ranzato, and N. de Freitas, "Predicting parameters in deep learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 2148–2156.

[208] B. B. Sau and V. N. Balasubramanian, "Deep model compression: Distilling knowledge from noisy teachers," 2016. [Online]. Available: arXiv:1610.09650.

[209] P. Luo *et al.*, "Face model compression by distilling knowledge from neurons," in *Proc. AAAI*, 2016, pp. 3560–3566.

[210] A. G. Howard *et al.*, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017. [Online]. Available: arXiv:1704.04861.

[211] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," 2017. [Online]. Available: arXiv:1610.02357.

[212] F. N. Iandola *et al.*, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and < 0.5 mb model size," 2016. [Online]. Available: arXiv:1602.07360.

[213] F. Yu *et al.*, "BDD100K: A diverse driving video database with scalable annotation tooling," 2018. [Online]. Available: arXiv:1805.04687.

[214] P. Koopman and M. Wagner, "Challenges in autonomous vehicle testing and validation," *SAE Int. J. Transp. Safety*, vol. 4, no. 1, pp. 15–24, 2016.

[215] R. W. Butler and G. B. Finelli, "The infeasibility of experimental quantification of life-critical software reliability," in *Proc. Conf. Softw. Citical Syst.*, 1991, pp. 66–76.

[216] D. Mills, "Network time protocol (version 3) specification, implementation," IETF, RFC 1305, 1992.

[217] (2020). *GPS Accuracy*. [Online]. Available: https://www.gps.gov/systems/gps/performance/accuracy/

[218] Y. Tian, K. Pei, S. Jana, and B. Ray, "DeepTest: Automated testing of deep-neural-network-driven autonomous cars," in *Proc. 40th Int. Conf. Softw. Eng.*, 2018, pp. 303–314.

[219] L. Tang, Y. Shi, Q. He, A. W. Sadek, and C. Qiao, "Performance test of autonomous vehicle LiDAR sensors under different weather conditions," *Transp. Res. Rec.*, vol. 2674, no. 1, pp. 319–329, 2020.

[220] Y. Tamai, T. Hasegawa, and S. Ozawa, "The ego-lane detection under rainy condition," in *Proc. 3rd World Congr. Intell. Transp. Syst.*, 1996, p. 3258.

[221] J.-W. Lee and B. B. Litkouhi, "System diagnosis in autonomous driving," U.S. Patent 9 168 924, Oct. 27, 2015,

[222] S. Zang, M. Ding, D. Smith, P. Tyler, T. Rakotoarivelo, and M. A. Kaafar, "The impact of adverse weather conditions on autonomous vehicles: How rain, snow, fog, and hail affect the performance of a self-driving car," *IEEE Veh. Technol. Mag.*, vol. 14, no. 2, pp. 103–111, Jun. 2019.

[223] A. Filgueira, H. González-Jorge, S. Lagüela, L. Díaz-Vilariño, and P. Arias, "Quantifying the influence of rain in LiDAR performance," *Measurement*, vol. 95, pp. 143–148, Jan. 2017.

[224] F. Bernardin *et al.*, "Measuring the effect of the rainfall on the windshield in terms of visual performance," *Accident Anal. Prevent.*, vol. 63, pp. 83–88, Feb. 2014.

[225] M. Hnewa and H. Radha, "Object detection under rainy conditions for autonomous vehicles," 2020. [Online]. Available: arXiv:2006.16471.

[226] J. Hilgert, K. Hirsch, T. Bertram, and M. Hiller, "Emergency path planning for autonomous vehicles using elastic band theory," in *Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatron. (AIM)*, vol. 2, 2003, pp. 1390–1395.

[227] J. Funke, M. Brown, S. M. Erlien, and J. C. Gerdes, "Collision avoidance and stabilization for autonomous vehicles in emergency scenarios," *IEEE Trans. Control Syst. Technol.*, vol. 25, no. 4, pp. 1204–1216, Jul. 2017.

[228] J. Guo, P. Hu, and R. Wang, "Nonlinear coordinated steering and braking control of vision-based autonomous vehicles in emergency obstacle avoidance," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 11, pp. 3230–3240, Nov. 2016.

[229] Y.-W. Seo, J. Lee, W. Zhang, and D. Wettergreen, "Recognition of highway workzones for reliable autonomous driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 708–718, Apr. 2015.

[230] A. Barwacz. (2019). *Self-Driving Work Zone Vehicles Enhance Safety*. [Online]. Available: https://www.gpsworld.com/self-driving-work-zone-vehicles-enhance-safety/

[231] Y. Cao *et al.*, "Adversarial sensor attack on LiDAR-based perception in autonomous driving," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security (CCS)*, 2019, pp. 2267–2281. [Online]. Available: https://doi.org/10.1145/3319535.3339815

[232] S. Komkov and A. Petiushko, "AdvHat: Real-world adversarial attack on ArcFace face ID system," 2019. [Online]. Available: arxiv.abs/1908.08705.

[233] D. J. Bernstein and T. Lange, "Post-quantum cryptography," *Nature*, vol. 549, pp. 188–194, Jan. 2017.

[234] Z. Ning, F. Zhang, W. Shi, and W. Shi, "Position paper: Challenges towards securing hardware-assisted execution environments," in *Proc. Hardw. Architect. Support Security Privacy (HASP)*, 2017, pp. 1–8. [Online]. Available: https://doi.org/10.1145/3092627.3092633

[235] S. Kato *et al.*, "Autoware on board: Enabling autonomous vehicles with embedded systems," in *Proc. ACM/IEEE 9th Int. Conf. Cyber Phys. Syst. (ICCPS)*, 2018, pp. 287–296.

