2021 IEEE International Conference on Robotics and Automation (ICRA 2021)
May 31 - June 4, 2021, Xi'an, China

Distributed Dynamic Map Fusion via Federated Learning for Intelligent Networked Vehicles

Zijian Zhang[1,5], Shuai Wang[1,2,3], Yuncong Hong[2], Liangkai Zhou[2], and Qi Hao[1,3,4,*]

***Abstract*— The technology of dynamic map fusion among networked vehicles has been developed to enlarge sensing ranges and improve sensing accuracies for individual vehicles. This paper proposes a federated learning (FL) based dynamic map fusion framework to achieve high map quality despite unknown numbers of objects in fields of view (FoVs), various sensing and model uncertainties, and missing data labels for online learning. The novelty of this work is threefold: (1) developing a three-stage fusion scheme to predict the number of objects effectively and to fuse multiple local maps with fidelity scores; (2) developing an FL algorithm which fine-tunes feature models (i.e., representation learning networks for feature extraction) distributively by aggregating model parameters; (3) developing a knowledge distillation method to generate FL training labels when data labels are unavailable. The proposed framework is implemented in the CARLA simulation platform. Extensive experimental results are provided to verify the superior performance and robustness of the developed map fusion and FL schemes.**

I. INTRODUCTION

The intelligent networked vehicle system (INVS) is an emerging vehicle-edge-cloud system that accomplishes cooperative perception, map management, planning and maneuvering tasks via vehicle-to-everything (V2X) communication [1]–[3]. Among all the tasks, distributed map management aims to enlarge sensing ranges and improve sensing accuracies for individual vehicles and plays a central role in INVSs [4]–[15]. While static maps describe stationary objects (e.g., roads, buildings, and trees), dynamic maps emphasize updating information of mobile objects (e.g., pedestrians, cars, and animals) in real time. Fig. 1 shows the architecture of an intelligent networked vehicle system. There are three main steps for dynamic map fusion: (1) local sensing and perception, (2) local map fusion and uploading, and (3) global map fusion and broadcasting.

Despite many efforts and successes in developing dynamic map fusion techniques [4]–[15], a number of technical challenges still need to be properly handled, including

This work was supported in part by the Science and Technology Innovation Committee of Shenzhen City under Grant JCYJ20200109141622964, in part by the Intel ICRI-IACV Research Fund (CG52514373), and in part by the National Natural Science Foundation of China under Grant 62001203,.

*Corresponding author: Qi Hao (hao.q@sustech.edu.cn).

[1]Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China.

[2]Department of Electrical and Electronic Engineering, Southern University of Science and Technology, Shenzhen 518055, China.

[3]Sifakis Research Institute of Trustworthy Autonomous Systems, Southern University of Science and Technology, Shenzhen 518055, China.

[4]Pazhou Lab, Guangzhou, 510330, China.

[5]Harbin Institute of Technology, 92 West Dazhi Street, Nan Gang District, Harbin, 150001, China.

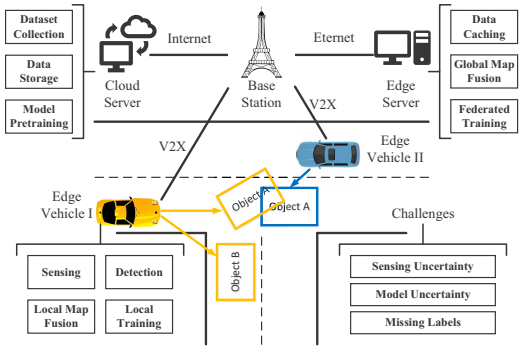


Fig. 1: An illustration of the intelligent networked vehicle system and three challenges for dynamic map fusion.

1) **Reduction of sensing uncertainties**. The sensing data at each individual vehicle may be missing, noisy, or mistaken due to sensor limitation and environmental complexity. A proper use of the redundant measurements from other vehicles can help reduce such uncertainties.
2) **Reduction of model uncertainties**. The construction of dynamic maps at vehicles or edge servers relies on the quality of feature models of mobile objects. Those feature models, trained with labeled datasets at the cloud, may not be perfectly optimized due to many factors.
3) **Missing labels for online learning**. While the datasets used at the cloud have been manually annotated, new data samples collected at vehicles are usually unlabeled. Those local sensory data samples should be automatically labeled for high-quality dynamic map updating.

Current dynamic map fusion techniques can be classified as data level [4], [9], feature level [5]–[8], and object level [9]–[15]. Those data-level and feature-level methods likely incur high communication overheads and most object-level methods do not take uncertainties into account. On the other hand, various deep and federated learning schemes have been used to improve the quality of feature models [8], [14], [15], but few of them use point-cloud data. Besides, as most cloud-based model training schemes assume that datasets have been properly labeled [4]–[9], [11], [14], [15], knowledge distillation (KD) methods [16]–[18] need further investigation for label generation among networked vehicles.

In this paper, we propose an FL based dynamic map fusion framework, which enables scored-based object-level fusion and distributed online learning to achieve high map quality and low communication overhead. The main contributions of this work are summarized as follows.

