Boosting Collaborative Vehicular Perception on the Edge with Vehicle-to-Vehicle Communication

Ruiyang Zhu
University of Michigan
ryanzhu@umich.edu

Xiao Zhu*
University of Michigan
shawnzhu@umich.edu

Anlan Zhang
University of Southern California
anlanzha@usc.edu

Xumiao Zhang
University of Michigan
xumiao@umich.edu

Jiachen Sun
University of Michigan
jiachens@umich.edu

Feng Qian
University of Southern California
fengqian@usc.edu

Hang Qiu
University of California, Riverside
hangq@ucr.edu

Z. Morley Mao
University of Michigan
zmao@umich.edu

Myungjin Lee
Cisco Research
myungjle@cisco.com

Abstract

Collaborative Vehicular Perception (CVP) enables connected and autonomous vehicles (CAVs) to cooperatively extend their views through wirelessly sharing their sensor data. Existing CVP systems employ either a vehicle-to-vehicle (V2V) or vehicle-to-infrastructure (V2I) view exchange paradigm. In this paper, we advocate a hybrid CVP design: our developed system, Harbor, employs V2I as its fundamental underlying framework, and *opportunistically* employs V2V to boost the performance. In Harbor, vehicles (*helpers*) may serve as relays to assist other vehicles (*helpees*) in reaching an edge node, which performs sensor data merging to produce the extended view. We judiciously partition the workload between the edge and vehicles, develop a robust helper-helpee assignment model, and solve it efficiently at runtime. We conduct both real-world tests and large-scale emulation experiments using two prevailing CAV applications: drivable space detection and object detection. Our real-world evaluation conducted at one of the world's first purpose-built autonomous driving testbeds demonstrates that Harbor outperforms state-of-the-art V2V- or V2I-only CVP schemes by up to 36% in detection accuracy, resulting in significantly fewer collisions under dangerous driving scenarios.

CCS Concepts

• **Networks** → **Network protocol design**; **Cyber-physical networks**; • **Applied computing** → **Transportation**.

Keywords

Cooperative Vehicular Sensing, Vehicular Networks, Autonomous Cars, LiDAR

*Now at Google.

SENSYS '24, November 4–7, 2024, Hangzhou, China

ACM ISBN 979-8-4007-0697-4/24/11
https://doi.org/10.1145/3666025.3699328

ACM Reference Format:
Ruiyang Zhu, Xiao Zhu, Anlan Zhang, Xumiao Zhang, Jiachen Sun, Feng Qian, Hang Qiu, Z. Morley Mao, and Myungjin Lee. 2024. Boosting Collaborative Vehicular Perception on the Edge with Vehicle-to-Vehicle Communication. In *The 22nd ACM Conference on Embedded Networked Sensor Systems (SENSYS '24), November 4–7, 2024, Hangzhou, China.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3666025.3699328

1 Introduction

Connected and autonomous vehicles (CAVs) use various on-board 3D sensors such as LiDAR [19] and stereo cameras [22] to perceive the environment [6, 7]. However, a single vehicle's sensing range is limited, and its view can be obstructed by obstacles [33, 40, 56]. To overcome these limitations, Collaborative Vehicular Perception (CVP) [63, 64, 87, 88] enables multiple vehicles to cooperatively extend their views through wirelessly sharing their sensor data, as powered by the increasingly mature onboard wireless infrastructure [1, 3, 4]. CVP can boost a wide range of autonomous driving and Advanced Driving Assistance Systems (ADAS) such as drivable space detection [39, 62] and object detection [50, 68, 83].

Early works of CVP only involve two vehicles sharing their views, relying on a simple vehicle-to-vehicle (V2V) view exchange paradigm [32, 63]. As researchers realize the importance of larger-scale view sharing, they incorporate edge nodes into CVP systems, which combine all the views uploaded by participating vehicles and distribute the processed results back to the vehicles [56, 67, 88]. This centralized vehicle-to-infrastructure (V2I) paradigm is more scalable from the computation perspective: a powerful cloud server can efficiently merge multiple vehicles' views in real-time. The challenges, however, stem from the unreliable wireless communication. Take the most promising cellular access for V2I as an example, from the most recent FCC cellular coverage report [13] and National Highway Traffic Safety Administration's (NHTSA) fatality analysis reports [8], the average coverage of major U.S. carriers ranges from 38.5% to 71.6% (rural and urban combined). Another recent study [52] suggests that "5-bar" 5G connectivity is statistically more unreliable than weaker signal strength in densely populated urban areas. Our measurement in a major U.S. city also confirms that onboard cellular (uplink) connectivity is highly heterogeneous. Vehicles may experience very poor performance during urban driving (*e.g.,* 8 to 20 seconds of low uplink bandwidth less than 1 Mbps,

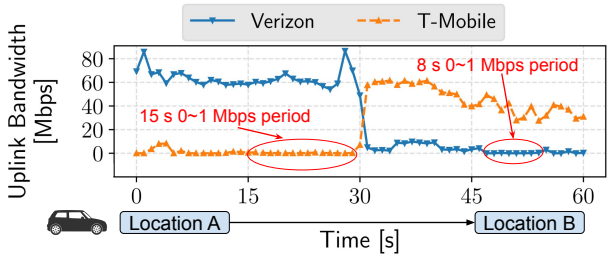

Figure 1: An example of cellular uplink heterogeneity, where the uplink bandwidth of two carriers is recorded simultaneously at the same place while in urban driving.

see Figure 1). Such heterogeneity and unreliability are *inherent* due to vehicles' mobility, which incurs frequent handovers [41], fluctuating channel quality, and continual interference.

In this paper, we advocate a hybrid design by leveraging the synergy between V2I and V2V. We instantiate the design into Harbor, a full-fledged CVP framework. Harbor adopts V2I as its fundamental underlying framework, and *opportunistically* employs V2V to boost the performance. As shown in Figure 2, when some vehicles (called *helpees*) have poor V2I connectivity, other vehicles (called *helpers*) can relay their sensor data to the edge. The edge will then merge the views and send processed results (*e.g.,* detected vehicles) back to the vehicles. Participation in Harbor can be incentivized through enhanced safety and CAV features, conceptually similar to the recent commercially deployed peer-to-peer CDN (PCDN)[1] infrastructure [79]. Nevertheless, realizing this seemingly straightforward idea involves making several non-trivial design decisions, as elaborated next.

Partitioning the Workload between the Edge and Vehicles. There exist many V2X works that bridge V2V and V2I links [59, 71, 74, 76]. Harbor fundamentally differs from these approaches by not only focusing on improving *network* connectivity but also addressing both computation (point cloud merging and detection) and communication. For communication, as described before, helpers relay traffic for helpees. For computation, we may also let some helpers merge point clouds for helpees. This design, however, significantly increases the system complexity and enlarges the attack surface (in contrast, a helpee–helper–edge communication path can be end-to-end encrypted). Therefore, in Harbor, only the edge is responsible for merging vehicles' views. A vehicle can only perform *local merging*: merge its own view with the results returned by the edge without further distributing the merged view to downstream. The vehicle-side merge can occur when the vehicle's uploaded view misses the edge-side merging deadline (to be detailed soon). This design simplifies Harbor by reducing the solution space and avoiding reconciling the helpee–helper paths of computation and communication.

Efficient Helper-helpee Assignment. Even with simplified computation flows, making helper-helpee assignments needs to consider three types of dynamics: V2V network connectivity, V2I network connectivity, and vehicle mobility. We formulate helper-helpee assignment as an optimization problem taking into account

[1]Internet users contribute their under-utilized edge devices as "mini" CDN nodes, which work in conjunction with traditional centralized CDN nodes.

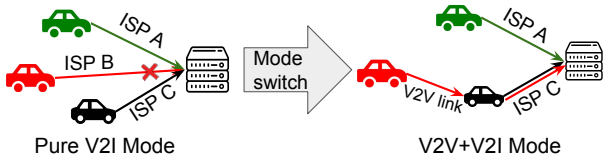

Figure 2: An example of jointly using V2V and V2I to bridge a disconnected/poor-performing car (red).

the above dynamics. In our model, a vehicle only sends its lightweight state (location, V2I bandwidth measurements, *etc.*) to the edge, either directly or via some helper.[2] A challenge here is how to measure the V2V bandwidth. A naïve approach to sending active probes is not scalable due to its quadratic complexity. Instead, Harbor introduces a novel approach that uses robust analytical models of V2V distance and wireless interference to implicitly approximate the V2V bandwidth, without actually probing it. This leads to an ultra-lightweight control plane. While it is difficult to precisely solve the above formulation at runtime, we develop a polynomial-time approximation algorithm based on efficient weighted bipartite matching. This enables Harbor to frequently update helper-helpee pairs under dynamic, complex traffic scenarios.

Optimizations to Further Improve Harbor**'s Performance.** We introduce two system-level mechanisms to boost Harbor's performance. On the edge side, the edge server enforces a deadline to determine when to start the data merging and perception tasks. It ensures timely delivery of perception results to the vehicles. On the vehicle side, the same wireless V2V medium is used to carry both the point cloud (uplink) and returned perception results (downlink). Vehicles properly prioritize between the two types of traffic at the MAC layer to tackle their unbalanced network resource usage.

Scalable and Lightweight Design. Harbor is designed to be both lightweight and scalable. This is achieved through leveraging V2V to assist V2I in an *opportunistic* manner. By utilizing spare V2I resources from vehicles through V2V, Harbor provides *lower end-to-end latency* compared to using V2I alone. Additionally, the lightweight control plane design and polynomial-time approximation optimization for helper-helpee assignment, ensures that the system can make real-time decisions with *low computation overhead*. These design choices make Harbor well-suited for complex traffic scenarios with dynamic network conditions.

We integrate the above components into Harbor, a distributed CVP framework that can support a wide range of CAV applications. We comprehensively evaluate its performance through real-world driving tests and emulations over real cellular networks, using two predominant CAV applications: drivable space detection [62] and object detection [50]. We highlight the key results as follows.

- Through emulations on various traffic densities, Harbor reduces end-to-end drivable space detection and object detection latency by 18% to 57% compared to using only V2V or V2I, without sacrificing the detection accuracy.
- We deploy Harbor and conduct driving tests at a real-world CAV testbed, *Mcity* [25]. The results reveal *abundant opportunities of leveraging V2V in practice*: Harbor improves the object detection

[2]The state information (control plane) is much lighter than the point cloud data (data plane). We thus assume the former can be delivered to the edge (otherwise the vehicle will fall back to offline mode).

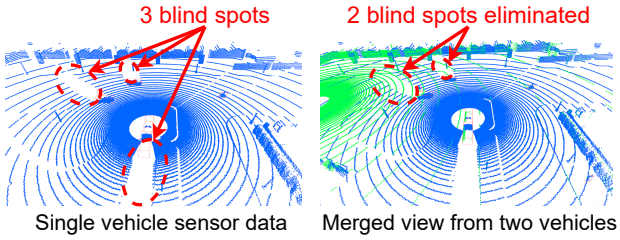

Figure 3: An example showing the benefits of CVP by merging a nearby vehicle's data (green) to ego vehicle (blue).

and drivable space detection accuracy by 12% on average (up to 36%) compared to pure V2V- or V2I-based CVP.

• We conduct a larger emulation experiment consisting of 20 to 100 vehicles and demonstrate that Harbor outperforms the V2V- or V2I-only baselines by an average of 43% and 14% in detection latency and accuracy under fast-changing traffic topology, showcasing superior scalability and performance.

