Federated Deep Learning Meets Autonomous Vehicle Perception: Design and Verification

Shuai Wang*, Chengyang Li*, Qi Hao, Chengzhong Xu, Derrick Wing Kwan Ng, Yonina C. Eldar, and H. Vincent Poor

***Abstract*—Realizing human-like perception is a challenge in open driving scenarios due to corner cases and visual occlusions. To gather knowledge of rare and occluded instances, federated learning empowered connected autonomous vehicle (FLCAV) has been proposed, which leverages vehicular networks to establish federated deep neural networks (DNNs) from distributed data captured by vehicles and road sensors. Without the need of data aggregation, FLCAV preserves privacy while reducing communication and annotation costs compared with conventional centralized learning. However, it is challenging to determine the network resources and road sensor poses for multi-stage training with multi-modal datasets in multi-variant scenarios. This article presents networking and training frameworks for FLCAV perception. Multi-layer graph resource allocation and vehicle-road pose contrastive methods are proposed to address the network management and sensor pose problems, respectively. We also develop CarlaFLCAV, a software platform that implements the above system and methods. Experimental results confirm the superiority of the proposed techniques compared with various benchmarks.**

***Index Terms*—Autonomous driving, federated learning, perception, simulation**

I. Introduction

Perception determines the way a connected autonomous vehicle (CAV) understands the world by transforming environments into digits via sensors, signal processors, and deep neural networks (DNNs) [1]. Conventionally, DNN training is based on centralized learning, which collects the datasets from CAVs, trains the DNNs at the cloud, and deploys trained DNNs on CAVs for inference. This paradigm is effective in closed driving areas under the L2–L3 autonomous driving requirements, where the owner of data-collection vehicles is also the DNN provider. However, future L4–L5 CAV systems must cope with multi-variate open scenarios, which involves corner cases due to infinite scenario space and visual occlusions due to high scenario complexity [1]. CAV companies, e.g., Tesla, Waymo, Baidu, suggested that these challenges be tackled via lifelong multi-stage training that updates the DNN parameters whenever rare or occluded objects are detected (e.g., Tesla Autopilot solution https://karpathy.ai/). In this case, sensor data is inherently distributed at customers' vehicles and contains high-resolution human-related private information, leading to the potential of sensitive information leakage [2].

Federated (deep) learning empowered CAV (FLCAV) is an emerging paradigm to overcome the privacy issue by training DNNs via parameter and output aggregation instead of dataset aggregation [2]–[7]: parameter aggregation leverages vehicular networks to migrate knowledge among different vehicles [4], [5]; output aggregation leverages road sensors' perception results to annotate occluded objects for local parameter updates [6], [7]. The performance of FLCAV highly depends on the associated network resources and road sensor poses. Existing works on FLCAV design these factors from either a driving (e.g., [2], [3]) or networking (e.g., [4], [5]) perspective, which ignores the interdependency between driving tasks (i.e., data consumer) and vehicular networks (i.e., data provider). This research gap has been identified as the core issue in CAV systems [8]. Yet, for FLCAV systems, how to close this gap is still an open issue.

This article integrates driving and networking features for more robust *system-level* FLCAV perception under practical network resource constraints. Frameworks of vehicle-edge-cloud networking, multi-stage DNN training, and scenario-task matching, are presented. On top of these frameworks, multi-layer graph resource allocation (MLGRA) and vehicle-road pose contrastive (VRPC) approaches are proposed. The MLGRA jointly allocates the limited network resources across different stages, tasks, and modalities to minimize perception errors. The VRPC automatically reduces (increases) the number of sensors in low (high) complexity scenarios. To verify the above methods, it is necessary to implement a simulator with high-fidelity rendering qualities, driving behaviors, and software interfaces [9]. However, currently there is no such close-to-reality FLCAV simulator. We thus develop CarlaFLCAV (https://github.com/SIAT-INVS/CarlaFLCAV), an open-source software platform for design and verification of FLCAV systems. The platform contains dataset generation, perception tasks, FL frameworks, and optimization modules, and aims to provide a concrete step towards FLCAV in the real world.

II. CAV Meets FL

A. CAV Perception: Concept, Challenges, and Trends

1) CAV Perception: As shown in Fig. 1a, a CAV perception system is an ensemble of the following modules [1]: 1) semantic perception, e.g., recognition of lanes, road arrows, traffic signs, and weather, which outputs semantics

Shuai Wang is with the Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China (e-mail: s.wang@siat.ac.cn).

Chengyang Li and Qi Hao are with the Department of Computer Science and Engineering, Southern University of Science and Technology (SUSTech), Shenzhen 518055, China (e-mail: licy@mail.sustech.edu.cn, hao.q@sustech.edu.cn).

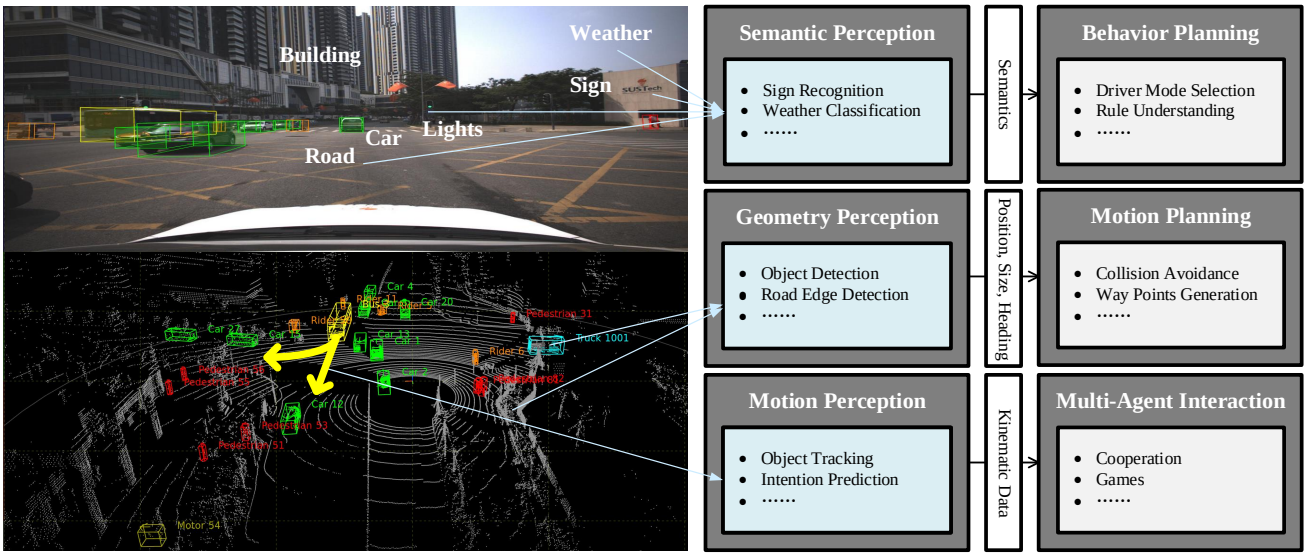
Chengzhong Xu is with State Key Lab of IOTSC, University of Macau, Macau, (e-mail: czxu@um.edu.mo)