TABLE I: A Comparison of Dynamic Map Fusion Schemes for Networked Intelligent Vehicles

Scheme	Literature	Sensing and Fusion			Perception Model		Data Label	Limitation
		Sensor Modality	Comm. Overhead	Weighted Fusion	Deep Networks	Federated Training	Online Generation	
Data Level	[4], [9]	LiDAR	+++	✓	✓	✗	✗	Demands for high communication workload and data synchronization
Feature Level	[5]–[7]	LiDAR	++	✓	✓	✗	✗	Demands for consistent feature extraction and data synchronization
	[8]	Camera	++	✓	✓	✓*	✓*	
Object Level	[10], [12] [13]	GPS	+	✗	✗	✗	✗	Impractical assumptions on detection models
	[4], [11]	LiDAR	+	✗	✓	✗	✗	Requirement for high-quality pretrained feature models
	[14], [15]	Camera	+	✗	✓	✓*	✓*	Low accuracy of object location and orientation
	Ours	LiDAR	+	✓	✓	✓	✓	Assumptions on high accuracy of ego-vehicle localization

The symbol "✓" means functionality available, "✗" means functionality not available, "✓*" means functionality not available but supported. The number of "+"s represents the level of communication overhead, more "+"s means higher overhead.

1) Developing a three-stage map fusion, including the density based spatial clustering of applications with noise (DBSCAN), the score-based weighted average, and the intersection over union (IoU) based box pruning, to achieve global map fusion at the edge server.
2) Developing a point-cloud based distributed FL algorithm, which fine-tunes feature models of mobile objects distributively by aggregating model parameters.
3) Developing a KD method to generate data labels for training individual vehicles. The labels of real-time sensory data are provided by road side units or generated from the three-stage fusion algorithm at the edge server.
4) Implementation of the proposed FL-based dynamic map fusion framework in the CARLA simulation platform [19] with extensive testing experiments in terms of developed evaluation benchmark metrics (the open-source codes are available at https://github.com/zijianzhang/CARLA_INVS).

II. Related Work

As summarized in Table. I, current dynamic map fusion techniques can be categorized into data level [4], [9], feature level [5]–[8], and object level [9]–[15]. Data level methods collect raw data (such as point clouds) from different vehicles and achieve global dynamic map fusion by using deep networks [4], [9] at the cost of high communication overhead and extra data synchronization. Feature level methods deploy identical feature extraction models (such as voxelization [5]–[7] and segmentation [8]) at different vehicles, and achieve global dynamic map fusion via feature collection and feature processing based on deep networks. Object level methods usually share the object lists among vehicles and achieve global dynamic map fusion through object association and multi-stage fusion procedures [9]–[15].

The object lists from different vehicles can be registered by using either ego-vehicle pose information [10], [12], [13] or object geometric features [4], [11], [14], [15]. The former scheme needs constant communications among all ego-vehicles to estimate the transforms between any two poses and relies on impractical assumptions on detection models. The later scheme relies more on the quality of object pose and size information predicted by each connected ego-vehicles and resolves conflicts via max-score fusion [4] or mean fusion [11]. Distance and IoU metrics have often been used to associate object lists, however leading to inaccurate estimation of the number or fused poses of objects [10], [12].

Various deep learning techniques have been developed to improve the quality of object feature models [4]–[9], [11], [14], [15]. In particular, to learn from distributed data at different vehicles, federated learning (FL) schemes [16]–[18] have been developed for camera images and convolutional neural network based feature models. This means that FL can be applied to image-based dynamic map fusion [8], [14], [15]. However, point cloud data can provide more accurate depth information, which have not been integrated with FL for dynamic maps. On the other hand, deep network models trained at the cloud [4]–[9], [11], [14], [15] assume that training datasets have been properly labeled, but real-world applications require label generation for new data samples during online learning procedures via knowledge transfer from "teacher" models to "student" models [16]–[18]. Existing knowledge transfer either uses predictions of a single vehicle or average predictions of multiple vehicles as the "teacher" model, which may not fully exploit the useful information from all connected vehicles.

In this work, we propose an FL based dynamic map fusion framework, which enables scored-based object-level fusion and distributed online learning to achieve high map quality and low communication overhead. In the fusion process, we choose a density based clustering algorithm for object association and develop a dynamic weight adjustment algorithm to adaptively fuse object information. Furthermore, the object-level fusion results are chosen as the teacher model output for online federated learning. The complete framework of FL and KD based dynamic map fusion is developed based on multi-agent point-cloud datasets in CARLA.