Despite the encouraging results above, Harbor does bear several limitations (§7). We thus do not claim any "optimality" of our solution (which is also difficult to define given the highly dynamic, multi-objective, and multi-stakeholder nature of CVP). Instead, we explore the design space of fusing V2V into the V2I-based CVP design. Through real-world deployment, we demonstrate that Harbor, a first-of-its-kind hybrid CVP architecture, is practical, beneficial, and scalable. This research does not raise any ethical issues.

2 Background and Motivation

Autonomous vehicles rely on various on-board 3D vision sensors to understand the physical world consisting of road segments, pedestrians, other vehicles, and more, to make driving decisions. For example, LiDAR [19] is a major on-board vision sensor, which fires laser lights at different angles and measures how long it takes for the lights to return to the sensor after reflection from objects. Based on this, LiDAR calculates the distance of these objects and produces 3D point clouds to represent the surrounding environment. Different software modules will further process the collected point cloud data and make appropriate driving decisions.

Collaborative Vehicular Perception. As mentioned in §1, collaborative vehicular perception (CVP) has the potential to benefit autonomous driving by extending its view by merging sensing information from different vehicles. Figure 3 shows an example that visualizes point cloud data generated from the state-of-the-art autonomous driving simulator, CARLA [11]. The blue points represent a single vehicle's data while the green ones represent data from a nearby vehicle. As illustrated, merging the two point clouds eliminates two of the three blind spots.

Sharing raw sensor data *vs.* sharing processed data. Existing CVP schemes can be grouped into three main categories based on the stage of data sharing: (1) Raw-data sharing schemes [49, 64, 87, 88] share the original raw sensor data; (2) Feature-level sharing schemes [31, 35, 78, 81, 84] send intermediate features of perception; (3) Object-level sharing schemes [54, 67, 69] directly share lightweight perception results such as object bounding boxes. In this work, we build Harbor based on sharing raw sensor data (point clouds) instead of detection results or processed features for two reasons. *First*, sharing raw data preserves the high level of detail provided by sensors. Based on the state-of-the-art work [53, 77] in computer vision, raw sensor data sharing achieves higher perception accuracy compared to sharing detection results or intermediate features. Various efforts [64, 87, 88] have made raw data sharing more scalable and practical. *Second*, sharing raw data is more versatile. Vehicles that receive processed features are constrained to use the same format and application for detection. In contrast, raw sensor data can be used by a wide range of CAV applications for multiple purposes. Although Harbor currently focuses on point-cloud-level collaboration, the same design principles can be applied to merge intermediate features or detection results (§6).

Limitations of V2V. It has been shown that V2V-based CVP is not scalable [88]. This is because in a V2V-based CVP, vehicles exchange sensor data streams for sharing [63]; depending on the design, either all vehicles receive all the point cloud data from other vehicles and perform collaborative perception, or only one of the vehicles receives data, performs perception, and disseminates the perception results. In either way, a V2V system would require a shared wireless network with sufficient capacity established by vehicles. Traditional WiFi standards (802.11 ac/n) fall short in supporting highly variable wireless channels under high mobility [64]. While WiFi standards like 802.11ad/ay offer Gbps bandwidth [70], they suffer from small coverage and poor mobility performance. Long-distance wireless communication technologies such as DSRC/802.11p [47] can achieve only ~6 Mbps, inadequate for accommodating a large number of vehicles.

Limitations of V2I. V2I-based CVP leverages a powerful edge server for view merging and perception. Vehicles upload their sensor data to the server without direct communications among them [88]. This allows for more network and compute resources to support large-scale CVP. However, V2I communications are not always reliable: even the most ubiquitous cellular networks still have coverage issues [13]. Based on the LTE coverage map from FCC [13] and the National Highway Traffic Safety Administration's Fatality Analysis Reporting System [8], the LTE coverage on the roads of 5 U.S. states with the largest number of traffic fatalities in 2021 is only 68.5%, 71.6%, and 38.5%, for *AT&T*, *Verizon*, and *T-Mobile*, respectively. Our driving tests in a major U.S. city also confirm that onboard cellular connectivity is fluctuating and highly heterogeneous, as described in §1 (Figure 1).

Joint Use of V2V and V2I. The limitations of existing V2V-only [32, 49, 63] or V2I-only [55, 56, 88] CVP architecture motivate us to explore a joint use of both. Harbor differs from existing work in innovating the network communication paradigm: it adopts a hybrid approach that combines the V2V and V2I communication for CVP application. In Harbor, V2V communication is used to bridge vehicles with poor V2I connectivity to participant in collaboration, thereby enhancing CVP performance and robustness across different network conditions.

3 Harbor Design

The observations in §2 motivate a hybrid CVP architecture that leverages the synergy between V2I and V2V. In this work, we aim to explore the benefits of leveraging V2V communication to enhance the performance of V2I-centric CVP systems. Our primary research question focuses on how the integration of V2V and V2I together can improve CVP tasks, particularly in terms of perception accuracy,

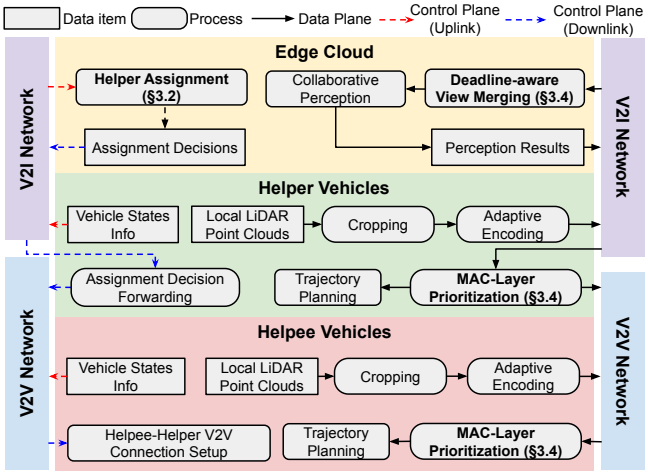

Figure 4: System architecture of Harbor. The control plane and data plane run in parallel and do not block each other.

detection latency, and system scalability. The resulting hybrid CVP system, Harbor, adopts V2I as its basic underlying framework, and *opportunistically* employs V2V to boost the performance. It enables multiple vehicles to leverage their V2V and/or V2I network access to share 3D sensor data. Harbor aims to reduce CVP latency and improve accuracy by combining V2V and V2I communications in an efficient and scalable manner, especially under dynamic wireless network and mobility conditions.

3.1 Hybrid System Architecture

Figure 4 shows the system architecture of Harbor. At any specific time, vehicles fall into two disjoint sets: a helper set and a helpee set. Vehicles in the helper set have good V2I connectivity, whereas vehicles in the helpee set do not. Helpers and helpees establish a V2V network so that helpees can leverage V2V to send their point clouds to the server through the V2I links of helpers, who also upload their own point clouds.

Assumptions made by Harbor**.** There are three assumptions for the designed system Harbor: (1) Collaborative vehicular perception is initiated only for vehicles that are in close proximity, therefore vehicles are connected to the same edge server during the collaboration period; (2) Vehicles exchange control plane messages periodically to the edge server for making *helper-helpee assignment* decisions; (3) The V2V network is used only for relaying data back/forth from the edge server to helpee vehicles. These assumptions are typically common and practical in collaborative vehicular perception systems [63, 64, 88] and other vehicular systems [59, 74].

Data Plane Operations. As shown in Figure 4, before uploading each point cloud frame, cropping and adaptive encoding [88] are performed to reduce the data size for transmission. On receiving a round of point clouds from different vehicles[3], the server merges these frames based on their locations (described shortly) and performs perception tasks such as drivable space detection [39, 62] and object detection [50, 68, 83] to generate detection results. Finally, the detection result is sent back to the vehicles. Note that in parallel to the above procedure, vehicles still run local detection. This allows a vehicle to fall back to using the local detection if remote results are not available.

Control Plane Message Exchanges. Harbor needs to strategically assign helpers to helpees (§3.2). A naïve way is to let each helpee find a helper in a distributed manner. However, distributed pairing lacks a holistic view of the network topology and V2I/V2V resources in the system. Harbor opts for a centralized method, where the server collects vehicle state information and makes assignment decisions accordingly. Dashed lines in Figure 4 show the control plane message exchanges in Harbor. Helpers periodically send their states to the server, including locations, V2I bandwidth measurements, and V2V network routing tables. Similarly, helpees broadcast their state information periodically using the V2V network and rely on helpers to forward it to the server. Such state information is lightweight in nature and can be delivered to the edge server efficiently, as validated in §5.5 and other existing CVP systems [63, 88]. The server then computes the best assignment (§3.2) and informs the concerned helpers. Upon receiving the notifications, each helper establishes a V2V connection with the assigned helpee for point cloud forwarding. Due to vehicles' mobility, Harbor needs to update helper assignments dynamically. Harbor server periodically computes the best assignment every T_p and updates the assignment to vehicles in the system.

Mobility/Handover Management. Inline with previous V2I-based CVP [88], we envision that each edge node serves a predefined geographic area (similar to the service area of a cellular base station). Optionally, an edge's service area can be divided into *subareas*. For example, for an edge node deployed at a major intersection, each subarea may correspond to one of the four road segments. Subareas can be overlapped. Vehicles' collaboration and edge-side merging are confined within a subarea. This helps reduce the search space of helper assignments and the edge's merging overhead. When a vehicle (either a helper or helpee) moves across the subarea boundary, its potential helper or helpee(s) will be immediately switched to those in the new subarea. The fast switch is owning to the *stateless* nature of helper assignment – assignment history is not considered when making assignment decisions. This makes Harbor adaptive to the volatile traffic dynamics. Our large-scale evaluation with up to 100 emulated vehicles (§5.4) performs such a subarea partition based on the *road_id* information of the map. A handover procedure similar to the above *intra-edge* handover can be applied to *inter-edge* handovers. We leave the detailed inter-edge handover design as our future work.

3.2 Modeling Helper Assignment

Efficient assignment of helpers to helpees is a crucial challenge in Harbor due to the dynamics of V2V and V2I links as well as the interplay among helpers and helpees. Our goal is to find an assignment that pairs helpees to helpers to maximize the overall system performance, specifically by minimizing all vehicles' point cloud upload time to deliver the sensor data to the edge server. Also, assisting helpees should bring minimal impact on helpers' own sensor data upload. As our goal is to pair helpees (V2V sources) with helpers (V2V destinations), our design is orthogonal to and

[3]Like existing work [64, 88], we assume point cloud capture on different vehicles are synchronized. In reality, point cloud data generation time across vehicles can have slight misalignment, which can be solved by [42, 87].

compatible with the underlying routing scheme (*e.g.,* [30, 34]) used in the V2V network.

Harbor performs helper assignment in two steps. First, it determines the *set* of helpers and helpees; second, it finds for each helpee its helper (if available). While ideally the two steps could be combined in a holistic optimization framework, we separate them to reduce the overall complexity of assignment search, which needs to be performed at a high frequency (every $T_p = 100$ ms in our implementation) to tackle the volatile traffic dynamics. In the first step, Harbor applies a simple heuristic: regard vehicles whose V2I uplink bandwidth is lower than (at least) a threshold as helpees (helpers). The threshold, which is empirically set to 1 Mbps [13], ensures that a helper has minimal bandwidth to upload its own sensor data before helping others.

Next, given the selected helpers and helpees, we consider how to search for the assignments. A prerequisite here is the V2V bandwidth. As mentioned in §1, actively probing every V2V link is not scalable. We thus introduce a novel approach that uses robust analytical models of V2V distance and wireless interference to implicitly approximate the V2V bandwidth, without actually probing it. This leads to a lightweight optimization model (elaborated next). To begin with, we identify three key factors that impact the performance of an end-to-end (E2E) path (which consists of V2I and possibly V2V links).

• **Physical distance.** In a V2V wireless network, the physical distance between a helpee and a helper can affect the throughput of the helpee-to-helper path [85].