Derrick Wing Kwan Ng is with the School of Electrical Engineering and Telecommunications, the University of New South Wales, Australia (e-mail: w.k.ng@unsw.edu.au).

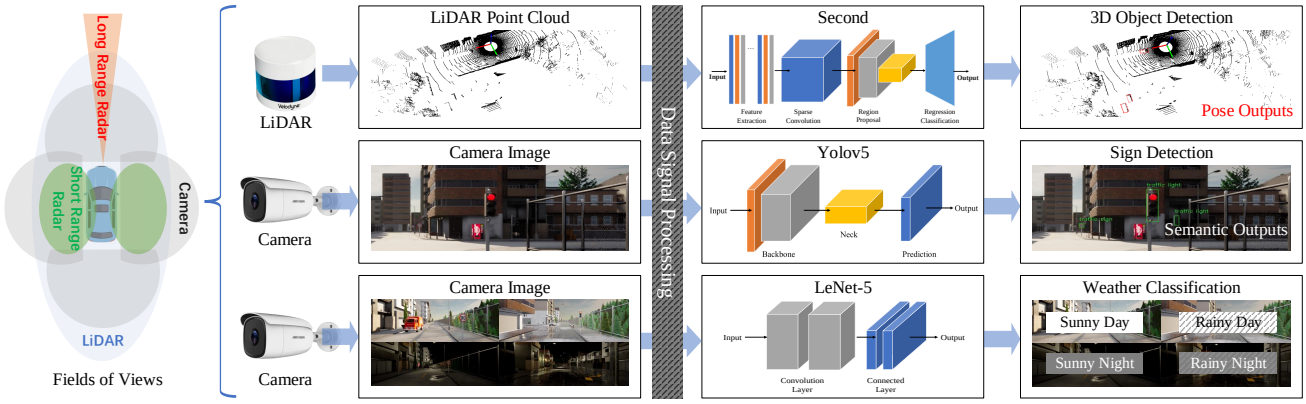
Yonina C. Eldar is with Weizmann Institute of Science, Rehovot, Israel (e-mail: yonina.eldar@weizmann.ac.il).

H. Vincent Poor is with the Department of Electrical and Computer Engineering, Princeton University, Princeton, NJ 08544 USA (e-mail: poor@princeton.edu).

*: Equal contribution.



(a)



(b)

Fig. 1: a) Categories and functionalities of CAV perception outputs. b) Multi-modal deep learning for CAV perception. From left to right: FoVs, hardwares, raw data, preprocessing methods, and DNNs for sign recognition, object detection, and weather classification. The datasets and perception outputs are generated by the CarlaFLCAV platform.

for rule-based behavior planning; 2) geometry perception, e.g., road edge detection and object detection, which outputs poses (positions, sizes, and orientations) for collision-avoidance motion planning; 3) motion perception, e.g., object tracking and intention prediction, which outputs kinematic data (velocities) for multi-agent interaction. Each module further consists of a cluster of tasks that may vary in different scenarios, but shares a common structure shown in Fig. 1b: 1) sensors transform the environment into digits (raw data) using active or passive electromagnetic waves; 2) digital signal processors clean/rotate/crop/complete the raw data; 3) DNNs extract features and generate semantic, geometry, motion outputs for subsequent CAV planning. CAV Perception tasks are highly-heterogenous, calling for multi-modal sensors such as RGB/infrared camera, LiDAR, radar, GPS and IMU to exploit their complementary features, e.g., field of views (FoVs). Consequently, multi-modal deep learning techniques are indispensable for CAV perception.

2) Perception Challenges: Major challenges of CAV perception in multi-variate open scenarios are summarized below.

- **Corner Case**. Scenario space in open areas grows exponentially, making it impossible to enumerate all the possible cases during the training stage [1]. In other words, there always exists new data outiside the distribution of training datasets, which are corner case instances.
- **Visual Occlusion**. In complex urban scenarios (e.g., cross-road, T-junction, roundabout), an object can be occluded by another in the FoV, which leads to incomplete data and perception errors [6].
- **Verification Cost**. Releasing new CAV perception DNNs requires rigorous verification. The cost of real-word testing is not acceptable due to the expensive infrastructures (e.g., vehicles, clouds, and networks to connect them all) and difficulties of testing in dangerous scenarios (e.g., crashes, overtaking, bad weather) [11].

3) Research Trends: Possible solutions to address the above challenges are as follows.

- **Lifelong multi-stage training**, which updates the parameters whenever a corner case is detected, is a promising solution to address the first challenge [3]. The update is executed on DNN copies, making sure that the inference DNNs are fixed during the training process. Releasing new-version DNNs requires the acknowledgement of DNN providers and costumers.
- As for the occlusion issue, the major solution is **cooperative perception with road sensors**. Practical road sensors are authoritative, having absolute positions, broader FoVs, and highly-optimized hardware units [6]. Furthermore, the probability of a target object being occluded in FoVs of all road sensors is significantly smaller than that in the FoV of a CAV. As such, the perception error due to occlusions is significantly mitigated.
- To reduce the high verification cost in reality, **CAV simulation** becomes a necessity [9]. Various CAV simulators have been released, e.g., Intel Car-Learning-to-Act (CARLA), NVIDIA DRIVE, Aerial-Informatics-and-Robotics-Simulation (AirSim), LG Silicon-Valley-Lab (LGSVL), Baidu Augmented-Autonomous-Driving-Simulation (AADS), OpenCDA, V2X-Sim [9], [11].