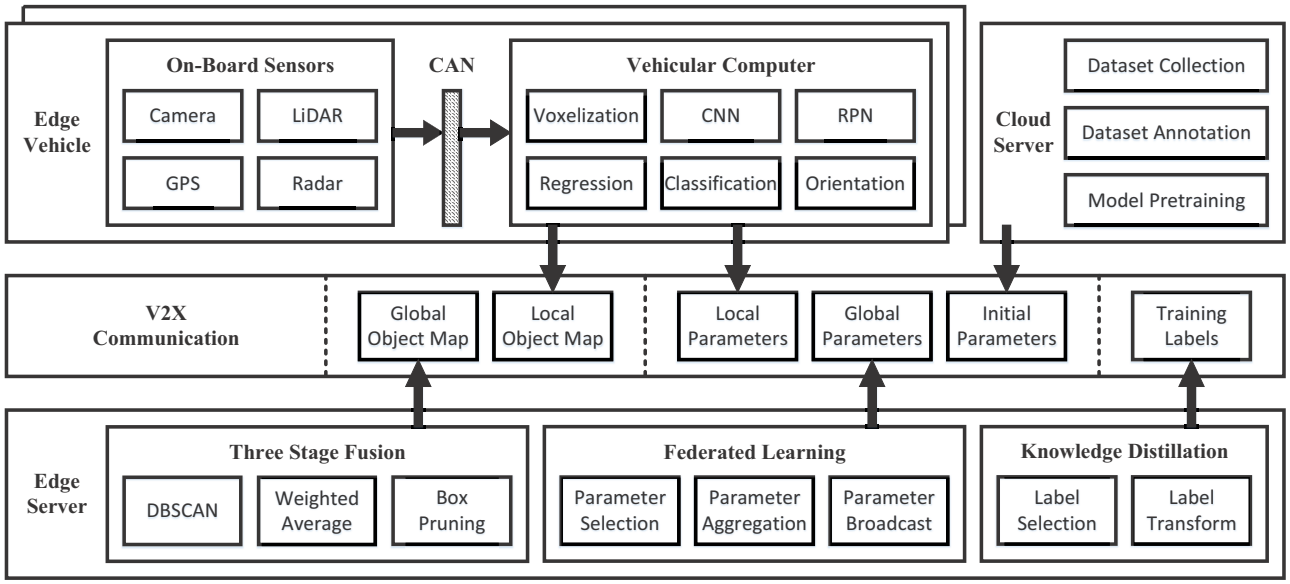


Fig. 2: The proposed FL-based dynamic map fusion framework, which contains edge vehicles to generate local maps, a cloud server to perform feature model pre-training, an edge server to perform global map fusion, federated learning and knowledge distillation.

III. System Architecture and Algorithm Design

We consider an environment which consists of M dynamic objects and K intelligent vehicles. At the t-th time slot, the m-th object (with $1 \leq m \leq M$) is represented by

$$\mathbf{v}_m(t) = [c_m, x_m, y_m, z_m, l_m, w_m, h_m, \theta_m]^T, \tag{1}$$

where c_m is the category, (x_m, y_m, z_m) are the center coordinates, (l_m, w_m, h_m) stand for the length, width, and height, and θ_m denotes the yaw rotation around the z-axis for the m-th object, respectively. The global object map is the set of all objects $\mathscr{G} = \{\mathbf{v}_m(t)\}_{m=1}^{M}$, which is the output of our system. To generate $\mathscr{G}$, this paper proposes an FL-based dynamic map fusion framework shown in Fig. 2. There are three major components and one communication module as follows.

Edge Vehicles. At the t-th time slot and the k-th vehicle, the raw data (e.g., image, point cloud) is denoted as a vector $\mathbf{d}^{[k]}(t) \in \mathbb{R}^D$, where D is the data dimension. The k-th vehicle fuses the raw data $\mathbf{d}^{[k]}(t)$ into an object list $\{\mathbf{u}_n^{[k]}(t), s_n^{[k]}(t)\}_{n=1}^{N_k}$ via a feature model $\Phi(\cdot|\mathbf{w}_k)$, where N_k is the number of objects, $\mathbf{u}_n^{[k]}(t)$ is the n-th object, $s_n^{[k]}(t)$ is the object score representing the confidence of prediction, and $\mathbf{w}_k \in \mathbb{R}^W$ is the model parameter vector with model size W. The local fusion procedure is written as $\{\mathbf{u}_n^{[k]}(t), s_n^{[k]}(t)\}_{n=1}^{N_k} \leftarrow \Phi\left(\mathbf{d}^{[k]}(t)|\mathbf{w}_k\right)$. Object $\mathbf{u}_n^{[k]}(t)$ has the same format as $\mathbf{v}_m(t)$ in (1), but adopts local coordinate system of vehicle k. If labels $\{\mathbf{b}_n^{[k]}(t)\}$ for the k-th local object lists are available, then parameter $\mathbf{w}_k$ can be locally updated. The vehicle will upload its ego-vehicle location, $\{\mathbf{u}_n^{[k]}(t), s_n^{[k]}(t)\}_{n=1}^{N_k}$ and $\mathbf{w}_k$ to the edge server via V2X communication. This part is shown at the upper left of Fig. 2.

Cloud Server. It performs dataset collection which gathers sensing data in various scenarios using dedicated vehicles, dataset annotation which labels the data manually, and model pretraining which outputs initial model parameters $\mathbf{w}^{[0]} \in \mathbb{R}^W$. The number of samples in each scenario is determined via learning curves [20], [21]. The cloud server will transmit $\mathbf{w}^{[0]}$ to edge server via the Internet and the edge server further broadcasts $\mathbf{w}^{[0]}$ to all vehicles. This part is shown at the upper right of Fig. 2.

Edge Server. It performs three-stage fusion which generates global objects $\mathscr{G}$ by fusing local objects $\{\mathbf{u}_n^{[k]}(t), s_n^{[k]}(t)\}_{n=1}^{N_k}$, federated learning which generates global model parameter vector $\mathbf{g}$ via aggregation of $\{\mathbf{w}_k\}_{k=1}^{K}$, and knowledge distillation which generates training labels $\{\mathbf{b}_n^{[k]}(t)\}$ either via teacher-student distillation [22] (e.g., road side units provide labels) or ensemble distillation [23] (e.g., global fusion provides labels). The edge server will broadcast $\mathscr{G}$, $\mathbf{g}$, and $\{\mathbf{b}_n^{[k]}(t)\}$ to all vehicles via V2X communication. This part is shown at the bottom of Fig. 2.

V2X Communication. There are three data formats being exchanged via V2X: the object maps $\{\mathbf{u}_n^{[k]}(t), s_n^{[k]}(t)\}_{n=1}^{N_k}$ and $\mathscr{G}$ for the purpose of map fusion; the model parameters $\mathbf{g}$, $\mathbf{w}^{[0]}$, $\{\mathbf{w}_k\}_{k=1}^{K}$ for the purpose of federated learning; the labels $\{\mathbf{b}_n^{[k]}(t)\}$ for the purpose of knowledge distillation.