• **V2V network interference.** Scenarios like multi-hoping between helper and helpee and multiple helpees transmitting data to the same helper may incur interferences [45, 61] and thus hurt the performance of a helpee-helper connection.

• **V2I network bandwidth.** The V2I bandwidth of a helpee's helper can also impact the sensor data upload time, especially when the helper-to-server path becomes the bottleneck of the E2E path. Note the V2I bandwidth of a helper is shared by itself and potentially one or more helpees.

It is important to note that the goal of our analytical model is not to *precisely estimate V2V bandwidth* for each helper-helpee vehicle pair, but to select a helper vehicle that has potentially better V2V performance compared to other helper vehicles. To achieve this, we leverage V2V distance and interference as two primary factors, both of which have been shown to strongly correlate with wireless network bandwidth [61, 85]. Additionally, given that the effective range of CVP systems is generally smaller (around 200m×100m [82]) compared to large-area mesh networks [27], the use of distance and interference as bandwidth estimators is both practical and effective, as validated in our experiments (§5.5).

Assuming the total number of helpees is $m = |E|$, an assignment $A = \{a_1, a_2, ..., a_m\}$ contains one or multiple assignment pairs from helpee e_i to its helper r_j (Table 1). For an assignment pair a_k, its score is defined as Equation 1a. The distance and interference scores quantify how good the assignment pair is. To obtain scores of A, we sum each pair's distance and interference score and then aggregate them using an aggregation function f (Equation 1b). While different aggregation functions can be applied, Harbor uses the harmonic mean to aggregate scores of different assignment pairs. The use of the harmonic mean helps filter out assignments with low-score pairs to prevent the entire collaboration system from being slowed down by a single "slow" vehicle, as it is very sensitive to small-value outliers.

Table 1: Notations for helper assignment algorithm.

Symbol	Meaning
$E = \{e_1, ..., e_{\|E\|}\}$	Helpee vehicle set: helpee 1 to helpee $\|E\|$
$R = \{r_1, ..., r_{\|R\|}\}$	Helper vehicle set: helper 1 to helper $\|R\|$
$a_k = (e_i, r_j)$	A (helpee, helper) assignment pair
$N(r_j)$	Helping capacity of helper j
$A = \{a_1, a_2, ..., a_m\}$	An assignment is a set of assignment pairs ($m \leq \|E\|$)
$S_{dist}(A)$	Distance score of assignment A
$S_{intf}(A)$	V2V interference score of assignment A
$D(e_i, r_j)$	Physical distance between helpee i and helper j
$P(a_k) = e_i \rightarrow ... \rightarrow r_j$	The network path for a_k (helpee i to helper j)
$IC(v_i, X)$	Interference count produced by vertices in graph X to vertex v_i
$IC(P(a_k), X)$	The sum of interference counts for all vertices in path $P(a_k)$
$\mathcal{G}(A)$	Graph produced by active transmitting and receiving nodes in assignment A

$$Score(a_k) = S_{dist}(a_k) + S_{intf}(a_k) \tag{1a}$$

$$Score(A) = f(Score(a_1), ..., Score(a_m)) \tag{1b}$$

$$\textbf{maximize } Score(A) \textbf{ s.t. } C_{bw}(a) \geq 0, \forall\, a \in A \tag{1c}$$

The objective of finding the best assignment A is defined as Equation 1c, and the assignment is subject to the bandwidth constraint C_{bw}. The intuition behind the constraint is that Harbor tries to leverage the spare V2I bandwidth from helpers to help helpees upload sensor data without hurting helpers' own performance. In cases where helpers are not capable of helping all helpees, *i.e.,* the constraint of the above optimization cannot be satisfied (*e.g.,* the helpers' V2I bandwidth is limited for helping all the helpees), Harbor schedules the assignment so that as many possible helpees are helped as possible. As Harbor updates assignments frequently (§3.1), unassigned helpees may be assigned later when there are more helpers or helpers have better V2I bandwidth.

We next describe how to calculate scores and constraints for each individual factor. Since distance and interference have different units and scales, we design each score function (S_{dist}, S_{intf}) to map the corresponding factor to a value within the range of $[0, 1]$ to avoid a single factor dominating the entire score. Table 1 summarizes notations used to calculate different scores of an assignment.

• **Distance Score.** Intuitively, the wireless throughput is better when two nodes physically reside closer to each other. Therefore, the assignment should prefer a helper closer to a helpee. The distance score for a pair $a_k = (e_i, r_j)$ is quantified by comparing the physical distance between e_i and r_j with the longest possible distance between e_i and any other helper in the system. Formally, for an assignment pair $a_k = (e_i, r_j)$, its distance score is defined as $S_{dist}(a_k) = 1 - \frac{D(e_i, r_j)}{max\{D(e_i, r_k)\}_{k=1}^{|R|}}$. This equation shows that a smaller physical distance between e_i and r_j leads to a higher score, indicating a nearby node is preferred for better V2V bandwidth. Also, the score value lies between 0 and 1, as desired. With $S_{dist}(a_k)$ for all a_k in A, the distance score for the assignment can be calculated using the aggregation function, as described in Equation 1b.

• **Interference Score.** The interference score is designed so that a higher score is generated when the assignment creates less interference. To fulfill this, we propose a graph-based method to

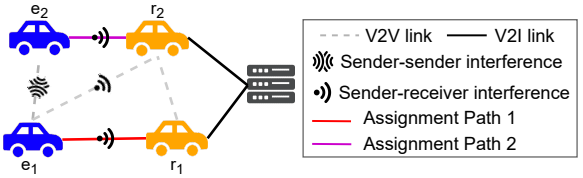

Figure 5: An example of graph-based interference score calculation (no link between e_2 and r_1 due to coverage).

quantify the level of interference by leveraging an observation: interference will occur between active senders and from senders to receivers when they are in each other's coverage range [45]. In Harbor, an active sender is a node that transmits bulk sensor data to another node through V2V, *i.e.,* helpees that send data to their helpers. For instance, in Figure 5, e_2, e_1 are active senders while r_1, r_2 are receivers and use V2I to send received sensor data to the server.

To quantify the interference, we define `Interference Count`, IC. At a high level, $IC(v_i, X)$ measures the number of interfering nodes (*i.e.,* in-coverage active senders) from the network topology X to a node vertex v_i, and $IC(a_k, X)$ measures the interference produced by X to a pair a_k. To calculate IC for a_k, we sum up the interference count for all nodes in the network path $P(a_k)$ of pair a_k, *i.e.,* $IC(a_k, X) = \Sigma_{v_i \in P(a_k)} IC(v_i, X)$. The entire graph can be constructed from control messages received by the server, which include the routing tables on vehicles (§3.1). The interference score function maps the interference count for a_k to a value between 0 and 1. The formula to calculate the interference score for a_k is $S_{intf}(a_k) = 1 - \frac{IC(a_k, \mathcal{G}(A)) - IC(a_k, \mathcal{G}(\{a_k\}))}{IC(a_k, G_0) - IC(a_k, \mathcal{G}(\{a_k\}))}$. G_0 represents the graph formed by assuming all nodes are actively generating/receiving network traffic. $IC(a_k, \mathcal{G}(\{a_k\}))$ calculates the interference caused only by the nodes in the network path of a_k. Similarly, $IC(a_k, \mathcal{G}(A))$ is the interference caused by the actual assignment A and $IC(a_k, G_0)$ is the maximum possible interference for the pair a_k. Take Figure 5 as an example, there are two connections in the assignment $A = \{a_1, a_2\}$, where $a_1 = (e_1, r_1)$ and $a_2 = (e_2, r_2)$. Their network paths are $p_1 = P(a_1) = e_1 \rightarrow r_1$ and $p_2 = P(a_2) = e_2 \rightarrow r_2$. To determine $IC(a_2, \mathcal{G}(A))$, we sum up the interference sources for all nodes in the network path p_2, *i.e.,* $IC(a_2, \mathcal{G}(A)) = \Sigma_{v_i \in p_2} IC(v_i, \mathcal{G}(A)) = IC(e_2, \mathcal{G}(A)) + IC(r_2, \mathcal{G}(A)) = |\{e_1\}| + |\{e_1, e_2\}| = 1 + 2 = 3$. Note r_1 is not counted when calculating $IC(r_2, A)$ because r_1 uses V2I for sending sensor data to the server and does not generate bulk sensor data transmission in the V2V network. Similarly, we can calculate $IC(a_2, \mathcal{G}(\{a_2\})) = 1$ and $IC(a_2, G_0) = 6$. Finally we have the interference score $S_{intf}(a_2) = 1 - \frac{3-1}{6-1} = 0.6$. Similarly, $IC(a_1, A)$ can be calculated and the interference score $S_{intf}(A)$ can be obtained. With both assignment pairs' scores calculated, the score for A can be obtained from Equation 1b.

• **Bandwidth Constraint.** A higher V2I bandwidth is more desired as it can potentially reduce the data upload time. In Equation 1c, a V2I bandwidth constraint C_{bw} is reinforced for each assignment pair. The bandwidth constraint ensures the number of helpees a helper helps does not exceed its "capability". To quantify this, we calculate the V2I helping capacity of each helper $N(r_j)$ based on the average bandwidth required for transmitting the encoded point clouds. The capacity is calculated as $N(r_j) = \lfloor \frac{BW(r_j) - BW_{req}}{BW_{req}} \rfloor$. $N(r_j)$ quantifies how many helpees the helper r_j can help. We set $BW_{req} = 4.8$ Mbps based on the bandwidth requirement for transmitting point cloud at 10 fps after encoding with high resolution [12, 88]. Then we can derive the equation for all the assignment pair a: $C_{bw}(a) = N(r_a) - \sum_{k=1}^{m} (\pi_2(a_k) = r_a) \geq 0$, where r_a is the helper in a and $\pi_2(a_k)$ selects the helper (second element) in the ordered assignment pair a_k.

3.3 Efficient Helper Assignment Algorithm

Intuitively, the assignment with the highest score can be obtained from a brute-force search. However, this method is not quite scalable as the number of possible assignments grows exponentially as $|E|$ (the number of helpees) increases, with a complexity of $O(|R|^{|E|})$. In order to handle real-time dynamics for more vehicles, we propose an efficient optimization to solve the best matching in polynomial time.

We transform the helper-helpee matching problem in Harbor to a weighted bipartite b-matching (`WBbM`) problem [28]. Consider the helper set and helpee set as the two matching sets of the bipartite graph, the weight of each potential edge between a helper-helpee pair is determined by Equation 1a. The objective of Harbor's assignment becomes finding the maximum score from all possible matching. Different from the well-known weighted bipartite matching (`WBM`) problem, where the solution yields a one-to-one matching, the `WBbM` model allows a node to match with multiple nodes. This fits with our system design, where one helper vehicle can help multiple helpees. For `WBbM`, each node has a capacity C, which determines the maximum number of nodes it can match to. In Harbor's context, the helping capacity $N(r_j)$ of helpers is determined by their V2I bandwidth, as mentioned before. For solving the `WBbM` problem, it can be reduced to a `WBM` problem by creating replications of nodes. We adopt this method in Harbor's algorithm optimization. Specifically, for each helper with $N(r_j) > 1$, we create $N(r_j)$ copies of it and connect it to all possible neighbor e_i of the original node r_j with the corresponding edge weight. We then solve the newly formed graph using the existing Hungarian algorithm for `WBM` problems. Since for any possible assignment, each r_j can help at most $N(r_j)$ helpees, we make sure that the V2I bandwidth constraint is satisfied as each node after replication will end with $\leq N(r_j)$ matchings. We summarize the proposed matching algorithm in pseudo-code as Algorithm 1. Given that the complexity of solving `WBM` is $O(|R|^2|E|)$, it is straightforward to prove that the complexity of the algorithm becomes $O(|R|^2|E|^2)$ compared to the $O(|R|^{|E|})$ exhaustive search. We evaluate the effectiveness of the `WBbM` optimization in §5.5.