B. Paradigm Shift: From Centralized to Federated Learning

1) Federated Learning: The need for multi-stage training results in the shift of learning paradigms. In one-stage training, the owner of data-collection vehicles is also the DNN provider and the aggregated datasets can be directly fed to DNN training pipelines. However, for multi-stage training, the abnormal data is distributed at customers' vehicles and contains human-related private information. As a consequence, conventional centralized learning is no longer applicable. To this end, FLCAV emerges, which trains the DNNs from distributed datasets via parameter aggregation, thus conveying the knowledge of corner cases to other vehicles and remote servers while preserving data privacy [2]. In addition, communication costs are reduced, since the size of a DNN is significantly smaller than that of a data sequence, e.g., the size of a 1-minute point-cloud sequence is $1000\,\text{MB}$ while that of a standard object detection DNN is $60\,\text{MB}$ [1] (note that FL takes multiple rounds of communication; but in multi-stage training, only a few rounds are needed after pre-training).

2) Federated Distillation: FL can be integrated with cooperative perception, giving rise to the federated distillation (FD) technique [7] that simultaneously exploits distributed computing platforms and multi-view sensory data. In the FDCAV framework, all road sensors upload their bounding boxes and perception uncertainties to a road server. The server computes the weighted average of these outputs and this road-average output is downloaded to surrounding vehicles. Each vehicle updates its local parameters by minimizing the contrastive loss between the its bounding boxes and the road-average boxes [7]. In other words, the road's aggregated outputs are considered as noisy labels for occluded objects, which significantly saves the annotation cost compared with conventional manual labeling. For instance, the Road Experience Management (REM) of Mobileye (https://www.mobileye.com/our-technology/rem/) adopts road sensors to realize automated road semantic identification and annotation.

C. Related Work on FLCAV

1) Limitations of Existing Work: Current literature on FLCAV can be categorized into two types: 1) network-layer designs for vehicular FL, and 2) application-layer designs for FL perception and planning. Specifically, network-layer designs for FL, e.g., [4], [5], [10], aim to improve FL performance by controlling the communication-related variables such as topology, throughput, latency, and device scheduling. These works address the challenges of FL from the network perspective while taking the high mobility of vehicles into account. However, they fail in matching FL to domain-specific CAV scenarios, tasks, datasets, and DNNs. On the other hand, application-layer designs e.g., [2], [3], [6], [7], improve the safety and efficacy of CAV systems by designing DNN structures and associated information fusion methods. These works provide solutions to CAV issues such as training, inference, synchronization, calibration, and simulation. Nonetheless, communications among CAVs, edge servers, and cloud servers therein are assumed to be perfect, which does not hold for practical CAV systems with limited resources.

2) Research Opportunities: Since both types of designs have different pros and cons, it is necessary to integrate them to achieve lower detection/classification/tracking errors under network resource and sensor implementation constraints. This leads to new technical problems, which cannot be tackled by conventional methods.

- **Opportunity 1 (Network Resource Allocation)**: How can we effectively distribute the network resources across different stages, tasks, and modalities to minimize the perception errors of final-stage DNNs while satisfying the stringent wireless and wireline communication constraints? Existing literature does not analyze CAV-domain-specific datasets and ignores the interdependency across different stages, tasks, or modalities.
- **Opportunity 2 (Sensor Pose Optimization)**: How can we efficiently optimize the poses of road sensors to detect and annotate more occluded objects under the implementation cost constraint? Conventional approaches adopt integer programming (IP) solvers or heuristic methods to maximize the coverage with a fixed number of sensors, which ignores the learning requirements for FLCAV.
- **Opportunity 3 (Software Engineering)**: How can we implement a high-fidelity FLCAV simulator that is close to reality? Existing methods for network-level FLCAV are tested in simple classification tasks (e.g., recognition of handwritten digits). Emerging autonomous driving simulators do not support FL (and associated optimization) functionalities.

III. System-Level Design for FLCAV Perception

A. Network Architecture

As shown in Fig. 2a, FLCAV consists of CAV, road, and cloud components, forming a wide-range cyber-physical