The above dynamic map fusion framework consists of three novel techniques: 1) three-stage fusion which tackles sensing uncertainties; 2) federated learning which tackles model uncertainties; 3) and knowledge distillation which tackles missing data labels. These techniques are summarized in the tables of Algorithm 1, Algorithm 2, and Algorithm 3.

Algorithm 1: Three-Stage Fusion. The first stage partitions the objects in local maps. Most association methods are sensitive to data with heterogeneous densities and hence unsuitable for dynamic map fusion, where the input data could be dense and complicated. We choose the DBSCAN method [24], which can deal with unbalanced clusters and outliers pretty well. The output of this stage is the predicted number of objects M and the association matrices $\{\mathbf{A}^{[k]}(t) \in$

Algorithm 1: Three-Stage Fusion

Input: local maps $\{\mathbf{u}_n^{[k]}(t), s_n^{[k]}(t)\}_{n=1}^{N_k}$
Output: global map $\mathscr{G}$, number of vehicles M, and association matrix $\{\mathbf{A}^{[k]}(t)\}$

$\mathscr{X} = \{\{\mathrm{u}_n^{[1]}(t)\}_{n=1}^{N_1}, \cdots, \{\mathbf{u}_n^{[K]}(t)\}_{n=1}^{N_K}\}$
$\left(M, \{\mathbf{A}^{[k]}(t)\}_{k=1}^{K}\right) \leftarrow \text{DBSCAN}(F_{\text{L}\to\text{G}}(\mathscr{X}))$
for *each global object* $m = 1, \cdots, M$ **do**
  for *each vehicle* $k = 1, \cdots, K$ **do**
    for *each object* $n = 1, \cdots, N_k$ **do**
      $\gamma_n^{[k]}(t) \leftarrow 0$
      if $a_{n,m}^{[k]}(t) = 1$ **then**
        $\gamma_n^{[k]}(t) \leftarrow \frac{\left[1+\exp\left(s_n^{[k]}\right)\right]^{-1}}{\sum_{i=1}^{K}\sum_{j=1}^{N_i} a_{j,m}^{[i]}(t)\left[1+\exp\left(s_j^{[i]}(t)\right)\right]^{-1}}$
      end
    end
  end
  $\mathbf{v}_m(t) \leftarrow \sum_{k=1}^{K}\sum_{n=1}^{N_k} a_{n,m}^{[k]}(t)\gamma_n^{[k]}(t)\mathbf{u}_n^{[k]}(t)$
  $q_m(t) \leftarrow \sum_{k=1}^{K}\sum_{n=1}^{N_k} a_{n,m}^{[k]}(t)\gamma_n^{[k]}(t)s_n^{[k]}(t)$
end
for *each object* $m = 1, \cdots, M$ **do**
  for *each object* $j = 1, \cdots, M$ **do**
    if $IoU(\mathbf{v}_m(t), \mathbf{v}_j(t)) > \delta$ **then**
      remove $\mathbf{v}_m(t)$ if $q_m(t) < q_j(t)$
      remove $\mathbf{v}_j(t)$ if $q_m(t) \geq q_j(t)$
    end
  end
end

$\{0,1\}^{N_k \times M}\}$, where the element at the n-th row and m-th column is denoted as $\{a_{n,m}^{[k]}(t)\}$. If the n-th object in the local map of vehicle k is associated with the m-th object in the global map, then $a_{n,m}^{[k]}(t) = 1$; otherwise $a_{n,m}^{[k]}(t) = 0$. The second stage generates the objects in the global map. To account for the uncertainties at different vehicles, a score-based weighted average method is proposed, which solves the following weighted least squares problem:

$$\min_{\{\mathbf{v}_m(t)\}} \sum_{k=1}^{K}\sum_{n=1}^{N_k} \gamma_n^{[k]}(t) a_{n,m}^{[k]}(t) \left\| \mathbf{v}_m(t) - F_{\text{L}\to\text{G}}\left(\mathbf{u}_n^{[k]}(t)\right) \right\|^2, \quad (2)$$

where the weights $\gamma_n^{[k]}(t)$ are determined by the sigmoid function of prediction scores $\{s_n^k(t)\}$, and $F_{\text{L}\to\text{G}}$ is the function transforming local coordinates to global coordinates (vice versa for $F_{\text{G}\to\text{L}}$). The third stage eliminates overlapped boxes, i.e., $\mathbf{v}_m(t)$ and $\mathbf{v}_j(t)$ with $m \neq j$ may occupy the same space. There are two possible cases: 1) one of the detected objects does not exist; 2) both objects exist, but our predictions of the objects are inaccurate. To accommodate both cases, we first compute the IoUs of all overlapped box groups. If the IoU is larger than a threshold δ, the object with the largest score is reserved and other objects are removed from the map. Otherwise, all the objects are reserved.