3.4 Timely Delivery of Detection Results

As vehicles need to digest the point cloud data in real-time, it is vital to deliver the remote detection results back to each vehicle in time. Outdated detection results may not be useful because the driving environment can change drastically within seconds. To achieve timely delivery of the detection results, Harbor harnesses both application-level deadline awareness and MAC-layer prioritization.

3.4.1 ***Deadline Awareness.*** To cope with the fast-changing network conditions, vehicles in Harbor perform point cloud encoding adaptation. Harbor uses an existing bandwidth estimation technique [88] to perform upload bandwidth estimation, based on which the encoding level is adjusted accordingly.

Algorithm 1: Harbor's Helper Assignment Algorithm.

Input: helpee set $\mathbf{E} = \{e_1, ..., e_i\}$, helper set $\mathbf{R} = \{r_1, ..., r_j\}$.
Output: A – selected assignment to pair helpees with the helpers.
/* Define V2I bandwidth, helping capacity, score map, and helper set after replication */
$BW, N, S, \mathbf{R}' \leftarrow$ HashMap, HashMap, HashMap, HashSet;
for $r_j \in \mathbf{R}$ **do**
  $BW_j \leftarrow$ getV2IBW(r_j);
  $N_j \leftarrow$ getHelpingCap(BW_j);
  for $k \in \{1, 2, \ldots, N_j\}$ **do**
    /* Replicate a virtual node for compute matching */
    $\mathbf{R}'$.Insert(r_j^k);
for $e_i \in \mathbf{E}$ **do**
  for $r_j \in \mathbf{R}'$ **do**
    /* Compute score for each helper-helpee pair */
    $S_i^j \leftarrow$ ComputeDistScore(e_i, r_j) + ComputeIntfScore(e_i, r_j);
/* Calculate the maximized weighted b-matching results. */
$A =$ Hungarian_WBM(E, R, S)

However, only applying encoding adaptation is inadequate. If the server waits for frames from all vehicles (generated in the same round) to arrive before performing computation, the E2E latency would be inflated by stragglers whose frame uploading is too slow. Based on our measurement, frame upload time can differ by over 0.75s across 6 vehicles due to the above reason. Harbor addresses this latency heterogeneity by incorporating a deadline by which frame merging and detection must start, even if not all frames from the current round have been received.

Specifically, Harbor's server determines this deadline in the following way. Each vehicle has a fixed E2E latency requirement T_{e2e} (§4) for each point cloud frame since the frame is captured from the sensor. Recall from §3.1 that in the design of Harbor, vehicles still run local detection in parallel and use the local detection if remote results are not received after T_{e2e}. Following the timelines in Figure 6, the server can calculate the vehicle-side deadline $t_5 = t_0 + T_{e2e}$. All the vehicle-side timestamps are embedded into control messages sent to the server (§3.1). The server then estimates the downlink one-way delay T_{owd} to get $t_4 = t_5 - T_{owd}$. Finally, based on its detection time T_{remote}, the server computes the deadline timestamp $t_3 = t_4 - T_{comp}$ at which it must start merging all the received sensor data and running detection. Since there are multiple vehicles and their time parameters can differ, the server uses the earliest t_3 as the estimated deadline. In Harbor, vehicles use NTP to synchronize time within ten milliseconds [36], which is sufficient for our application.

Local Merging. When a detection result arrives at a vehicle, the vehicle will first check if the result arrives too late to be used – specifically, whether the result is delivered beyond T_{e2e} since the upload of the sensor data. Any late-arrival result will be discarded. The vehicle maintains a local deadline (different from the edge-side deadline); when the local deadline is missed, the vehicle will fall back to using its local detection result. Next, the vehicle will check if the edge-returned result uses the view uploaded by the vehicle itself. The result is accompanied by a small bitmap to indicate which vehicles' views are processed by the edge and henceforth used in the result. If a vehicle's corresponding bit is missing (due to missing the edge-side merging deadline), the vehicle will perform *local merging* by merging its local detection with the edge-returned result. The merging is local in that the merged view is only used locally, and the vehicle (as a helper) will not further distribute it downstream; instead, a helper always passes the original result from (and signed by) the edge. This helps reduce the attack surface and prevent the erroneous situation where a single vehicle's detection failure pollutes the edge's merged view. If the edge's view passes both checks, it will be used by the vehicle.

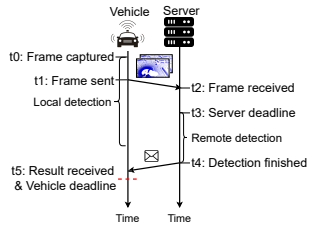

Figure 6: Timeline of Harbor's vehicle-side and server-side data processing.

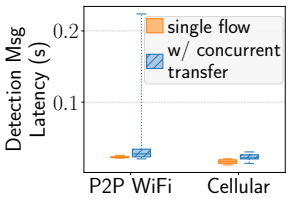

Figure 7: Impact of network types on the detection result delivery latency.

3.4.2 ***MAC-layer Prioritization.*** Different from previous work [88] where only a cellular V2I downlink is used to transmit result messages, in a hybrid CVP involving both V2V and V2I links, sensor data (uplink) and detection result (downlink) compete for transmission in opposite directions over the same V2V wireless medium. Also, their bandwidth consumption is highly imbalanced, with the uplink data dominating the bidirectional traffic. We find such imbalance can significantly inflate the end-to-end detection: when bidirectional traffic is present, overwhelmed by the uplink traffic, the downlink traffic is allocated with little network resources (Figure 7). This thus requires prioritizing the downlink traffic that contains the detection results. We find that performing traffic prioritization at the *application* layer on a helper vehicle does not work. This is because when the application layer of a helper (as a relay) perceives the uplink traffic from the helpee (the origin), the traffic has already saturated the wireless medium and thus starved the downlink traffic.

To address the above challenge, we propose a MAC-layer message prioritization design that prioritizes different message types using priority queues. For queues with higher priorities, we adjust the MAC-layer parameters (§4) to enable a higher probability of accessing the medium. This design allows the lightweight but latency-sensitive detection result messages to be prioritized over data-intensive sensor data transmission, achieving lower E2E latency while maintaining the network throughput for sensor data transmission. Note that the MAC-layer prioritization module reduces the downlink delay for delivering the detection results and is orthogonal to the helper assignment algorithm which optimizes the collaborative data uploading. We validate the effectiveness of MAC-layer message prioritization in §5.5.

4 System Implementation

Server. The server performs helper assignment, sensor data merging, and downstream detection. Harbor's drivable space detection uses the RANSAC [38] algorithm to extract the road plane and mark the points on the road plane as "drivable" (other points are marked as objects). Then it converts the road plane into an occupancy grid of $1m \times 1m$ and labels each grid as either "drivable", "occupied" or "unknown". Harbor's object detection adopts PointPillars [50] as the

backbone object detection neural network. We use the pre-trained models provided by OpenCOOD [65] for detecting objects.

Vehicle. V2V control message exchanges are implemented over UDP due to the broadcast requirement. V2V sensor data exchanges and V2I communications are implemented over TCP. We use the OLSR routing Protocol [34] in V2V network and Draco [12] to encode/decode point cloud data. We apply distance-based cropping: each vehicle crops the point cloud data within a fixed threshold of 50 m. Applying more sophisticated data partition algorithms [88] is easy. We implement the MAC-layer prioritization in Linux kernel [14]. We create a higher priority queue for delivering detection results in addition to the regular MAC-layer frame processing queue. We adjust the contention window ($CW_{min} = 2$) and arbitrated interframe spacing ($AIFS = 1$) to increase the probability of successfully transmitting the detection result packets. We derive a 500 ms latency threshold T_{e2e} by considering several safety-critical scenarios from previous work [63, 88]. This latency threshold is on par with that defined in a previous V2I-based CVP (EMP [88]) plus the edge inference and result delivery time not considered by EMP.

5 Evaluation

We evaluate Harbor and compare it with several state-of-the-art solutions under realistic traffic scenarios and network conditions. We quantify improvements in end-to-end latency and accuracy using trace-driven emulations (§5.2). We show that CVP empowered by Harbor can improve perception accuracy and driving safety in real-world driving scenarios (§5.3). We demonstrate the system scalability and robustness under large number of vehicles and fast-changing topologies in §5.4. Lastly, we present a system overhead study in §5.5.

5.1 Experimental Setup and Methodology

We consider the following evaluation metrics.

• **Object detection accuracy**: It quantifies the difference between predicted object bounding boxes and the ground truth. We use Average Precision (AP) [2] as the metric to evaluate object detection. It is calculated with a threshold of Intersection over Union (IoU) [5] to evaluate the accuracy of object detection. The AP metric counts prediction bounding boxes with IoU higher than the threshold as true positives. We set the IoU threshold as 0.5, a value commonly used for evaluating point cloud object detection [50, 68, 82].

• **Drivable space detection accuracy**: It quantifies the difference between the actual detection result and the ground truth. We compare the predicted grid labels with ground-truth labels. Detection accuracy is defined as $Accuracy = \frac{TP+TN}{\text{Total number of grids}}$. TP (True Positives) is the number of grids whose prediction and ground-truth are both drivable, while TN (True Negatives) is the number of grids whose prediction and ground truth are both occupied or unknown.

• **Detection latency**: It measures the time elapsed from when a point cloud is captured on a vehicle to when the perception result (*e.g.,* drivable space detection and object detection) is ready to use by downstream applications.

We compare Harbor with the following state-of-the-art solutions. **EMP** [88]: a V2I-based cooperative perception system where each vehicle uses V2I to upload LiDAR data and vehicles without V2I connection cannot join the collaboration. **AVR** [63]: a system that uses V2V to send 3D point clouds (same data as Harbor) from other vehicles to a randomly chosen leader vehicle for sharing. **CarSpeak** [49]: a V2V system where each vehicle broadcasts sensor data to all other vehicles for perception. We also implemented its loss-resilient compression technique. For all schemes, the vehicles use detection results described in §3.4.1.

Table 2: Summary of experiment settings.

Experiments	# of CAVs	Speed (km/h)	V2I BW (Mbps)	Traffic Scenes	Point cloud data type
Emulation	2 - 100	0 - 70.0	5.4 - 55.9	roundabout, road segments, intersections, entrance ramp	CARLA
Testbed	3	0 - 32.3	3.3 - 35.7	crossroads, T-intersections, road segments	Real-world

Trace-driven emulation. We conduct large-scale trace-driven emulation using traffic and network traces. Our traffic traces consist of vehicle trajectories and LiDAR point clouds collected from the CARLA simulator [11] in various driving scenarios and traffic topologies (Table 2). We create nodes in Mininet-WiFi [15] and replay the trajectories to emulate the V2V network. Our V2I network traces are LTE and 5G uplink throughput traces collected from real-world driving [58, 88]. The V2I traces are assigned to each vehicle based on a uniform distribution. We use a Linux machine with a 16-core CPU and 32GB memory to run Mininet-WiFi, which creates various numbers of wireless nodes and a server node. Each wireless node runs a Harbor vehicle instance and has two network interfaces: an 802.11g WLAN for communication between vehicles as it is widely used by previous work [66, 75], and an Ethernet interface for connection with the server. We use Linux *tc* to replay our network traces over the Ethernet interface to emulate the V2I network. The avg (stddev) bandwidth of V2V links is measured to be 12.35 (5.68 Mbps) under the emulated 802.11g setup. The point cloud capture rate is set to 10 fps in emulations and real-world experiments.