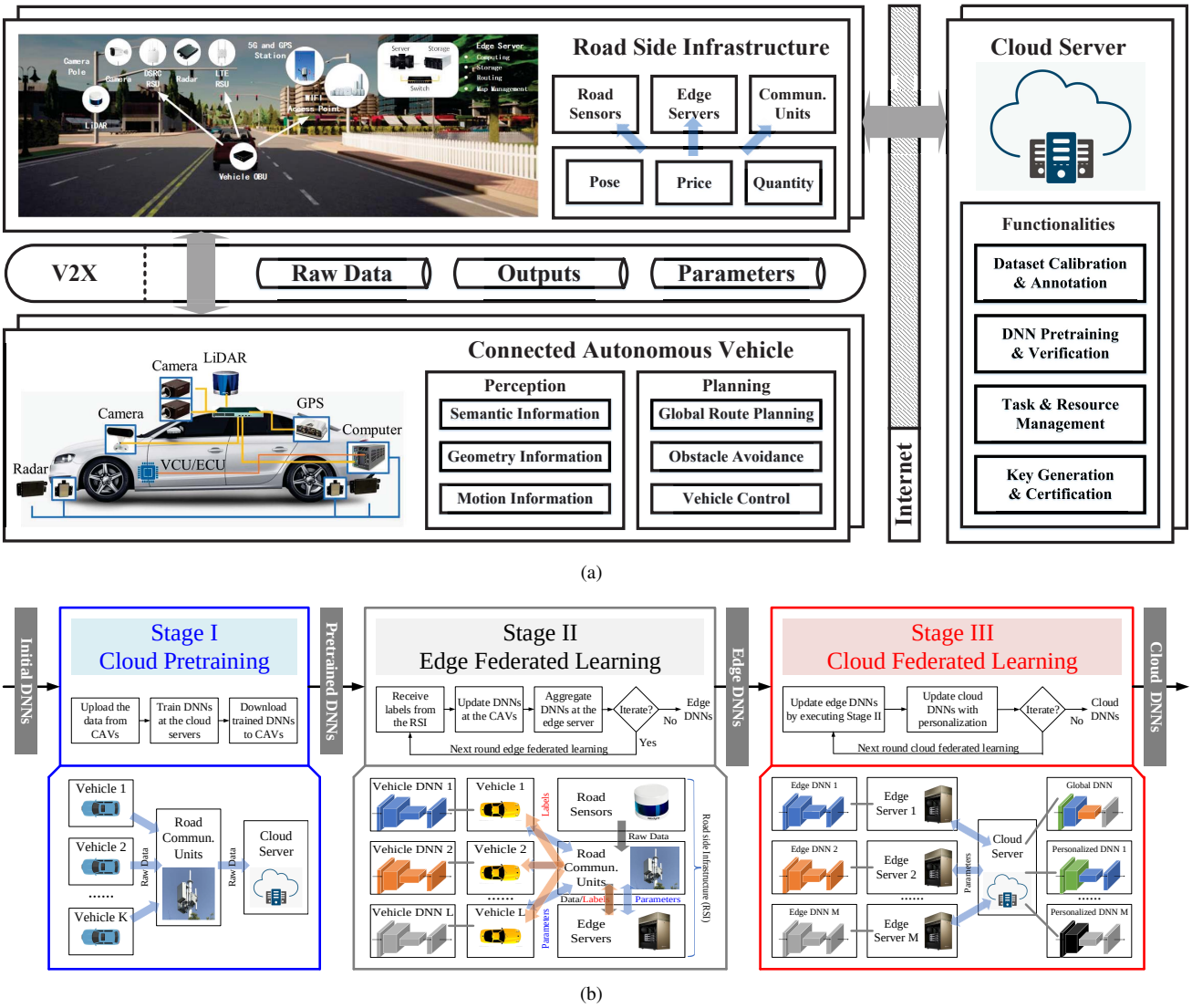


Fig. 2: a) Network architecture of FLCAV with the vehicle, road, and cloud modules and their associated V2X and the Internet links; b) Multi-stage training pipeline of FLCAV with intra-stage and inter-stage network flows.

system. CAVs are integrated systems equipped with multi-modal sensors, mobile computing platforms, and advanced communication units that 1) connect the vehicle sensors, computers, and chassis via controller area network (CAN), and 2) connect the vehicles with the cloud and road via vehicle-to-everything (V2X) technology [13]. CAV is a server within its CAN and a client within its V2X network.

Roadside infrastructure can be categorized into road sensors, road communication units, and road computing servers [8]. First, the pose of a road sensor includes position, yaw/roll/pitch angle, and resolution, which directly determines the coverage of the environment and objects therein. Second, road communication units adopt the V2X to link surrounding vehicles, forming a vehicular edge network. Finally, road servers process real-time tasks, e.g., feeding the data of road sensors into road DNNs, merging the multi-sensor data using fusion techniques (e.g., iterative closest point (ICP)), and acting as parameter servers for edge FL.

A cloud server differs from a road (edge) server since the Internet is a part of the end-to-end communication. Therefore, cloud servers execute long-term tasks, e.g., storage and annotation of datasets, training of perception DNNs, simulation and verification, and task and resource management. Note that to prevent potential cyber attacks, any vehicle joining FLCAV should be registered with a unique identifier allocated by authorities. The cloud server is responsible for maintaining the identifier and also executes security key generation and certification [8].

B. Training Pipeline

Training FLCAV perception systems in Fig. 2b consists of cloud pretraining, edge FL, and cloud FL stages. Specifically, cloud pretraining adopts annotated datasets on the cloud to transform initial DNNs into pretrained DNNs that are released to all CAVs. Then, an FL request is generated at the CAV upon a false detection event (e.g., due to occlusions or corner

cases). The FL request is sent to the edge parameter sever, who calls for roadside infrastructures and other vehicles to join the edge FL group via V2X. This helps the CAV fix the bug residing in its local DNN, as the knowledge migrates from other agents to it and vice versa. With a few rounds of output and parameter exchange, the edge parameter server can form edge DNNs that serve as good representations of the local region. Finally, the FL group can be enlarged by cloud FL, which aggregates the edge DNNs from remote areas via the Internet. To improve robustness, the cloud FL stage may adopt personalization such that each edge client trains its own regional DNNs while contributing to the global cloud DNNs [12].

From an application perspective, the FLCAV system needs to train a set (let N denote its cardinality) of FLCAV DNNs for associated perception tasks in 3 (or more) consecutive stages. From a network perspective, FLCAV involves two types of communications, i.e., wireless and wireline communication, and their transmission capacities are finite. Combining both, we conclude that the summation of wireless/wireline throughput (in MBytes) over N tasks and 3 stages should be smaller than the corresponding network throughput budgets, and there exists a tradeoff among different tasks and stages. Here we consider the uplink transmission in Fig. 2b, as the downlink counterpart is usually not the bottleneck in practice.

- For Stage I, data samples should be uploaded from vehicles to the edge and then to the cloud. The number of samples $\times$ the data size of each sample should not exceed the minimum throughput of wireless and wireline communication allocated to Stage I.
- For Stage II, parameters are exchanged between the edge server and vehicles through wireless communication. The number of edge FL rounds $\times$ the number of vehicles in each FL group $\times$ the data size of DNNs should not exceed the wireless throughput allocated to Stage II.
- For Stage III, parameters are exchanged between the edge and cloud servers through wireline communication. The number of cloud FL rounds $\times$ the number of edge servers $\times$ the data size of DNNs should not exceed the wireline throughput allocated to Stage III.

Note that perception outputs (e.g., bounding boxes) are also shared among nodes, but the associated communication overhead is negligible compared with those of samples and DNNs.

C. Task Generation

Tasks should match scenarios [14] and their construction shown in Fig. 3a consists of the following 4 steps.