Algorithm 2: Federated Learning

Input: raw data $\mathbf{d}^{[k]}(t)$ and labels $\{\mathbf{b}_n^{[k]}(t)\}$ of vehicle k from time $t = T_1$ to $t = T_2$
Output: global model parameter vector $\mathbf{g}$

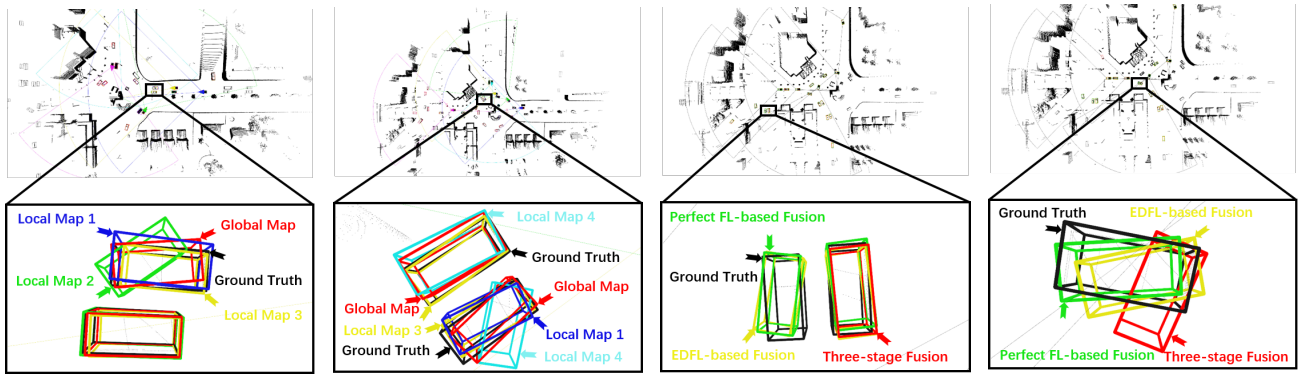
initialize $\mathbf{g}^{[1]} = \mathbf{w}_1^{[1]} = \cdots = \mathbf{w}_K^{[1]} = \mathbf{w}^{[0]}$
for *each round* $i = 1, \cdots, I_{\max}$ **do**
  for *each vehicle* $k = 1, \cdots, K$ ***in parallel*** **do**
    split $\{\mathbf{d}^{[k]}(t), \mathbf{b}_n^{[k]}(t)\}_{t=T_1}^{T_2}$ into batches $\mathscr{B} = \{\mathscr{A}_1, \mathscr{A}_2, \cdots\}$;
    for *each local epoch* $\tau = 1, \cdots, E$ **do**
      for *each batch* $\mathscr{A}_j \in \mathscr{B}$ **do**
        $\mathbf{w}_k^{[i]} \leftarrow \mathbf{w}_k^{[i]} - \varepsilon \sum_{(\mathbf{d}^{[k]}(t), \mathbf{b}_n^{[k]}(t)) \in \mathscr{A}_j} \nabla_{\mathbf{w}_k} \Xi\left(\mathbf{w}_k^{[i]}, \mathbf{d}^{[k]}(t), \mathbf{b}_n^{[k]}(t)\right)$
      end
    end
    return $\mathbf{w}_k^{[i]}$ to the edge server
  end
  server executes $\mathbf{g}^{[i+1]} = \mathbf{w}_1^{[i+1]} = \cdots = \mathbf{w}_K^{[i+1]} = \frac{1}{K}\sum_{k=1}^{K} \mathbf{w}_k^{[i]}$
end

Algorithm 2: Federated Learning. FL aims to find a common model parameter vector such that the total training loss function is minimized:

$$\min_{\mathbf{w}_1 = \cdots = \mathbf{w}_K} \sum_{k=1}^{K}\sum_{t=T_1}^{T_2} \Xi\left(\mathbf{w}_k, \mathbf{d}^{[k]}(t), \mathbf{b}_n^{[k]}(t)\right), \quad (3)$$

where (T_1, T_2) are the starting time and finishing time of training frames, $\Xi(\cdot) = \beta_1 L_{\text{class}} + \beta_2\left(L_{\text{angle}} + L_{\text{box}}\right) + \beta_3 L_{\text{dir}}$, L_{class} is the classification loss related to $\{c_n^{[k]}\}$, L_{angle} is the smooth l_1 function related to $\{\theta_n^{[k]}\}$, L_{box} is the box regression loss function related to $\{x_n^{[k]}, y_n^{[k]}, z_n^{[k]}, l_n^{[k]}, w_n^{[k]}, h_n^{[k]}\}$, L_{dir} the soft max function related to $\{\theta_n^{[k]}\}$ (to distinguish opposite directions), and $(\beta_1, \beta_2, \beta_3)$ are tuning parameters. The training of FL model parameter (i.e., solving (3)) is a distributed and iterative procedure, where each iteration involves two steps: 1) updating the local parameter vectors $(\mathbf{w}_1, \cdots, \mathbf{w}_K)$ using $\left(\mathbf{d}^{[1]}(t), \{\mathbf{b}_n^{[1]}(t)\}, \cdots, \mathbf{d}^{[K]}(t), \{\mathbf{b}_n^{[K]}(t)\}\right)$ at the vehicles $(1, \cdots, K)$, respectively; and 2) computing the global parameter vector $\mathbf{g}$ by aggregating $(\mathbf{w}_1, \cdots, \mathbf{w}_K)$ at the edge server. The entire procedure is stopped until the maximum number of iterations $I_{\max}$ is reached.

Algorithm 3: Knowledge Distillation. KD aims to generate training labels for vehicles in the student set $\mathscr{S}$. The student set is determined as follows: if the vehicle finds that its own local map and the global map differ too much, it will be selected into $\mathscr{S}$. We consider two types of KD: teacher-student distillation and ensemble distillation. For the first type, it is assumed that there exists a teacher model that can produce ground truth labels. For example, some objects are intelligent vehicles that can upload their locations and orientations; or the road side units provide labels in their FoVs. For the second type, the global map, which fuses the



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

Fig. 3: a) and b) An illustration of dynamic maps with and without fusion. Black box: ground truth. Red box: fusion results. Other colored boxes: local detection results. c) and d) An illustration of the three stage fusion, perfect FL-based fusion, and EDFL-based fusion results. Black box: ground truth. Red box: prediction with three stage fusion. Green box: prediction with perfect FL-based fusion. Yellow box: prediction with EDFL-based fusion.