Real-world CAV testing. We also deploy 3 Lincoln MKZ vehicles as CAVs on four realistic driving scenarios at an autonomous driving testbed, *Mcity* [25]. Our testbed is a real-world mock city for testing CAV applications, which consists of road segments, traffic lights/signs, and testing vehicles. Each vehicle is equipped with OxTS RT3000v3 GPS [9], Velodyne VLP-32C LiDAR [20], and Cohda MK6C OBU [24] for V2V+V2I communication. The same machine in emulation is used as the edge server. We also deploy several other non-connected vehicles as targets for detection. We run experiments on 4 different driving scenarios and run our evaluation on a total of 1602 LiDAR frames.

Real-world vehicle driving test. To examine Harbor under realistic V2V and V2I network conditions on a larger scale, we run real-world field tests by driving 6 cars on the road. Each vehicle is equipped with a laptop that runs a Harbor vehicle instance to replay sensor data from CARLA. In this case, although vehicles still use synthetic sensor data, they experience realistic V2V and V2I network conditions as real-world driving. We use the same server machine in emulation experiments to run the Harbor server instance. For each laptop, we tether a smartphone to it. The smartphones have GPS and each runs an app [17] to fetch locations. The cellular data plans used include AT&T [10], Verizon [21] and

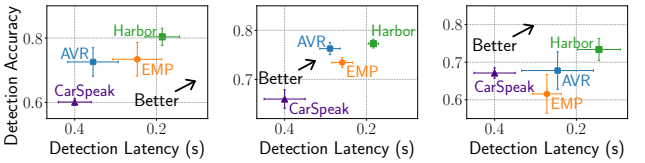

Figure 8: Emulation experiment results for E2E drivable space detection performance of Harbor and baselines.

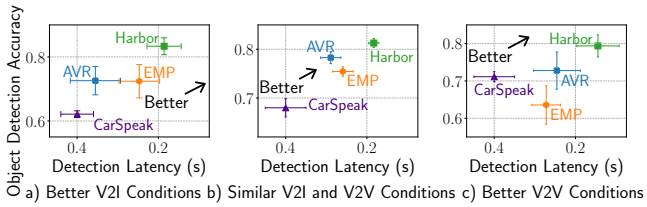

Figure 9: Emulation experiment results for E2E object detection performance of Harbor and baselines.

T-Mobile [18]. The laptops form an ad-hoc network using their WiFi interfaces for V2V communications. While technologies like DSRC [47] are designed for V2V communication, they often require specialized hardware and programming interfaces. Therefore, we choose to use ad hoc WiFi for V2V because of its compatibility with existing Linux OS as previous work [49]. In real-world CAV tests and vehicle driving tests, the bandwidth statistics (average/standard deviation) of the V2I (V2V) networks are 19.98 ± 9.08 (10.41 ± 1.48) Mbps based on our measurements.

5.2 Trace-driven Emulation Results

We first show the latency and accuracy performance of Harbor by running emulation experiments on the CARLA dataset with 2 - 20 CAVs under various traffic densities and network conditions (Table 2). We measure both the detection accuracy and latency overhead for each scheme.

We vary the number of vehicles, mobility patterns, and V2I network conditions, which create **100** different settings in total. We further categorize these settings into 3 classes, based on average V2I bandwidth, V2V distances and the duration of helpee vehicles: (1) better V2I conditions, (2) similar V2I and V2V conditions, and (3) better V2V conditions.

Figure 8 and Figure 9 show the End-to-end (E2E) performance of Harbor and baseline schemes using drivable space detection and object detection as the perception applications. In the aggregated results, we show the mean latency on the x-axis and mean detection accuracy on the y-axis. The error bar shows the standard deviation of each scheme under various settings. As shown, Harbor dramatically improves the detection latency. The mean latency is improved by 39.3% (38.5%) and 36.7% (34.7%) compared to AVR and EMP on drivable space detection (object detection), respectively. The improvements over CarSpeak are even higher: 57.1% (54.2%). Harbor also improves the detection accuracy compared to the baselines. Specifically, on average, it improves the accuracy by 3.5% (7.1%), 7.8% (7.6%), and 12.1% (11.6%), compared to AVR, EMP, and CarSpeak on the two apps. The reasons for the benefits are two-fold. First, Harbor tries to bridge more vehicles together despite their V2I/V2V disconnections by jointly using their V2V and V2I links,

Table 3: Collaborative perception accuracy under different collaboration schemes.

Traffic Scene	Object Detection Acc./Drivable Space Detection Acc.			
	Local-only	EMP	AVR	Harbor
Testbed-Overall	42.67/40.77%	69.84/57.89%	69.90/52.53%	**82.08/70.67%**
- Right turn	28.91/33.28%	59.60/53.84%	42.66/49.59%	**77.93/71.44%**
- Left turn A	34.60/44.57%	81.16/59.79%	79.49/57.25%	**83.55/63.87%**
- Left turn B	35.94/44.48%	79.50/58.12%	61.53/55.86%	**81.30/72.89%**
- Lane merge	71.25/40.78%	72.52/59.81%	80.46/47.43%	**87.57/74.51%**

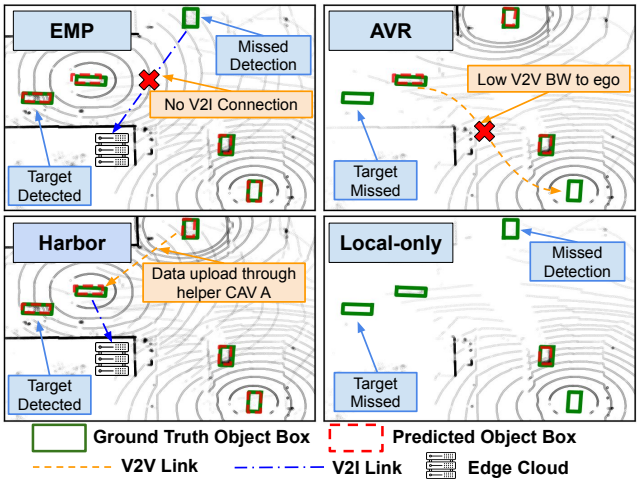

Figure 10: Object detection results of various schemes at *Mcity*. The scenario photo is shown in Figure 11.

leading to a more complete view. Second, Harbor reduces the E2E latency of remote detection results delivery, making the results more likely to be delivered in time and hence utilized. It is expected that the latency improvements of the two apps are similar as the only difference affecting the E2E latency between the two is the computation time difference.

5.3 Performance on Real-world Testbed

In our real-world CAV testbed *Mcity*, we create four traffic scenes (Figure 11) that are challenging based on the NHTSA pre-crash scenario typology [57]: (1) an unprotected right turn in a crossroad; (2) an unprotected left turn in a crossroad (Left turn A); (3) an unprotected left turn in a T-intersection (Left turn B); (4) a lane merging from a parking space.

Under the four real-world driving scenarios, we first evaluate the end-to-end system performance to demonstrate how Harbor benefits CVP tasks after data sharing. Table 3 summarizes the perception accuracy of both object and drivable space detection tasks of different schemes. Overall, Harbor improves the object detection (drivable space detection) accuracy by 2% - 18% (4 - 18%) compared to EMP, 2% - 36% (6 - 27%) compared to AVR, and 16% - 49% (19 - 38%) compared to local-only. Specifically, Figure 10 depicts a specific scene to show how Harbor outperforms the baselines. In this scenario, the target vehicle and other vehicles in the scene are consistently detected by Harbor, while AVR does not detect the target because limited V2V bandwidth of one CAV to the ego vehicle. While EMP detects the target vehicle, it misses another vehicle in the opposite lane because another CAV does not upload its data on time to the edge server due to limited V2I bandwidth. The benefits of Harbor in perception accuracy come from the fact that Harbor

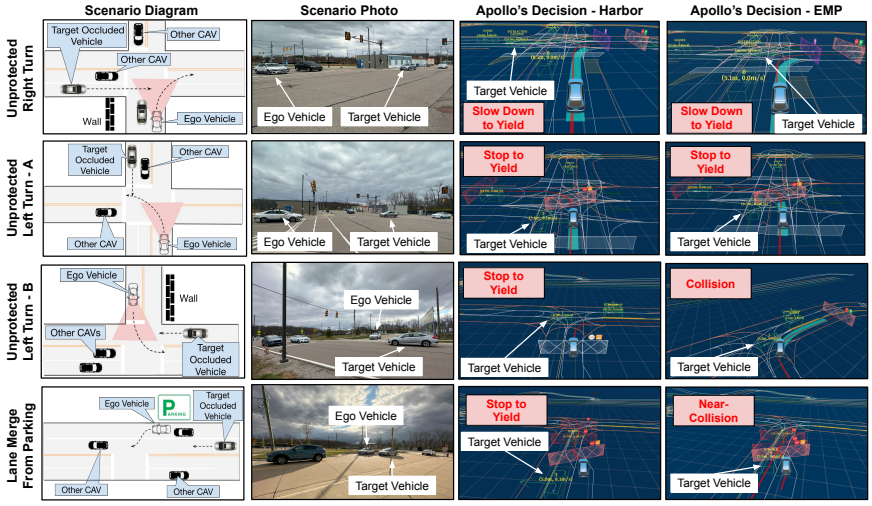

Figure 11: End-to-end driving outcome using Baidu Apollo on real-world scenarios.

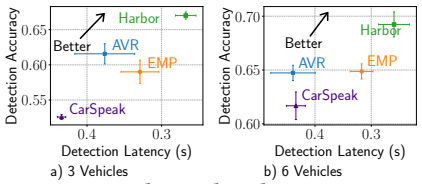

Figure 12: E2E live vehicular experiments.

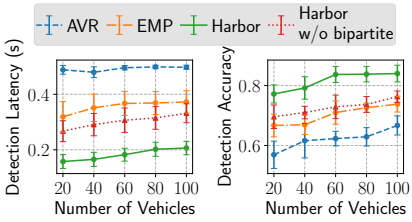

Figure 13: Performance of Harbor w/ large number of CAVs.

Table 4: Additional driving reaction time (compared to Local-only) and outcome of different collaboration schemes.

Traffic Scene	Additional Reaction Time (s)/Driving Outcome		
	EMP	AVR	Harbor
Testbed-Overall	+ 0.76/1 crash	+ 0.58 /1 crash	**+ 1.26/0 crash**
- Right turn	+ 0.69/safe-pass	+ 0.41/crash	**+ 1.60/safe-pass**
- Left turn A	+ 1.05/safe-pass	+ 0.20/safe-pass	**+ 1.23/safe-pass**
- Left turn B	+ 0.73/crash	+ 1.13/safe-pass	**+ 1.42/safe-pass**
- Lane merge	+ 0.57/near-miss	+ 0.57/near-miss	**+ 0.79/safe-pass**

Notes: A near-miss occurs when the ego and target vehicle pass within 3 m of each other.

bridges vehicles with limited V2I performance by judiciously using the spare V2I resources from other nearby CAVs, resulting in a larger coverage of 3D data of the environment.

End-to-end driving outcome. To better understand the driving outcome of each scenario, we deploy the latest Baidu Apollo Driving Software [23] to analyze the driving decision by applying Harbor (Table 4). As depicted in Figure 11, EMP fails in 2 of the 4 challenging cases (1 collision and 1 near-collision). The target vehicle is occluded hence undetected by the ego vehicle. Furthermore, the pure V2I scheme also misses the detection from the server side because other CAVs do not upload their sensor data in time due to poor V2I conditions. In contrast, under all 4 scenarios, Harbor can make prompt decisions, including reducing speed and stopping to yield, to avoid the potential conflict of its planned driving trajectory with the target occluded vehicle thanks to the collaboration to detect the target vehicle earlier in the scene. We also perform analysis on the additional reaction time (*i.e.,* how much earlier the ego vehicle detects the target) for the ego AV to make driving decisions compared to local-only perception (Table 4). While existing collaboration methods also increase the reaction time for the ego by detecting the target earlier, Harbor results in an average of 0.5 - 0.7 seconds (65%) more reaction time compared to the baselines.