1) **Operational Design Domain (ODD) Specification**. Given the target ODD, e.g., urban (focus of this paper), rural, campus, mine, port, or parking-lot, we define the key parameters including traffic density, speed limits, rules, and FoV requirements [15].
2) **Scenario Sampling**. Scenarios are sampled inside the ODD according to industrial standards such as ISO and SAE [15]. Learning and optimization based methods can also be adopted for scenario space exploration. CarlaFLCAV provides straight-road, cross-road, T-road, and roundabout scenarios.
3) **Task Generation**. Scenario-specific tasks are generated, where each task is defined as a set of data, labels and DNNs [14]. We construct fewer tasks for low-complexity scenarios to save computational costs and redundant tasks for high-complexity scenarios to guarantee driving safety.
4) **Task Evaluation**. Perception tasks are categorized into different priorities via a task importance evaluator, which removes a task from the task list and a task is deemed important if the performance loss is significant [14].

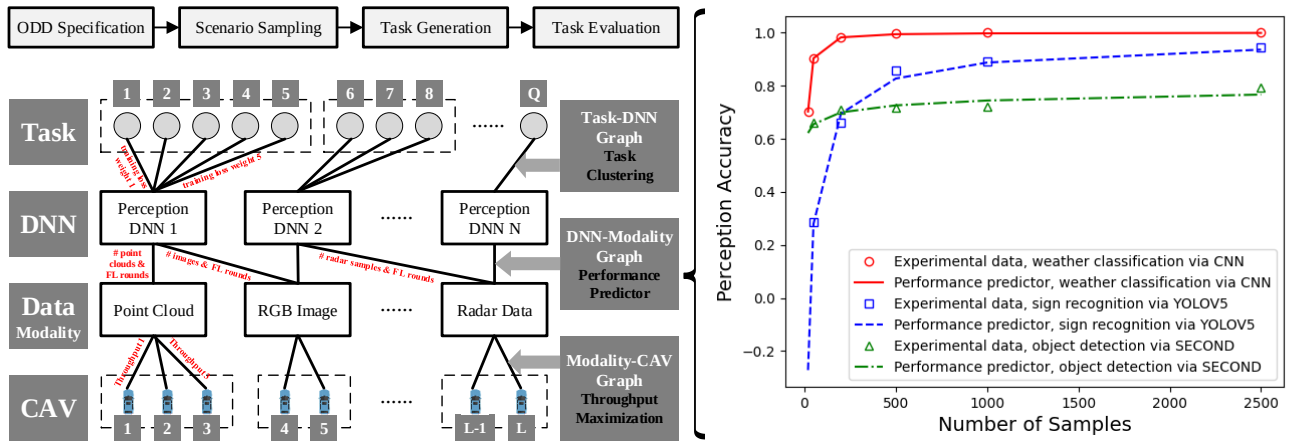
IV. Resource Optimization for FLCAV Perception

A. Multi-Modal Resource Allocation

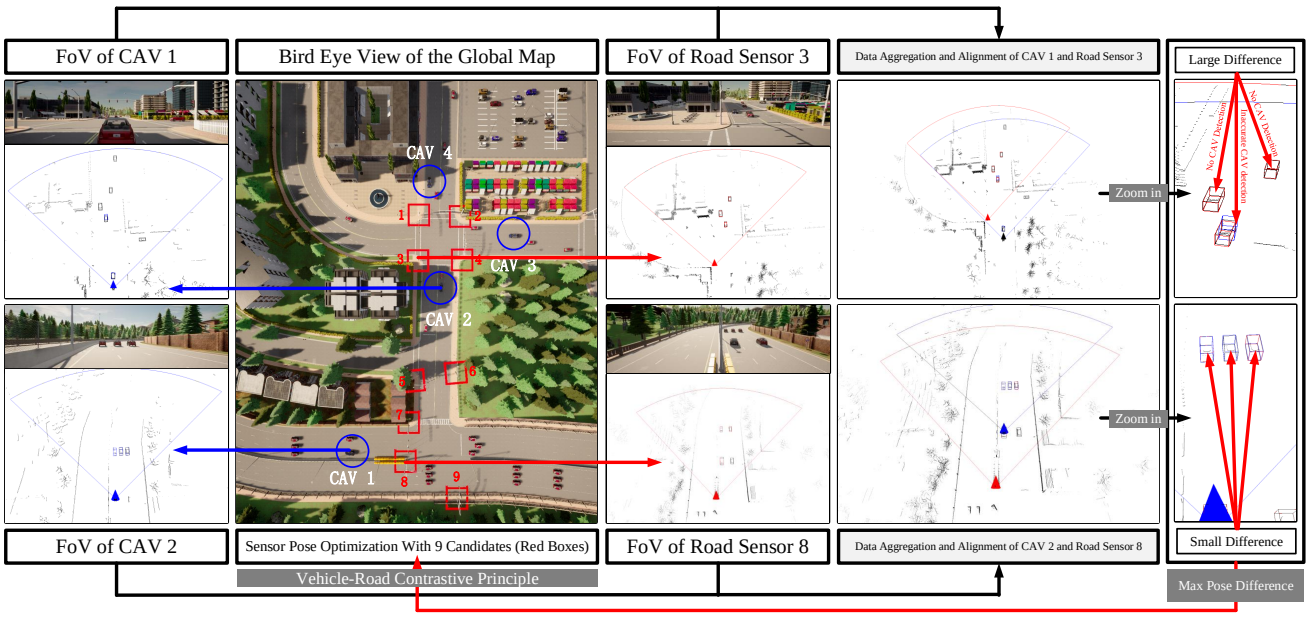
1) Motivation: Intuitively, more resources should be allocated to perception tasks accomplished by deeper neural networks rather than equally allocated to different tasks. Consider training a convolution neural network (CNN) for weather classification and a sparsely embedded convolutional detection (SECOND) network for object detection. Due to larger number of parameters ($\sim$5 millions) in SECOND, the FLCAV network should allocate more communication resources to vehicles that upload point clouds for pretraining SECOND in Stage I, and call for more vehicles and FL rounds for updating SECOND in Stages II and III. In current vehicular networks, the purpose of resource optimization is to improve key communication metrics such as throughput, connectivity, and latency, which treats data equally and violates the above intuition.

2) Method: Since communication (i.e., a data pipeline) becomes a sub-task in the FLCAV paradigm, we need to directly minimize the perception error instead of the communication costs. The problem becomes how to model the relation between perception errors and network flows. Here we present a multi-layer graph resource allocation (MLGRA) approach shown in Fig. 3a, where each vertex represents some task, DNN, modality, or CAV, and each edge represents training weight, data priority (i.e., number of samples and FL rounds allocated to each DNN), or vehicle throughput.