Algorithm 3: Knowledge Distillation

Input: local maps $\{\mathbf{u}_n^{[k]}(t), s_n^{[k]}(t)\}_{n=1}^{N_k}$; global fusion map $\{\mathbf{v}_m\}$; association matrices $\{\mathbf{A}^{[k]}\}$; set of teachers $\mathscr{T}$ and teacher model $\{f_1, \cdots f_{|\mathscr{T}|}\}$; set of students $\mathscr{S}$

Output: labels $\{\mathbf{b}_n^{[k]}(t)\}$ for $k \in \mathscr{S}$

initialize $\mathbf{b}_n^{[k]}(t) = [\,]$
for *each vehicle* $k \in \mathscr{S}$ **do**
  for *each object* $n = 1, \cdots, N_k$ **do**
    for *each teacher* $i \in \mathscr{T}$ **do**
      if *i has label of object n* **then**
        $\mathbf{b}_n^{[k]}(t) \leftarrow f_i(\mathbf{u}_n^{[k]}(t))$ and continue, where f_i is the teacher model
      end
    end
    if $\mathbf{b}_n^{[k]}(t) = [\,]$ **then**
      $\mathbf{b}_n^{[k]}(t) \leftarrow F_{\mathrm{G}\to\mathrm{L}}\left(\sum_{m=1}^{M} a_{n,m}^{[k]} \mathbf{v}_m\right)$
    end
  end
end

information from all local maps, is used to produce training labels. Notice that leveraging the consensus between vehicles (fusion) could potentially add more information, but this will inevitably lead to higher bias (bias-variance trade-off). However, as shown in the experimental results, this type of KD also improves the system performance. The integration of FL and this type of KD is termed ensemble distillation federated learning (EDFL).

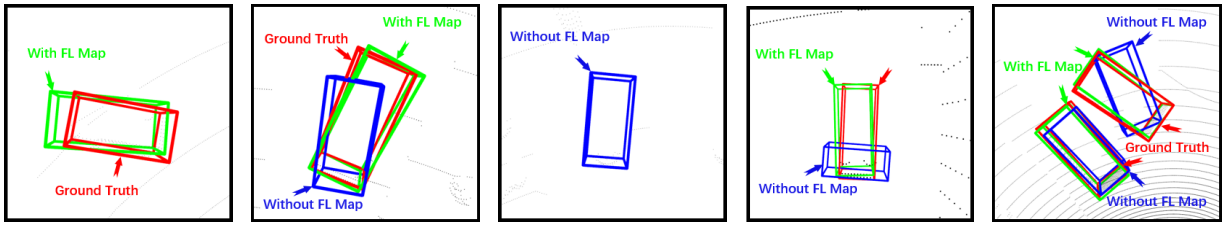
IV. Experimental Results

We employ the CARLA simulation platform to generate training and testing scenarios and multi-agent point cloud datasets for INVS. Each intelligent vehicle (Tesla Model 3) is equipped with a 64-line LiDAR at 20 Hz and a GPS device. The default LiDAR range is set to 100 m, and its FoV is 90° on front. Since the sensing data generated by CARLA is not compatible to the SECOND network [25], [26], we develop a data transformation module such that the generated dataset satisfies the KITTI standard [27].

In the pretraining stage, three intelligent vehicles with different traveling routes are employed to collect sensing data in the 'Town03' map of CARLA. The pre-training dataset includes 60000 frames of point clouds, and 900 frames are chosen for training object feature models. The Adam optimizer is adopted with a learning rate ranging from 10^{-4} to 10^{-6}. The number of epochs is set to 50. After the pretraining stage, the parameters of all intelligent vehicles are set to $\mathbf{w}_1 = \cdots \mathbf{w}_K = \mathbf{w}^{[0]}$.

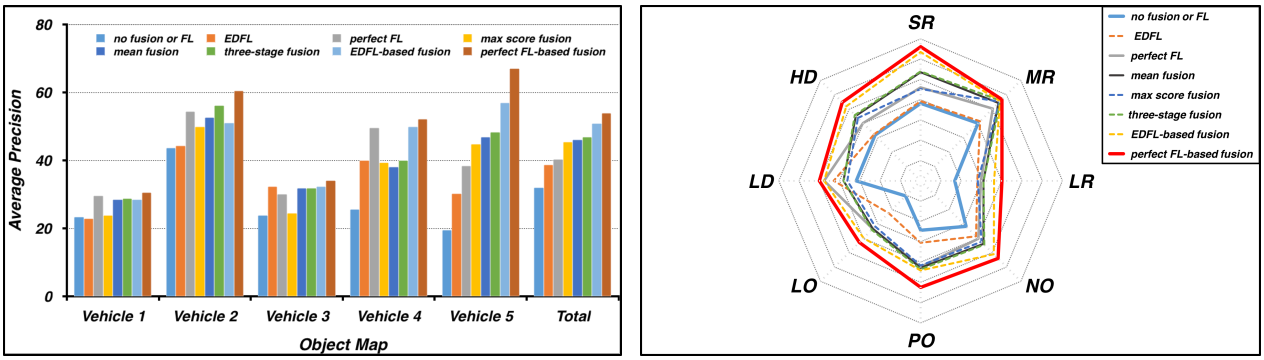
Then we simulate a crossroad traffic scenario in case of $K = 5$ and $M = 37$, that is, 5 intelligent vehicles and 32 ordinary vehicles. The entire scenario lasts for 50.5 seconds and contains 1010 frames. The first 510 frames are used for FL and KD (i.e., $T_1 = 0$ and $T_2 = 25.5$ in problem (3)). The sampling rate is chosen as $3:1$ and hence 170 frames are used for training. The 500 frames from $T_2 = 25.5$ to $T_3 = 50.5$ are used for inference, fusion, and testing. For FL, the number of local updates is set to $E = 2$ and the total number of FL iterations $I_{\max} = 5$. We adopt the same training optimizer and learning rates as the cloud pretraining procedure. The average precision at IoU$= 0.7$ is used for performance evaluation. We consider both perfect FL and EDFL algorithms. For perfect FL algorithm, 170 training samples from T_1 to T_2 are perfectly labeled by a teacher model. For EDFL algorithm, 170 training samples from T_1 to T_2 are not labeled. The edge aggregates 170 frames of the global map via the three-stage fusion. All the vehicles use the global map to obtain their training labels, and the 170 samples with the imperfect labels are used for training.