Field driving test with more vehicles. To showcase the real-world impact of Harbor for more vehicles, Figure 12 plots the field test results: Harbor outperforms EMP, AVR, and Carspeak by reducing 18.6%, 29.9% and 37.7% of detection latency and improving 8.0%, 5.78% and 11.0% on detection accuracy. As the number of vehicles increased from 3 to 6, Harbor's performance remains consistently the best, showing better scalability. In real-world driving tests, we have observed that the RTT between commercial cellular network to a server is 97.4 ± 1.49ms, much longer than a real vehicle-to-edge communication latency (about 20 ms [89]). Compared to emulation experiments, Harbor's improvements in latency are slightly lower due to higher base V2I latency.

5.4 Scaling to Large-scale Deployment

We evaluate how Harbor scales to environments with a larger number of vehicles. Following the vehicle density and speed information from 2 public vehicle trajectory datasets [16, 91], we place up to 100 vehicles in the CARLA simulator and use trace-driven emulation to replay their trajectories and data for evaluation. Note that as the total number of vehicles increases, both the number of helpers and helpees increase.

Figure 13 shows how the performance of Harbor changes with an increasing number of vehicles. Harbor outperforms EMP (AVR) by 37.1% (51.3%) in average detection latency, and 11.3% (19.4%) in detection accuracy, respectively. We observe that applying a V2I-based scheme is indeed more scalable than a pure V2V scheme, as shown before [88]. Harbor achieves better scalability than V2I by using the V2V medium for helpees and applying bipartite optimization to adapt to network dynamics. Note that Harbor not only improves the detection latency but also accuracy compared to Harbor without bipartite optimization. This is because the bipartite optimization enables faster computation of assignments and henceforth reacts faster to network changes, resulting in more data being delivered to the edge server and used for perception.

Robustness of Harbor **under fast changing topologies.** Next, we analyze the robustness of Harbor under variable traffic topologies. To quantify the degree of topology change, we use a metric calculated from vehicle trajectories: the average relative distance variation. A higher variation of the relative distance indicates that the relative position of ego-CAV varies more compared to others, making the topology more fast-changing. The average relative distance variation is derived as follows. (1) For each CAV, we calculate the standard deviation of its relative distance to all the other CAVs over the entire trajectory every t seconds (we set $t = 0.1$). (2) We then average the standard deviations from all CAVs by the number

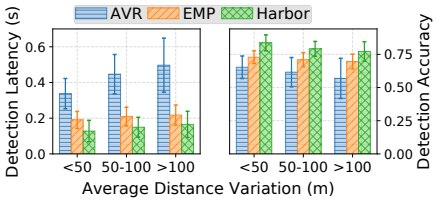

Figure 14: Performance of Harbor under various levels of topology changes.

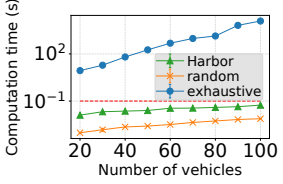

Figure 15: Harbor's helper assignment overhead analysis.

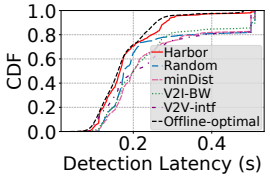

Figure 16: Benefits of Harbor's assignment strategy.

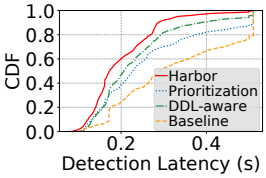

Figure 17: Benefits of detection result delivery strategy.

of vehicles. We conduct this experiment using 50 emulated vehicles. Figure 14 shows the performance of Harbor under various levels of topology changes. Under slower changing traffic topologies (average relative distance variation <50m), Harbor outperforms AVR (EMP) by 32.6% (61.2%) in latency and 10.7% (18.3%) in accuracy. Even under the fastest-changing traffic conditions (average relative distance variation >100m), Harbor's improvements over AVR and EMP only reduce slightly, despite the excessive mobility's impact on V2V network performance, thanks to Harbor's robust helper assignment algorithm.

Table 5: Control-plane and data-plane message overhead for each point cloud frame per vehicle(mean±std deviation).

Metrics	AVR	EMP	Harbor (Helper/Helpee)
Control data (Bytes)	24	24	67±5.5 / 67±5.5
Transmission Latency (ms)	28.1±24.6	27.6±18.6	27.9±17.8 / 32.4±22.0
Bandwidth Req. (Mbps)	<0.1	<0.1	<0.1 / <0.1
Sensor data (KB)	42.9±5.07	46.2±3.5	45.9±3.2 / 40.6±7.1
Transmission Latency (ms)	158.3±31.2	91.5±19.9	36.3±19.1 / 72.1±22.2
Bandwidth Req. (Mbps)	3.4±0.5	3.7±0.3	3.6±0.3 / 3.2±0.6

Notes: *Bandwidth Req.* - Bandwidth Requirement at 10 FPS.

5.5 System Overhead & Microbenchmarks

We study the overhead and benefits of design decisions made by Harbor from three perspectives: (1) the control-plane and data plane message overhead, which measures control/data message size, its transmission latency and the corresponding bandwidth requirements; (2) the runtime performance of Harbor's helper assignment algorithm and its benefits to the E2E detection latency; (3) the latency improvements made by timely delivery of detection results strategy. For ease of presentation, we use drivable space detection as the example application for the remaining part of the evaluation.

5.5.1 System Overhead Analysis. **Control-plane and data-plane message overhead.** Table 5 compares the control plane and data plane message overhead between Harbor and other existing V2I/V2V-only CVP solutions. Apart from the vehicle location data also used in EMP and AVR, Harbor's control message further includes V2I bandwidth measurements and V2V routing tables. The additional data size in the control plane is negligible (fewer than 50 bytes per point cloud frame). Therefore, Harbor's control-plane message overhead is similar to EMP and AVR, as mentioned in §3.1. Also, as the last row in Table 5 suggests, helpee vehicles in Harbor have slightly higher transmission latency for control messages. This is attributed to its hybrid (V2I+V2V) network topology. For data messages, all three schemes upload similar size of point cloud data (after encoding), while Harbor' transmission latency is significantly lower than EMP and AVR because Harbor judiciously allows helpee vehicles to use the spare V2I bandwidth from helpers. Vehicles with poor V2I conditions perform poorly on EMP as they use V2I-only communication. AVR suffers from higher data-plane latency than EMP due to less scalable V2V-only communication, as shown in [88]. For bandwidth requirements, Harbor requires the same level of uplink bandwidth as V2I-based systems, while achieving better latency and scalability.

Overhead of Harbor's assignment algorithm. We evaluate the runtime performance of Harbor's helper-helpee assignment algorithm by comparing it with two baselines: (1) **Random**: randomly assign helpers to helpees. (2) **Exhaustive**: perform brute-force search to find the optimal assignment based on Harbor's formulation. We consider two metrics: the assignment execution latency (Figure 15) and the end-to-end detection latency (Figure 16). As shown in Figure 15, the scheduling time of the exhaustive search grows quickly as the number of vehicles increases – taking more than 10s to compute the assignment for 20 vehicles. On the other hand, as shown in Figure 16, while the random scheme provides decisions within 100ms (the LiDAR generation cycle), its end-to-end detection latency is high, due to its selected suboptimal paths. In contrast, Harbor with its built-in bipartite optimization can efficiently perform helper assignment within 100ms (Figure 15). This, together with the resulting judiciously determined paths, leads to an average end-to-end detection latency of 175ms even for 100 vehicles, outperforming all the other baseline schemes (Figure 16).

5.5.2 Microbenchmarks. **Strategic Helper Assignment.** We examine the benefits of one of Harbor's strategic helper assignment through trace-driven emulation. To make a fair comparison, we modify only the assignment scheduling logic in Harbor to have five different baseline assignment schemes: (1) **Random:** randomly assign helpers to helpees; (2) **Min-distance:** assign helpers by minimizing the total distance between each helper-helpee pair; (3) **V2I-BW:** assign helpers by selecting the helpers with higher V2I bandwidths; (4) **V2V-intf:** assign helpers by maximizing the interference score on the V2V network; (5) **Offline-optimal:** an offline-generated solution by examining all possible assignments at each scheduling interval with the best performance.

Figure 16 shows the benefits of Harbor's strategic helper assignment leveraging different information sources. The 90^{th} and 95^{th} percentile latency have improved by 43.8% and 38.8% compared to the baselines. As the baseline assignment algorithms often optimize a single factor, they fail to perform well in all settings. By considering all different factors that affect V2I and V2V data transmission, Harbor becomes more robust under various driving and network conditions, thus greatly improving the tail latency. Furthermore, Harbor achieves 90^{th} and 95^{th} percentile tail latency that is 91.8% to 92.8% close to the offline-optimal scheme.

Timely Detection Result Delivery. We compare Harbor's fast detection result strategy with three baseline strategies: (1) **Baseline**: Harbor without the entire fast detection result delivery design (§3.4); (2) **Prioritization**, which adds the MAC layer prioritization on top of the baseline; (3) Deadline-aware (**DDL-aware**), which incorporates server-side deadline-awareness in frame merging to the baseline. Figure 17 shows the effectiveness of Harbor's decisions in fast detection result delivery. Both MAC-layer result message prioritization and server-side deadline awareness reduce the average detection latency by over 19.0%. Harbor achieves a higher latency improvement of 38.3% by leveraging both techniques. Harbor also improves 90^{th} percentile latency by 25.6% (42.6%), compared to Prioritization (DDL-aware). While the MAC-layer result message prioritization module necessitates modifications to the lower layers of the operating system's network stack, its demonstrated effectiveness suggests that future integration with existing CAV hardware platforms (e.g., NVIDIA DRIVE AGX [26]) could enable its practical deployment in current autonomous vehicle systems. Such integration would allow this optimization to enhance real-world CVP solutions by improving data transmission efficiency and reducing latency in various situations.

6 Discussion and Related Work

Security and Privacy of Collaborative Perception Systems. While beyond the scope of this work, security problems might arise in collaborative vehicular perception systems. Malicious vehicles can send fake/modified data to affect the collaborative detection results [72]. Under the context of Harbor, both infrastructure and vehicles can take actions to improve system security: 1) Edge server can leverage public key infrastructures (PKI) [73] to issue certificates to authorized vehicles and revoke access whenever anomaly behavior is detected. 2) On vehicles, Trusted Execution Environment (TEE) [43] can be deployed to help prevent data spoofing and modification from attackers. At the application level, Harbor can leverage existing multi-vehicle collaboration defense method [86] to check sensor data consistency.

Hybrid V2V+V2I Vehicular Communications. [59] leverages V2V network to speed up V2I file uploading to the roadside unit for a single source vehicle. [74] proposes the V2X protocol for optimizing network message propagation by allowing protocol switching between V2V and V2I. Different from previous work in this space, CVP imposes new challenges in that data uploading from different vehicles needs to be jointly optimized, *i.e.,* collaboration of multiple data traffic, to reduce perception latency. Our system Harbor focuses on assisting 3D CVP tasks by strategically utilizing V2V and V2I network resources. The design of Harbor optimizes the overall delay for all nodes participating in the collaboration and jointly handles bandwidth-intensive uplink and latency-sensitive downlink transmissions.