- **Task-DNN Graph**. Tasks are clustered into groups via a task-DNN graph such that similar tasks share the same DNN and annotated dataset. For example, the tasks of box regression (i.e., determining the poses of objects), object classification (e.g., determining if the object is a car or a truck), and orientation classification (e.g., determining the front side) can be accomplished by one DNN with a common feature extractor and 3 separate headers. The training loss function is the weighted sum of three metrics.
- **DNN-Modality Graph**. Different DNNs may be trained with different data modalities, and their connections form a DNN-modality graph. Edges of the graph represent priorities of different data modalities, which can be measured by fitting performance predictors (e.g., parametric inverse-power model in Fig. 3a) to historical KPI datasets (e.g., perception accuracy) as shown in Fig. 3a.
- **Modality-Vehicle Graph**. For each data modality (e.g., point cloud data), the associated data samples may come



(a)



(b)

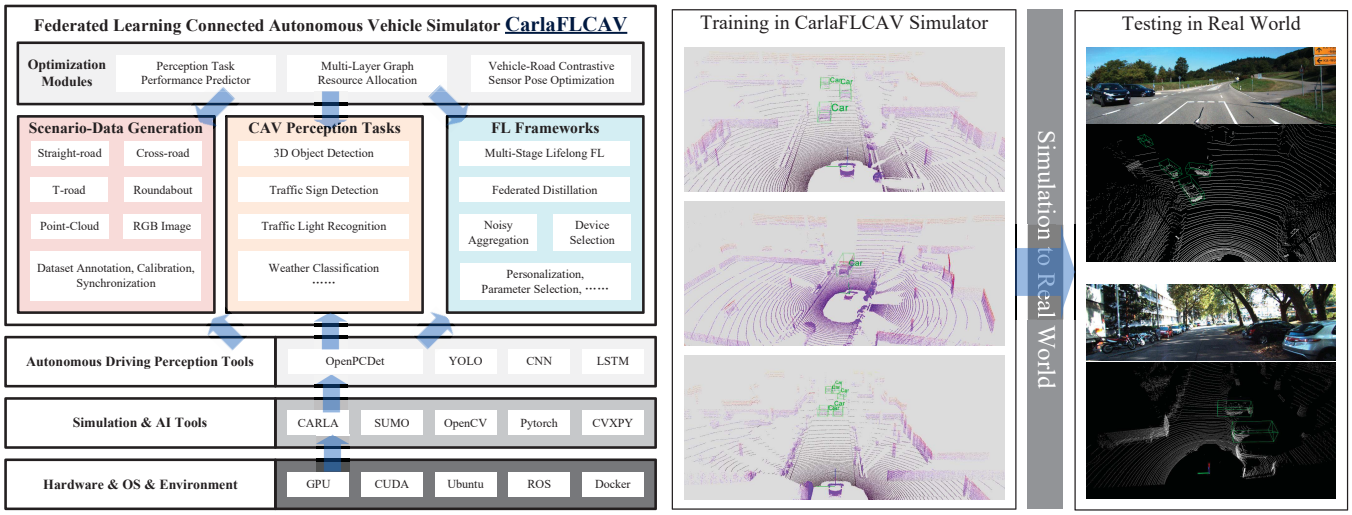
Fig. 3: a) Illustration of the MLGRA method and the good fitness of the performance predictor to the experimental data generated by CarlaFLCAV; b) Illustration of the road-assisted FD and the VRPC principle. The black box is the ground truth; the blue box is from the CAVs; the red box is from the road sensors.

from multiple autonomous vehicles. If the data is independent and identically distributed (IID) among different vehicles, we can maximize the total throughout via water-filling algorithms. If the data is non-IID due to different weather and FoVs, it is necessary to evaluate the quality of vehicles' data using the uncertainty sampling method. For example, a throughput limit should be imposed on CAVs that repeat uploading the same data to improve the dataset quality.

B. Road Sensor Pose Optimization

1) Motivation: For FLCAV, the key is to train the vehicle DNNs instead of monitoring the environment. Therefore, in contrast to conventional methods that maximize the number of visible objects [6], our principle places sensors at critical scenarios that contain adversarial objects that can defeat the DNNs. To illustrate their difference, we adopt CarlaFLCAV to generate an urban traffic map with straight-road and cross-road scenarios, where 9 possible sensor locations are marked as red boxes in the middle of Fig. 3b.

- Conventional methods [6] place the road sensor at location 8, as we set its nearby traffic density to the highest value. However, the vehicle DNN would not learn any new knowledge, as the bounding boxes generated by the vehicle (e.g., CAV 2) and the road sensor 8 are similar.
- Our principle places the road sensor at location 3, despite the fact that its nearby traffic density is low. Here, the CAV could change its local parameters to the maximum extent with the road's outputs. This is because the bounding boxes generated by the vehicle and road sensor 3 are very different. For example, CAV 1 it misses two objects and generates one inaccurate box due to its FoV being



(a) (b)

Fig. 4: a) Software architecture and supported functionalities of CarlaFLCAV; b) Training DNNs in the CarlaFLCAV simulator and testing with real world datasets. The DNN trained with CarlaFLCAV successfully detects all objects in the real world.

blocked by its front car.

2) Method: To find the critical scenarios, our VRPC principle minimizes the pose similarity between the detected objects at the CAV and those at the road sensor. In particular, we deploy the pretrained DNN on a CAV and test the vehicle in a target map containing multiple scenarios, each specified with zone limits. If the CAV fails to detect an object or generates a false positive in its FoV, the 3D position of this object (false positive) $\mathbf{e}$ is registered into a database. Let the set $\mathcal{E} = \{\mathbf{e}_1, \mathbf{e}_2, \cdots\}$ denote all registered error items after sufficient simulation time, and scenarios containing false detections are deemed important. The cardinality of the set $\{\mathbf{e} \in \mathcal{E} : \|\mathbf{e} - \mathbf{x}\| \leq r\}$ represents the expected number of false detections that could be calibrated by the road sensor, which is to be maximized, where $\mathbf{x}$ and r are the position and accurate-detection range of the road sensor. The problem of optimizing $\mathbf{x}$ is then a discrete optimization problem where a finite set of possible locations is available, as sensors can only be attached to utility poles and traffic lights. In addition, r is a monotonically increasing function of the implementation cost, which can also be obtained via curve fitting. For example, for LiDAR object detection, the detection error is determined by the number of sensed points on the object, which is monotonically decreasing w.r.t. the sensor-object distance and increasing w.r.t. the number of laser channels. Increasing the number of laser channels will increase r, but will also increase the LiDAR price, e.g., with the same budget we can buy 1x 64-line LiDAR or 3x 32-line LiDAR or 9x 16-line LiDAR. Consequently, r controls the trade-off between the perception performance and the infrastructure cost.