We develop a set of performance evaluation benchmark metrics for sensing ranges, occlusion rates, and traffic densities, including short range (SR), middle range (MR), long range (LR), no occlusion (NO), partial occlusion (PO),



(a) Vehicle 1 (b) Vehicle 2 (c) Vehicle 3 (d) Vehicle 4 (e) Vehicle 5

Fig. 4: An illustration of dynamic maps with or without FL. Red box: ground truth. Blue box: predictions without FL. Green box: predictions with FL.



(a) Object Maps Average Precision at IoU=0.7 (b) Methods Performance at IoU=0.7

Fig. 5: a) A comparison of different map fusion schemes in INVS at 0.7 IoU for individual vehicles and all vehicles. b) A comparison of map fusion schemes in INVS between the proposed method and other methods in terms of proposed benchmark metrics (SR, MR, LR, NO, PO, LO, LD, HD) at IoU$=0.7$, where the perfect FL-based fusion yields the upper bound of map fusion performance when there are no missing data labels.

large amounts of occlusion (LO), low density(LD) and high density (HD). Objects are classified into SR, MR, and LR according to the object-to-vehicle distances. Objects are classified into NO, PO, and LO according to the number of point clouds reflected from objects within a certain sensing range. A frame is labeled as HD if at least one object in that frame is detected by ≥ 3 intelligent vehicles.

Fig. 3(a) and Fig. 3(b) illustrate the performance of three stage fusion. Fig. 3(a) shows that despite the green vehicle generating false orientations, the global map (i.e., the red box) can correct the orientation error for the green vehicle. Fig. 3(b) shows that despite the blue vehicle missing one object, the global map (i.e., the red box) can recover that object for the blue vehicle. Fig. 4 illustrates the performance of FL without fusion. It can be seen that FL recovers the missing objects, and correct false orientation and false positive detection. Fig. 3(c) and Fig. 3(d) illustrate the performance of FL-based fusion. It can be seen that with imperfect labels, the EDFL-based fusion outperforms the three-stage fusion and achieves performance close to the perfect FL-based fusion. Fig. 5(a) compares the scheme without fusion or FL, the EDFL, the perfect FL, the mean fusion [4], the max score fusion [11], the three-stage fusion, the perfect FL-based fusion, and the EDFL-based fusion for individual vehicles and all vehicles. Compared with max score fusion and mean fusion, the proposed three-stage fusion algorithm achieves the highest precision scores (when IoU$= 0.7$). Compared with the scheme without fusion or FL, the proposed perfect FL and EDFL algorithms can help increase the average precision scores. Compared with all the other methods. the proposed perfect FL-based fusion and the EDFL-based fusion achieve much better performance. Fig. 5(b) shows that the perfect FL-based fusion scheme and the scheme with no fusion or FL achieve the most outer and inner lines, respectively. The proposed three-stage fusion algorithm achieves the best performance for all the benchmark metrics among all fusion methods. The performance of the proposed EDFL-based fusion algorithm is close to that of perfect FL-based fusion, especially for LR and LO metrics.

V. Conclusion

This paper presented an FL-based dynamic map fusion framework, which achieves high quality map fusion and low communication overhead. The FL and KD methodologies were developed to achieve distributive and online feature model updating, as well as to consider common uncertainties that occur in real-world applications. Experimental results based on multi-agent simulations in CARLA demonstrated that even without data labels, the proposed FL-based dynamic map fusion algorithm outperforms other existing methods in terms of the proposed benchmark metrics in INVSs. Such a framework can be further extended to distributed static semantic map fusion.

REFERENCES

[1] N. Lu, N. Cheng, N. Zhang, X. Shen, and J. W. Mark, "Connected vehicles: Solutions and challenges," *IEEE Internet of Things Journal*, vol. 1, no. 4, pp. 289–299, Aug. 2014.

[2] A. Eskandarian, C. Wu, and C. Sun, "Research advances and challenges of autonomous and connected ground vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 2, pp. 683–711, Feb. 2021.

[3] O. Shorinwa, J. Yu, T. Halsted, A. Koufos, and M. Schwager, "Distributed multi-target tracking for autonomous vehicle fleets," in *Proceedings of 2020 IEEE International Conference on Robotics and Automation (ICRA)*, Paris, France, May–Aug. 2020, pp. 3495–3501.

[4] E. Arnold, M. Dianati, R. de Temple, and S. Fallah, "Cooperative perception for 3d object detection in driving scenarios using infrastructure sensors," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–13, 2020.