Cooperative Vehicular Perception. Various efforts have been made on cooperative vehicular perception. Existing data sharing systems either transfer sensor data or processed features between vehicles [32, 49, 63, 64, 78] or directly send each vehicle's data to a server [48, 55, 88]. Our work advocates using both V2V and V2I to better adapt to different network connectivity and bandwidth conditions. We develop solutions for the dynamic establishment of V2V and V2I channels, and algorithms that efficiently assign relay vehicles considering the wireless network and physical properties. As mentioned in §2, while we focus on CVP data-sharing at the raw-data level, the underlying principle of Harbor can be applied to collaborative perception designs that share intermediate features. We left the integration of intermediate feature-based sharing with Harbor as future work.

Collaborative Mobile Systems. Harbor is partially inspired by mobile systems involving network-interface-level or device-level collaborations [29, 44, 46, 60, 90]. They focus on various applications such as smartphone file download and video streaming. In contrast, our work targets the CVP domain, and thus faces unique challenges as described earlier (§1).

Peer-to-peer Wireless Multicast Systems. Collaborative vehicular perception tasks involve sharing data among multiple vehicles or agents in a dynamic wireless network. While Harbor focuses on combining V2V and V2I for better CVP performance, peer-to-peer (P2P) multicast technologies can potentially further enhance the efficiency of such data exchanges. Innovations in wireless multicast protocols [37, 51, 80] demonstrate that multicast can improve bandwidth utilization and reduce communication overhead in scenarios with multiple participants. These technologies are particularly relevant for CVP systems like Harbor, where multiple vehicles need to access the same control data and perception results. Incorporating peer-to-peer multicast strategies could further reduce the latency of data dissemination, especially in environments with dense vehicular traffic. Future extensions of Harbor could leverage these technologies to optimize data sharing between helper and helpee vehicles, further enhancing the system's scalability and efficiency.

7 Limitations and Conclusion

Harbor bears several limitations. First, due to its opportunistic nature, Harbor is best-effort, without any service quality guarantee or bound. Second, our design and evaluation focus on a single edge node's service area (with multiple subareas supported though, see §3.1). How to scale up Harbor to the city level requires further research. Third, our current implementation only incorporates basic security measures such as end-to-end encryption for point cloud upload's confidentiality and edge's signature for edge-returned results' integrity. More efforts (*e.g.,* multi-vehicle collaboration defense [86]) are needed to defend various attacks in hybrid CVP.

Despite the above limitations, as a first-of-its-kind hybrid CVP system, Harbor considerably outperforms existing V2V- and V2I-only schemes, as backed up by extensive field tests and large-scale emulation. We envision deep collaborations among CAVs, edge/cloud, smart road infrastructure, and even unmanned aerial vehicles (UAVs) to be a key enabler of next-generation transportation systems. Our work makes an important step towards this ambitious goal.

Acknowledgments

We would like to thank our anonymous shepherd and reviewers for their valuable comments and feedback. This work was supported by NSF under CMMI-2038215, CNS-2323174, and the National AI Institute for Edge Computing Leveraging Next Generation Wireless Networks, Grant # 2112562.

References

[1] 2013. Tesla Model S software release notes v5.8. https://www.tesla.com/sites/default/files/blog_attachments/software_update_5.8.pdf.

[2] 2013. The KITTI benchmark suite. https://www.cvlibs.net/datasets/kitti/eval_object.php?obj_benchmark=3d.

[3] 2015. AT&T and Audi to Wirelessly Connect all 2016 Model Year Vehicles. https://about.att.com/story/att_and_audi_to_wirelessly_connect_all_2016_model_year_vehicles.html.

[4] 2016. Mercedes-Benz mbrace. https://www.mbusa.com/vcm/MB/DigitalAssets/pdfmb/mbraceservicebrochures/1527_MBfactsheet_0814_KH_v2.pdf.

[5] 2019. Intersection over Union (IoU). https://cocodataset.org/#detection-eval.

[6] 2019. Study shows autonomous vehicles can help improve traffic flow. https://phys.org/news/2018-02-autonomous-vehicles-traffic.html.

[7] 2020. Autonomous vehicles for safety. https://www.nhtsa.gov/technology-innovation/automated-vehicles-safety.

[8] 2020. Fatality Analysis Reporting System. https://www.nhtsa.gov/crash-data-systems/fatality-analysis-reporting-system.

[9] 2020. OxTS RT3000 v3 GPS+IMU system. https://www.oxts.com/products/rt3000-v3/.

[10] 2021. AT&T Maps - Wireless Coverage. https://www.att.com/maps/wireless-coverage.html.

[11] 2021. CARLA: Open-source simulator for autonomous driving research. https://carla.org/.

[12] 2021. Draco 3D Graphics Compression. https://google.github.io/draco/.

[13] 2021. FCC Mobile Broadband Maps. https://www.fcc.gov/BroadbandData/MobileMaps.

[14] 2021. Linux Kernel v5.8. https://github.com/torvalds/linux/tree/v5.8.

[15] 2021. Mininet-WiFi: Emulator for Software-Defined Wireless Networks. https://github.com/intrig-unicamp/mininet-wifi.

[16] 2021. Next Generation Simulation (NGSIM) Dataset. https://ops.fhwa.dot.gov/trafficanalysistools/ngsim.html.

[17] 2021. Share GPS. http://jillybunch.com/sharegps/.

[18] 2021. T-Mobile Coverage Map. https://www.t-mobile.com/coverage/coverage-map/.

[19] 2021. Velodyne LiDAR HDL-32E sensor. https://velodynelidar.com/products/hdl-32e/.

[20] 2021. Velodyne LiDAR HDL-64E. https://www.velodynelidar.com/hdl-64e.html.

[21] 2021. Verizon Coverage Map. https://www.verizon.com/coverage-map/.

[22] 2021. ZED Stereo Camera. https://www.stereolabs.com/zed/.

[23] 2022. Baidu Apollo. https://www.apollo.auto/.

[24] 2022. Cohda Wireless OBU Solution. https://www.cohdawireless.com/solutions/mk6/.

[25] 2023. Mcity - University of Michigan. https://mcity.umich.edu/.

[26] 2024. NVIDIA AGX Systems. https://www.nvidia.com/en-us/deep-learning-ai/products/agx-systems/.

[27] Daniel Aguayo, John Bicket, Sanjit Biswas, Glenn Judd, and Robert Morris. 2004. Link-level measurements from an 802.11 b mesh network. In *Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*. 121–132.

[28] Faez Ahmed, John P Dickerson, and Mark Fuge. 2017. Diverse weighted bipartite b-matching. *arXiv preprint arXiv:1702.07134* (2017).

[29] Ganesh Ananthanarayanan, Venkata N Padmanabhan, Lenin Ravindranath, and Chandramohan A Thekkath. 2007. Combine: leveraging the power of wireless peers through collaborative downloading. In *MobiSys*. ACM.

[30] Ian D Chakeres and Elizabeth M Belding-Royer. 2004. AODV routing protocol implementation design. In *24th International Conference on Distributed Computing Systems Workshops, 2004. Proceedings*. IEEE, 698–703.

[31] Qi Chen, Xu Ma, Sihai Tang, Jingda Guo, Qing Yang, and Song Fu. 2019. F-cooper: Feature based cooperative perception for autonomous vehicle edge computing system using 3D point clouds. In *Proceedings of the 4th ACM/IEEE Symposium on Edge Computing*. 88–100.

[32] Qi Chen, Sihai Tang, Qing Yang, and Song Fu. 2019. Cooper: Cooperative perception for connected autonomous vehicles based on 3d point clouds. In *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 514–524.

[33] Amit Chougule, Vinay Chamola, Aishwarya Sam, Fei Richard Yu, and Biplab Sikdar. 2023. A Comprehensive review on limitations of autonomous driving and its impact on accidents and collisions. *IEEE Open Journal of Vehicular Technology* (2023).

[34] Thomas Clausen, Philippe Jacquet, Cédric Adjih, Anis Laouiti, Pascale Minet, Paul Muhlethaler, Amir Qayyum, and Laurent Viennot. 2003. Optimized link state routing protocol (OLSR). (2003).

[35] Jiaxun Cui, Hang Qiu, Dian Chen, Peter Stone, and Yuke Zhu. 2022. COOPERNAUT: End-to-End Driving with Cooperative Perception for Networked Vehicles. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 17252–17262.

[36] Sandeep D'souza, Heiko Koehler, Akhilesh Joshi, Satyam Vaghani, and Ragunathan Rajkumar. 2019. Quartz: time-as-a-service for coordination in geo-distributed systems. In *Proceedings of the 4th ACM/IEEE Symposium on Edge Computing*. 264–279.

[37] Mohammed Elbadry, Fan Ye, and Peter Milder. 2024. Wireless Multicast Rate Control Adaptive to Application Goodput and Loss Requirements. In *2024 IEEE/ACM Ninth International Conference on Internet-of-Things Design and Implementation (IoTDI)*. IEEE, 25–36.

[38] Martin A Fischler and Robert C Bolles. 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM* 24, 6 (1981), 381–395.

[39] Alexander Frickenstein, Manoj-Rohit Vemparala, Jakob Mayr, Naveen-Shankar Nagaraja, Christian Unger, Federico Tombari, and Walter Stechele. 2020. Binary DAD-Net: Binarized driveable area detection network for autonomous driving. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2295–2301.

[40] Offer Grembek, Alex Kurzhanskiy, Aditya Medury, Pravin Varaiya, and Mengqiao Yu. 2019. Making intersections safer with I2V communication. *Transportation Research Part C: Emerging Technologies* 102 (2019), 396–410.

[41] Ahmad Hassan, Arvind Narayanan, Anlan Zhang, Wei Ye, Ruiyang Zhu, Shuowei Jin, Jason Carpenter, Z. Morley Mao, Feng Qian, and Zhi-Li Zhang. 2022. Vivisecting Mobility Management in 5G Cellular Networks. In *Proceedings of the ACM SIGCOMM 2022 Conference* (Amsterdam, Netherlands) *(SIGCOMM '22)*. Association for Computing Machinery, New York, NY, USA, 86–100. https://doi.org/10.1145/3544216.3544217

[42] Yuze He, Li Ma, Zhehao Jiang, Yi Tang, and Guoliang Xing. 2021. VI-eye: Semantic-based 3D point cloud registration for infrastructure-assisted autonomous driving. In *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*. 573–586.

[43] Shengtuo Hu, Qi Alfred Chen, Jiwon Joung, Can Carlak, Yiheng Feng, Z Morley Mao, and Henry X Liu. 2020. Cvshield: Guarding sensor data in connected vehicle with trusted execution environment. In *Proceedings of the Second ACM Workshop on Automotive and Aerial Vehicle Security*. 1–4.

[44] Shuowei Jin, Ruiyang Zhu, Ahmad Hassan, Xiao Zhu, Xumiao Zhang, Z Morley Mao, Feng Qian, and Zhi-Li Zhang. 2024. OASIS: Collaborative Neural-Enhanced Mobile Video Streaming. In *Proceedings of the 15th ACM Multimedia Systems Conference*. 45–55.

[45] Anand Kashyap, Samrat Ganguly, and Samir R Das. 2007. A measurement-based approach to modeling link capacity in 802.11-based wireless networks. In *Proceedings of the 13th annual ACM international conference on Mobile computing and networking*. 242–253.

[46] Lorenzo Keller, Anh Le, Blerim Cici, Hulya Seferoglu, Christina Fragouli, and Athina Markopoulou. 2012. Microcast: Cooperative video streaming on smartphones. In *Proceedings of the 10th international conference on Mobile systems, applications, and services*. 57–70.

[47] John B Kenney. 2011. Dedicated short-range communications (DSRC) standards in the United States. *Proc. IEEE* 99, 7 (2011), 1162–1182.