[236] S. LeVine. (Mar. 2017). *What It Really Costs to Turn a Car Into a Self-Driving Vehicle*. [Online]. Available: https://qz.com/924212/what-it-really-costs-to-turn-a-car-into-a-self-driving-vehicle/

[237] D. Sedgwick. (Feb. 2017). *When Driverless Cars Call for Backup*. [Online]. Available: https://www.autonews.com/article/20170218/OEM10/302209969/when-driverless-cars-call-for-backup

[238] G. Guilford. (Apr. 2018). *Ford Can Only Afford to Give Up on Cars Because of American Protectionism*. [Online]. Available: https://qz.com/1262815/fords-move-to-stop-making-cars-was-enabled-by-american-protectionism/

[239] A. Luft. (Jul. 2020). *The Chevrolet Sonic's Days Are Numbered*. [Online]. Available: https://gmauthority.com/blog/2020/07/the-chevrolet-sonics-days-are-numbered/

[240] UCSUSA. (Mar. 2018). *Electric Vehicle Batteries: Materials, Cost, Lifespan*. [Online]. Available: https://www.ucsusa.org/resources/ev-batteries

[241] K. Bowers *et al.*, "Smart infrastructure: Getting more from strategic assets," 2016.

[242] Z. Pala and N. Inanc, "Smart parking applications using RFID technology," in *Proc. 1st Annu. RFID Eurasia*, Oct. 2007, pp. 1–3.

[243] A. Al-Dweik, R. Muresan, M. Mayhew, and M. Lieberman, "IoT-based multifunctional scalable real-time enhanced road side unit for intelligent transportation systems," in *Proc. CCECE*, Apr. 2017, pp. 1–6.

[244] L. Lin, X. Liao, H. Jin, and P. Li, "Computation offloading toward edge computing," *Proc. IEEE*, vol. 107, no. 8, pp. 1584–1607, Aug. 2019.

[245] Z. Dong, W. Shi, G. Tong, and K. Yang, "Collaborative autonomous driving: Vision and challenges," in *Proc. Int. Conf. Connected Auton. Driving (MetroCAD)*, 2020, pp. 17–26.

[246] National Highway Traffic Safety Administration *et al.*, "Traffic safety facts: 2007 data: Pedestrians," *Ann. Emerg. Med.*, vol. 53, no. 6, p. 824, 2009.

[247] (2020). *SafeDI Scenario-Based AV Policy Framework—An Overview for Policy-Makers*. [Online]. Available: https://www.weforum.org/whitepapers/safe-drive-initiative-safedi-scenario-based-av-policy-framework-an-overview-for-policy-makers

[248] B. Schoettle, *Sensor Fusion: A Comparison of Sensing Capabilities of Human Drivers and Highly Automated Vehicles*, Univ. Michigan, Ann Arbor, MI, USA, 2017.

[249] S. Liu, L. Liu, J. Tang, B. Yu, Y. Wang, and W. Shi, "Edge computing for autonomous driving: Opportunities and challenges," *Proc. IEEE*, vol. 107, no. 8, pp. 1697–1716, Aug. 2019.

[250] C. Hubmann, M. Becker, D. Althoff, D. Lenz, and C. Stiller, "Decision making for autonomous driving considering interaction and uncertain prediction of surrounding vehicles," in *Proc. IEEE Intell. Veh. Symp. (IV)*, 2017, pp. 1671–1678.

[251] R. Deng, B. Di, and L. Song, "Cooperative collision avoidance for overtaking maneuvers in cellular V2X-based autonomous driving," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 4434–4446, May 2019.

[252] (2020). *Empowering Safe Autonomous Driving*. [Online]. Available: https://www.dspace.com/en/inc/home/applicationfields/our_solutions_for/driver_assistance_systems.cfm

[253] (2020). *Automated Driving Toolbox: Design, Simulate, and Test ADAS and Autonomous Driving Systems*. [Online]. Available: https://www.mathworks.com/products/automated-driving.html

[254] (2020). *AVL DRIVINGCUBE: A New Way to Speed Up the Validation and Approval Process of ADAS/AD Systems*. [Online]. Available: https://www.avl.com/pos-test/-/asset_publisher/gkkFgTqjTyJh/content/avl-drivingcube

[255] N. P. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, vol. 3, 2004, pp. 2149–2154.

[256] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," 2017. [Online]. Available: arXiv:1711.03938.

[257] Y. Wang, L. Liu, X. Zhang, and W. Shi, "HydraOne: An indoor experimental research and education platform for CAVs," in *Proc. 2nd USENIX Workshop Hot Topics Edge Comput. (HotEdge)*, Renton, WA, USA, Jul. 2019. [Online]. Available: https://www.usenix.org/conference/hotedge19/presentation/wang

[258] (2018). *PerceptIn's Self-Driving Vehicles Go on Sale in November for 40 000*. [Online]. Available: https://venturebeat.com/2018/09/12/perceptins-self-driving-vehicles-go-on-sale-in-november-for-40000/

[259] D. Goswami *et al.*, "Challenges in automotive cyber-physical systems design," in *Proc. IEEE Int. Conf. Embedded Comput. Syst. (SAMOS)*, 2012, pp. 346–354.

[260] C. Lv, X. Hu, A. Sangiovanni-Vincentelli, Y. Li, C. M. Martinez, and D. Cao, "Driving-style-based codesign optimization of an automated electric vehicle: A cyber-physical system approach," *IEEE Trans. Ind. Electron.*, vol. 66, no. 4, pp. 2965–2975, Apr. 2019.