[5] Q. Chen, X. Ma, S. Tang, J. Guo, Q. Yang, and S. Fu, "F-cooper: feature based cooperative perception for autonomous vehicle edge computing system using 3d point clouds," in *Proceedings of 2019 ACM/IEEE Symposium on Edge Computing*, Arlington, Virginia, Nov. 2019, pp. 88–100.

[6] E. E. Marvasti, A. Raftari, A. E. Marvasti, Y. P. Fallah, R. Guo, and H. Lu, "Cooperative lidar object detection via feature sharing in deep networks," *arxiv preprint arXiv: 2002.08440*, 2020.

[7] T.-H. Wang, S. Manivasagam, M. Liang, B. Yang, W. Zeng, and R. Urtasun, "V2VNet: Vehicle-to-vehicle communication for joint perception and prediction," in *Proceedings of 2020 European Conference on Computer Vision (ECCV)*, Sec, Glasgow, Aug. 2020, pp. 3961–3966.

[8] Z. Xiao, Z. Mo, K. Jiang, and D. Yang, "Multimedia fusion at semantic level in vehicle cooperactive perception," in *Proceedings of 2018 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, San Diego, USA, Jul. 2018, pp. 1–6.

[9] Q. Chen, S. Tang, Q. Yang, and S. Fu, "Cooper: Cooperative perception for connected autonomous vehicles based on 3d point clouds," in *Proceedings of 2019 IEEE International Conference on Distributed Computing Systems (ICDCS)*, Dallas, TX, 2019, pp. 514–524.

[10] A. Miller, K. Rim, P. Chopra, P. Kelkar, and M. Likhachev, "Cooperative perception and localization for cooperative driving," in *Proceedings of 2020 IEEE International Conference on Robotics and Automation (ICRA)*, Paris, France, May–Aug. 2020, pp. 1256–1262.

[11] B. Hurl, R. Cohen, K. Czarnecki, and S. Waslander, "Trupercept: Trust modelling for autonomous vehicle cooperative perception from synthetic data," *arXiv preprint arXiv:1909.07867*, 2019.

[12] M. Ambrosin, I. J. Alvarez, C. Buerkle, L. L. Yang, F. Oboril, M. R. Sastry, and K. Sivanesan, "Object-level perception sharing among connected vehicles," in *Proceedings of 2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, Auckland, New Zealand, Oct. 2019, pp. 1566–1573.

[13] D. D. Yoon, G. M. N. Ali, and B. Ayalew, "Cooperative perception in connected vehicle traffic under field-of-view and participation variations," in *Proceedings of 2019 IEEE Connected and Automated Vehicles Symposium (CAVS)*, Honolulu, HI, Sep. 2019, pp. 1–6.

[14] R. Yee, E. Chan, B. Cheng, and G. Bansal, "Collaborative perception for automated vehicles leveraging vehicle-to-vehicle communications," in *Proceedings of 2018 IEEE Intelligent Vehicles Symposium (IV)*, Changshu, China, Jun. 2018, pp. 1099–1106.

[15] Z. Y. Rawashdeh and Z. Wang, "Collaborative automated driving: A machine learning-based method to enhance the accuracy of shared information," in *Proceedings of 2018 International Conference on Intelligent Transportation Systems (ITSC)*, Maui, HI, Nov. 2018, pp. 3961–3966.

[16] 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 Robotics and Automation Letters*, vol. 5, no. 2, pp. 3509–3516, Apr. 2020.

[17] B. Liu, L. Wang, and M. Liu, "Lifelong federated reinforcement learning: A learning architecture for navigation in cloud robotic systems," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4555–4562, Oct. 2019.

[18] W. Zhuang, Y. Wen, X. Zhang, X. Gan, D. Yin, D. Zhou, S. Zhang, and S. Yi, "Performance optimization for federated person re-identification via benchmark analysis," *arXiv preprint arXiv:2008.11560*, 2020.

[19] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Proceedings of the 1st Annual Conference on Robot Learning (CoRL)*, Mountain View, CA, Oct. 2017, pp. 1–16.

[20] S. Wang, R. Wang, Q. Hao, , Y.-C. Wu, and H. V. Poor, "Learning centric power allocation for edge intelligence," in *Proceedings of 2020 IEEE International Conference on Communications (ICC)*, Dublin, Ireland, Jun. 2020, pp. 1–6.

[21] Y. Sun, D. Li, X. Wu, and Q. Hao, "Visual perception based situation analysis of traffic scenes for autonomous driving applications," in *2020 IEEE International Conference on Intelligent Transportation Systems (ITSC)*, Rhodes, Greece, Jul. 2020, pp. 1–7.

[22] Q. Guo, X. Wang, Y. Wu, Z. Yu, D. Liang, X. Hu, and P. Luo, "Online knowledge distillation via collaborative learning," in *Proceedings of 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, Jun. 2020, pp. 11 017–11 029.

[23] T. Lin, L. Kong, S. U. Stich, and M. Jaggi, "Ensemble distillation for robust model fusion in federated learning," in *Proceedings of 2020 Conference on Neural Information Processing Systems (NeurIPS)*, Vancouver, Canada, Dec. 2020, pp. 1–25.

[24] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of 1996 International Conference on Knowledge Discovery and Data Mining (KDD)*, Portland, Oregon, 1996, pp. 226–231.

[25] Y. Yan, Y. Mao, and B. Li, "Second: Sparsely embedded convolutional detection," *Sensors*, vol. 18, no. 10, p. 3337, 2018.

[26] S. Shi, Z. Wang, J. Shi, X. Wang, and H. Li, "From points to parts: 3d object detection from point cloud with part-aware and part-aggregation network," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–17, 2020.

[27] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.