[48] Swarun Kumar, Shyamnath Gollakota, and Dina Katabi. 2012. A cloud-assisted design for autonomous driving. In *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*. 41–46.

[49] Swarun Kumar, Lixin Shi, Nabeel Ahmed, Stephanie Gil, Dina Katabi, and Daniela Rus. 2012. Carspeak: a content-centric network for autonomous driving. *ACM SIGCOMM Computer Communication Review* 42, 4 (2012), 259–270.

[50] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. 2019. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 12697–12705.

[51] Chenglin Li, Hongkai Xiong, Junni Zou, and Dapeng Oliver Wu. 2017. Joint dynamic rate control and transmission scheduling for scalable video multirate multicast over wireless networks. *IEEE Transactions on Multimedia* 20, 2 (2017), 361–378.

[52] Yang Li, Hao Lin, Zhenhua Li, Yunhao Liu, Feng Qian, Liangyi Gong, Xianlong Xin, and Tianyin Xu. 2021. A nationwide study on cellular reliability: Measurement, analysis, and enhancements. In *Proceedings of the 2021 ACM SIGCOMM 2021 Conference*. 597–609.

[53] Yiming Li, Dekun Ma, Ziyan An, Zixun Wang, Yiqi Zhong, Siheng Chen, and Chen Feng. 2022. V2X-Sim: Multi-agent collaborative perception dataset and benchmark for autonomous driving. *IEEE Robotics and Automation Letters* 7, 4 (2022), 10914–10921.

[54] Hansi Liu, Pengfei Ren, Shubham Jain, Mohannad Murad, Marco Gruteser, and Fan Bai. 2019. FusionEye: Perception Sharing for Connected Vehicles and its Bandwidth-Accuracy Trade-offs. In *2019 16th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. IEEE, 1–9.

[55] Qiang Liu, Tao Han, Jiang Linda Xie, and BaekGyu Kim. 2021. LiveMap: Real-time dynamic map in automotive edge computing. In *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE, 1–10.

[56] Guiyang Luo, Chongzhang Shao, Nan Cheng, Haibo Zhou, Hui Zhang, Quan Yuan, and Jinglin Li. 2023. Edgecooper: Network-aware cooperative lidar perception for enhanced vehicular awareness. *IEEE Journal on Selected Areas in Communications*

(2023).
[57] Wassim G Najm, Raja Ranganathan, Gowrishankar Srinivasan, John D Smith, Samuel Toma, Elizabeth D Swanson, August Burgett, et al. 2013. *Description of light-vehicle pre-crash scenarios for safety applications based on vehicle-to-vehicle communications.* Technical Report. United States. Department of Transportation. National Highway Traffic Safety ....
[58] Arvind Narayanan, Eman Ramadan, Rishabh Mehta, Xinyue Hu, Qingxu Liu, Rostand AK Fezeu, Udhaya Kumar Dayalan, Saurabh Verma, Peiqi Ji, Tao Li, et al. 2020. Lumos5g: Mapping and predicting commercial mmwave 5g throughput. In *Proceedings of the ACM Internet Measurement Conference*. 176–193.
[59] Yuanzhi Ni, Jianping He, Lin Cai, and Yuming Bo. 2018. Data uploading in hybrid V2V/V2I vehicular networks: Modeling and cooperative strategy. *IEEE Transactions on Vehicular Technology* 67, 5 (2018), 4602–4614.
[60] Cătălin Nicutar, Dragoş Niculescu, and Costin Raiciu. 2014. Using cooperation for low power low latency cellular connectivity. In *CoNEXT*. ACM, 337–348.
[61] Jitendra Padhye, Sharad Agarwal, Venkata N Padmanabhan, Lili Qiu, Ananth Rao, and Brian Zill. 2005. Estimation of link interference in static multi-hop wireless networks. In *Proceedings of the 5th ACM SIGCOMM Conference on Internet Measurement*. 28–28.
[62] Anshul Paigwar, Özgür Erkent, David Sierra-Gonzalez, and Christian Laugier. 2020. Gndnet: Fast ground plane estimation and point cloud segmentation for autonomous vehicles. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2150–2156.
[63] Hang Qiu, Fawad Ahmad, Fan Bai, Marco Gruteser, and Ramesh Govindan. 2018. AVR: Augmented vehicular reality. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 81–95.
[64] Hang Qiu, Pohan Huang, Namo Asavisanu, Xiaochen Liu, Konstantinos Psounis, and Ramesh Govindan. 2021. Autocast: Scalable infrastructure-less cooperative perception for distributed collaborative driving. *arXiv preprint arXiv:2112.14947* (2021).
[65] Xin Xia Xu Han Jinlong Li Jiaqi Ma Runsheng Xu, Hao Xiang. 2022. OPV2V: An Open Benchmark Dataset and Fusion Pipeline for Perception with Vehicle-to-Vehicle Communication. In *2022 IEEE International Conference on Robotics and Automation (ICRA)*.
[66] Vishal Sharma, Harsukhpreet Singh, and M Kaur. 2013. Implementation and Analysis of OFDM based IEEE 802.11 g VANET. *International Journal of Computer Networking, Wireless and Mobile Communications (IJCNWMC), ISSN* (2013), 2250–1568.
[67] Shuyao Shi, Jiahe Cui, Zhehao Jiang, Zhenyu Yan, Guoliang Xing, Jianwei Niu, and Zhenchao Ouyang. 2022. VIPS: Real-time perception fusion for infrastructure-assisted autonomous driving. In *Proceedings of the 28th Annual International Conference on Mobile Computing And Networking*. 133–146.
[68] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. 2019. Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 770–779.
[69] Zhiying Song, Fuxi Wen, Hailiang Zhang, and Jun Li. 2022. An Efficient and Robust Object-Level Cooperative Perception Framework for Connected and Automated Driving. *arXiv preprint arXiv:2210.06289* (2022).
[70] Sanjib Sur, Xinyu Zhang, Parmesh Ramanathan, and Ranveer Chandra. 2016. {BeamSpy}: Enabling Robust 60 {GHz} Links Under Blockage. In *13th USENIX symposium on networked systems design and implementation (NSDI 16)*. 193–206.
[71] Muhammad Naeem Tahir, Pekka Leviäkangas, and Marcos Katz. 2022. Connected vehicles: V2V and V2I road weather and traffic communication using cellular technologies. *Sensors* 22, 3 (2022), 1142.
[72] James Tu, Tsunhsuan Wang, Jingkang Wang, Sivabalan Manivasagam, Mengye Ren, and Raquel Urtasun. 2021. Adversarial Attacks On Multi-Agent Communication. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 7768–7777.
[73] Rens Wouter van der Heijden, Stefan Dietzel, Tim Leinmüller, and Frank Kargl. 2018. Survey on misbehavior detection in cooperative intelligent transportation systems. *IEEE Communications Surveys & Tutorials* 21, 1 (2018), 779–811.
[74] Anna Maria Vegni and Thomas DC Little. 2011. Hybrid vehicular communications based on V2V-V2I protocol switching. *International Journal of Vehicle Information and Communication Systems* 2, 3-4 (2011), 213–231.
[75] Dian Vektorendra Wahyuda, Dedy Achmadi, Riri Fitri Sari, et al. 2017. Comparison of different WLAN standard on propagation performance in V2V named data networking. In *2017 IEEE Asia Pacific Conference on Wireless and Mobile (APWiMob)*. IEEE, 128–133.
[76] Qing Wang, Pingyi Fan, and Khaled Ben Letaief. 2011. On the joint V2I and V2V scheduling for cooperative VANETs with network coding. *IEEE Transactions on Vehicular Technology* 61, 1 (2011), 62–73.
[77] Tianhang Wang, Guang Chen, Kai Chen, Zhengfa Liu, Bo Zhang, Alois Knoll, and Changjun Jiang. 2023. UMC: A Unified Bandwidth-efficient and Multi-resolution based Collaborative Perception Framework. *arXiv preprint arXiv:2303.12400* (2023).
[78] Tsun-Hsuan Wang, Sivabalan Manivasagam, Ming Liang, Bin Yang, Wenyuan Zeng, and Raquel Urtasun. 2020. V2vnet: Vehicle-to-vehicle communication for joint perception and prediction. In *European Conference on Computer Vision*. Springer, 605–621.
[79] Dehui Wei, Jiao Zhang, Haozhe Li, Zhichen Xue, Yajie Peng, and Rui Han. 2023. Multipath Smart Preloading Algorithms in Short Video Peer-to-Peer CDN Transmission Architecture. *IEEE Network* (2023).
[80] Fei Wu, Yang Yang, Ouyang Zhang, Kannan Srinivasan, and Ness B Shroff. 2016. Anonymous-query based rate control for wireless multicast: Approaching optimality with constant feedback. In *Proceedings of the 17th ACM International Symposium on Mobile Ad Hoc Networking and Computing*. 191–200.
[81] Runsheng Xu, Hao Xiang, Zhengzhong Tu, Xin Xia, Ming-Hsuan Yang, and Jiaqi Ma. 2022. V2x-vit: Vehicle-to-everything cooperative perception with vision transformer. In *European conference on computer vision*. Springer, 107–124.
[82] Runsheng Xu, Hao Xiang, Xin Xia, Xu Han, Jinlong Liu, and Jiaqi Ma. 2021. OPV2V: An Open Benchmark Dataset and Fusion Pipeline for Perception with Vehicle-to-Vehicle Communication. *arXiv preprint arXiv:2109.07644* (2021).
[83] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. 2021. Center-based 3d object detection and tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 11784–11793.
[84] Yunshuang Yuan, Hao Cheng, and Monika Sester. 2022. Keypoints-Based Deep Feature Fusion for Cooperative Vehicle Detection of Autonomous Driving. *IEEE Robotics and Automation Letters* 7, 2 (2022), 3054–3061.
[85] Hongqiang Zhai and Yuguang Fang. 2006. Physical carrier sensing and spatial reuse in multirate and multihop wireless ad hoc networks. In *Proceedings IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications*. Citeseer, 1–12.
[86] Qingzhao Zhang, Shuowei Jin, Ruiyang Zhu, Jiachen Sun, Xumiao Zhang, Qi Alfred Chen, and Z Morley Mao. 2024. On data fabrication in collaborative vehicular perception: Attacks and countermeasures. In *33rd USENIX Security Symposium (USENIX Security 24)*. 6309–6326.
[87] Qingzhao Zhang, Xumiao Zhang, Ruiyang Zhu, Fan Bai, Mohammad Naserian, and Z Morley Mao. 2023. Robust Real-time Multi-vehicle Collaboration on Asynchronous Sensors. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*. 1–15.
[88] Xumiao Zhang, Anlan Zhang, Jiachen Sun, Xiao Zhu, Y Ethan Guo, Feng Qian, and Z Morley Mao. 2021. EMP: Edge-assisted Multi-vehicle Perception. In *ACM MobiCom*.
[89] Pengyuan Zhou, Wenxiao Zhang, Tristan Braud, Pan Hui, and Jussi Kangasharju. 2018. Arve: Augmented reality applications in vehicle to edge networks. In *Proceedings of the 2018 Workshop on Mobile Edge Communications*. 25–30.
[90] Xiao Zhu, Jiachen Sun, Xumiao Zhang, Y Ethan Guo, Feng Qian, and Z Morley Mao. 2020. MPBond: efficient network-level collaboration among personal mobile devices. In *Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services*. 364–376.
[91] Zhengxia Zou, Rusheng Zhang, Shengyin Shen, Gaurav Pandey, Punarjay Chakravarty, Armin Parchami, and Henry X Liu. 2022. Real-time full-stack traffic scene perception for autonomous driving with roadside cameras. In *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 890–896.