V. Implementation and Experiments

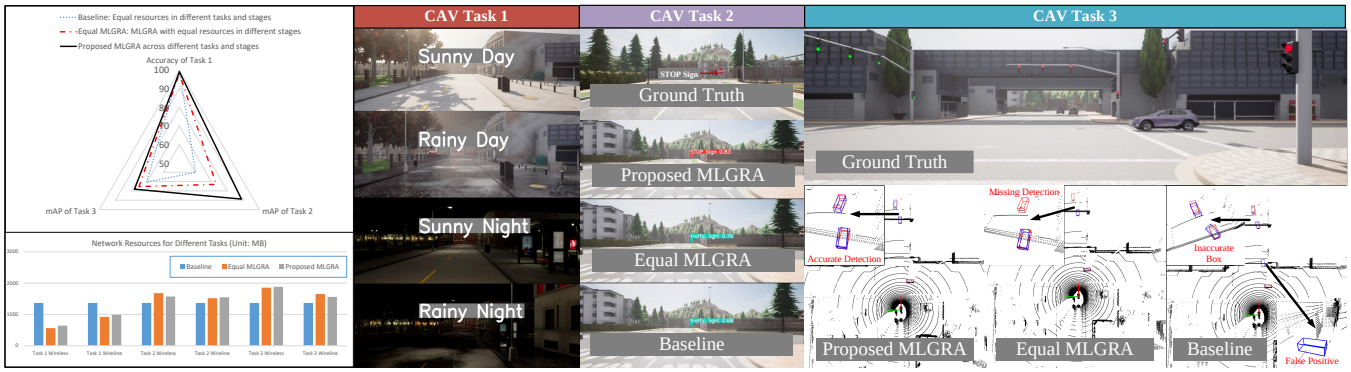
A. Software Architecture

CarlaFLCAV (shown in Fig. 4a) is an open-source FLCAV simulation platform that supports: (1) multi-modal dataset generation, including point-cloud, image, radar data with associated calibration, synchronization, and annotation; (2) training and inference examples for CAV perception, including object detection, traffic sign detection, and weather classification; (3) various FL frameworks, including FedAvg, device selection, noisy aggregation, parameter selection, distillation, and personalization; and (4) optimization modules, including network resource and road sensor pose optimization. The implementation of (2) is based on OpenPCDet, Yolov5, LeNet-5, with the associated results shown in Fig. 1b. The implementation of (4) is illustrated in Fig. 3. Below we focus on (1) and (3).

Specifically, raw sensory data is recorded using CARLA [11], which adopts Unreal Engine 4 for state-of-the-art visual rendering and physics simulation. Then, calibration represents sensed information in a common coordinate system via rotation and transition matrices. CarlaFLCAV assumes perfect calibration, but practical systems may involve errors since sensors' intrinsic (e.g., shape of the camera lens) and extrinsic (i.e., pose) parameters need to be estimated. Time stamping adopts LiDAR as a reference, i.e., each laser spin is a frame. Synchronization among different sensors can be realized via hardware or software trigger, and the worst-case time difference is at the millisecond level. Finally, data annotation tracks the categories, poses, and occlusions of objects via CARLA APIs (https://carla.readthedocs.io/en/latest/).

For wireline FL, the DNN parameter exchange can be implemented based on the Robot Operating System (ROS) communication, which offers inter-process communication among distributed nodes by publishing or subscribing topics. For wireless FL, due to limited capacity of wireless channels, only a subset of CAVs can be selected to convey their parameter updates at each FL round. Thus, CarlaFLCAV implements importance-aware device selection, where vehicles with larger gradient norms are assigned a higher probability to be scheduled. Besides, CarlaFLCAV includes noisy aggregation, which

(a)



(b)

Fig. 5: a) Comparison among different schemes for single-task perception. Dataset: 5000 point-cloud samples in Town02 for pretraining; 4 CAVs (each with 500 samples) in Town05 for edge FL; 3 CAVs (each with 500 samples) in Town03 for cloud FL; 2432 samples in Town05 for testing. b) Comparison among different schemes for multi-task perception. Task-1 dataset: 4000 RGB images in Town02 for pretraining dataset; 4 CAVs (each with 100 images) in Town05 for edge FL; 3 CAVs (each with 100 images) in Town03 for cloud FL; 2000 images in Town02 for testing. Task-2 dataset: 2500 RGB images in Town02 for pretraining; 4 CAVs (each with 1000 images) in Town05 for edge FL;3 CAVs (each with 1000 images) in Town03 for cloud FL; 3279 images in Town05 for testing. Task-3 dataset: the same as Fig. 5a with road sensor placed at position 3.

injects random noises into the DNN parameters as a mask to protect the data privacy against model inversion attacks. DNN frozen is optional, which fixes partial layer parameters to reduce the communication cost.

B. Experimental Validation

First, to evaluate how close CarlaFLCAV is to real-life conditions, we train the SECOND network (in Fig. 1b) with 3000 point cloud samples generated by CarlaFLCAV for 60 epochs and test the trained SECOND on a real-world dataset KITTI. The mean average precision (mAP) at IoU$=0.5$ is 58% for object detection. Qualitative results are shown in Fig. 4b, where the DNN trained with CarlaFLCAV detects all objects in real data.

Next, to verify the effectiveness of the multi-stage FLCAV, we consider the single-task case (i.e., object detection) and compare the mAP of final DNNs for different schemes, under the same wireless and wireline uplink resource budgets (i.e., 4 GB). The settings and results are shown in Fig. 5a. Our major findings are summarized below.

(i) All FL schemes achieve higher mAPs than centralized learning. This demonstrates the necessity of exploiting domain-information of Town05.

(ii) With cloud FL, the mAP is improved, since the CAVs in Town03 provide new knowledge about corner-case and occluded objects.

(iii) With equal resources in different stages, the VRPC placing the road sensor at position 3 significantly improves

the mAP compared with the conventional method placing the sensor at position 8.

(iv) By jointly optimizing the network resources across three stages, the mAP is further increased to 84.58, which demonstrates the effectiveness of MLGRA. Our result implies that more wireless (wireline) resources should be allocated to the edge (cloud) FL stage.

(v) Under the same VRPC and MLGRA methods, the mAP score with device selection is slightly higher than direct FL, as the stragglers are removed from FL groups.

Finally, we simulate the multi-task multi-modal perception, including object detection, sign recognition, and weather classification tasks. The settings and results are shown in Fig. 5b.

(i) Since our goal is to maximize the perception accuracy of all tasks, a larger area indicates better performance. The baseline scheme (i.e., blue dotted-line) has the smallest area, which can be treated as a worst-case performance bound.

(ii) Leveraging the performance predictor in Fig. 3a, the equal MLGRA method (i.e., red dashed-line) achieves a larger triangle area than that of the baseline. This is because it automatically allocates more resources to object detection and sign recognition as shown in Fig. 5b, which are more important and difficult tasks.

(iii) With joint resource allocation across different stages and tasks, the proposed MLGRA method (i.e., black solid-line) achieves the largest triangle in Fig. 5b. The perception accuracies are $99.0, 88.8, 78.16$ for tasks $1, 2, 3$, respectively.

(iv) For task 2, the proposed MLGRA successfully detects the STOP sign in rainy days, while other methods misclassify the sign as the traffic light. For task 3, the proposed MLGRA successfully detects all the objects at the cross-road, while other methods involve inaccurate detections and false positives. This demonstrates the excellent generalization performance of MLGRA.

VI. Conclusion

This article has reviewed the integration of FL and CAV for overcoming perception challenges in open driving scenarios. The vehicle-edge-cloud networking, multi-stage training, and multi-task generation frameworks of FLCAV were presented. The MLGRA and VRPC methods were proposed to solve the network management and sensor pose problems, respectively. The frameworks and methods were verified in a software platform CarlaFLCAV. Future directions are listed below.

Simulation-to-reality transfer. Real-world CAV datasets involve far more physical mechanisms (e.g., illuminations) and interactive behaviors (e.g., competitions) than simulation datasets. Deep generative adversarial networks can be adopted to close the gap between the digital and physical systems.

Autonomous driving under perception uncertainties. Perception uncertainties will propagate to the subsequent planning system. Hence, a scientific approach for computing the perception uncertainties is needed, which adjusts the collision avoidance constraints to balance safety and efficiency. End-to-end autonomous driving that directly maps sensor inputs into vehicle actions is also a promising solution to address the uncertainty propagation issue.

References

[1] D. Feng et al., "Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 3, pp. 1341–1360, Mar. 2021.

[2] S. Savazzi, M. Nicoli, M. Bennis, S. Kianoush, and L. Barbieri, "Opportunities of federated learning in connected, cooperative and automated industrial systems," *IEEE Commun. Mag.*, vol. 59, no. 2, pp. 16–21, Feb. 2021.

[3] B. Liu, L. Wang, M. Liu, and C.-Z. Xu, "Federated imitation learning: A novel framework for cloud robotic systems with heterogeneous sensor data," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 3509–3516, Apr. 2020.

[4] J. Posner, L. Tseng, M. Aloqaily, and Y. Jararweh, "Federated learning in vehicular networks: Opportunities and solutions," *IEEE Netw.*, vol. 35, no. 2, pp. 152–159, Mar./Apr. 2021.

[5] J. Zhang and K. B. Letaief, "Mobile edge intelligence and computing for the Internet of vehicles," *Proc. IEEE*, vol. 108, no. 2, pp. 246–261, Feb. 2020.

[6] E. Arnold, M. Dianati, R. de Temple, and S. Fallah, "Cooperative perception for 3D object detection in driving scenarios using infrastructure sensors," in *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 3, pp. 1852–1864, Mar. 2022.

[7] Y. Li, S. Ren, P. Wu, S. Chen, C. Feng, and W. Zhang, "Learning distilled collaboration graph for multi-agent perception," *Adv. Neural Inf. Process. Syst.*, virtual, Dec. 2021, pp. 1–12.

[8] I. Yaqoob, L. U. Khan, S. M. A. Kazmi, M. Imran, N. Guizani, and C. S. Hong, "Autonomous driving cars in smart cities: Recent advances, requirements, and challenges," *IEEE Netw.*, vol. 34, no. 1, pp. 174–181, Jan./Feb. 2020.

[9] W. Li et al., "AADS: Augmented autonomous driving simulation using data-driven algorithms," *Sci. Robot.*, vol. 14, no. 28, pp. 1–12, Mar. 2019.

[10] M. Chen, D. Gündüz, K. Huang, W. Saad, M. Bennis, A. V. Feljan, and H. V. Poor, "Distributed learning in wireless networks: Recent progress and future challenges," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 12, pp. 3579–3605, Dec. 2021.

[11] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proc. CoRL*, 2017, pp. 1–16.

[12] Q. Wu, X. Chen, Z. Zhou, and J. Zhang, "FedHome: Cloud-edge based personalized federated learning for in-home health monitoring," *IEEE Trans. Mob. Comput.*, Dec. 2020. DOI: 10.1109/TMC.2020.3045266.

[13] A. Eskandarian, C. Wu, and C. Sun, "Research advances and challenges of autonomous and connected ground vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 2, pp. 683–711, Feb. 2021.

[14] Z. Zheng, Q. Chen, C. Hu, D. Wang, and F. Liu, "On-edge multi-task transfer learning: Model and practice with data-driven task allocation," *IEEE Tran. Parall. Distr.*, vol. 31, no. 6, pp. 1357–1371, Jun. 2020.

[15] X. Zhang et al., "Finding critical scenarios for automated driving systems: A systematic literature review," [Online]. Available: https://arxiv.org/abs/2110.08664v1, Oct. 2021.