RECAP: 3D Traffic Reconstruction

Christina Suyong Shin
University of Southern California
USA

Weiwu Pang
University of Southern California
USA

Chuan Li
General Motors
USA

Fan Bai
General Motors
USA

Fawad Ahmad
Rochester Institute of Technology
USA

Jeongyeup Paek
Chung-Ang University
South Korea

Ramesh Govindan
University of Southern California
USA

Abstract

On-vehicle 3D sensing technologies, such as LiDARs and stereo cameras, enable a novel capability, 3D traffic reconstruction. This produces a volumetric video consisting of a sequence of 3D frames capturing the time evolution of road traffic. 3D traffic reconstruction can help trained investigators reconstruct the scene of an accident. In this paper, we describe the design and implementation of RECAP, a system that continuously and opportunistically produces 3D traffic reconstructions from multiple vehicles. RECAP builds upon prior work on point cloud registration, but adapts it to settings with minimal point cloud overlap (both in the spatial and temporal sense) and develops techniques to minimize error and computation time in multi-way registration. On-road experiments and trace-driven simulations show that RECAP can, within minutes, generate highly accurate reconstructions that have 2× or more lower errors than competing approaches.

CCS Concepts

• **Networks** → **Cyber-physical networks**; • **Computer systems organization** → *Special purpose systems*.

Keywords

Collaborative Sensing, Accident Traffic Reconstruction, Iterative Closest Point

ACM Reference Format:
Christina Suyong Shin, Weiwu Pang, Chuan Li, Fan Bai, Fawad Ahmad, Jeongyeup Paek, and Ramesh Govindan. 2024. RECAP: 3D Traffic Reconstruction. In *The 30th Annual International Conference on Mobile Computing and Networking (ACM MobiCom '24), November 18–22, 2024, Washington D.C., DC, USA.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3636534.3690691

This material is based upon work supported in part by the National Science Foundation under Grant No. 1956445 and by a grant from General Motors.

1 Introduction

Sensing has long been an integral part of vehicular traffic management [41]. Road-embedded induction sensors, magnetometers, and above-road video cameras provide information about instantaneous traffic flow as well as aggregated views of traffic at intersections and major roadways. The commoditization of 3D sensing technologies, such as LiDARs and stereo cameras, will likely enable a novel capability, *3D traffic reconstruction*, which produces a *volumetric video* consisting of a sequence of 3D frames. Each frame captures traffic participants (*e.g.,* vehicles, pedestrians, or bicyclists) and the surrounding scene in three dimensions. The volumetric video then captures, also in 3D, the time evolution of traffic. In this sense, 3D traffic reconstruction produces a *live digital twin* of traffic at an intersection or on the roadway.

3D traffic reconstruction can improve traffic safety and traffic management. For instance, it can aid *accident reconstruction*, in which certified professionals attempt to reconstruct the scene of an accident *post-facto*. In contrast, an accurate 3D traffic reconstruction can provide *direct* views of traffic dynamics before, during, and after the accident. As such, at-fault assessment is likely to be made much easier with this capability. 3D traffic reconstruction can also be used for roadway planning and other traffic management tasks (§2).

Motivated by the imminent availability of 3D sensors on vehicles, we consider ***opportunistic*** 3D traffic reconstruction, which combines 3D sensor data from vehicles that choose to participate to enable the capability.

We describe the design and implementation of RECAP, a system that provides opportunistic 3D traffic reconstruction, which, to our knowledge, no prior work has considered.

In RECAP, vehicles stream compressed 3D sensor data to a cloud service that performs 3D traffic reconstruction. Especially for accident reconstruction, accuracy is the primary requirement. RECAP must also ensure coverage and relatively quick time-to-reconstruction (§2.1).

RECAP's core problem is to *fuse* 3D frames taken from different moving vehicles' 3D sensors at each instant accurately and quickly. A 3D frame from a LiDAR, for instance, is represented by a *point cloud*, and point cloud fusion converts each point's position to a common frame of reference and merges all the points together to produce the fused 3D frame. Unfortunately, potential approaches for transforming all LiDARs to a common coordinate system using GPS or on-board high-definition (HD) maps are insufficient as are alternative reconstruction approaches such as photogrammetry (§2.2).

Point cloud registration [10, 12, 14] can accurately convert point clouds (pairwise or multi-way) to a common frame of reference. However, early approaches to pairwise registration use iterative search methods that can be compute-intensive and susceptible to local minima. More recent data-driven approaches generalize poorly. Multi-way registration approaches [14] use pairwise registration and can be adversely affected by pairwise registration error. Finally, by itself, point cloud registration cannot ensure coverage; registration requires point cloud overlap, and the faster vehicles move, the shorter the time during which their point clouds overlap, resulting in shorter duration of reconstructions.

To overcome these challenges, RECAP makes three main contributions (§3):

- An ***overlap-scoped registration*** technique that uses position hints to determine point cloud overlap, then runs registration *only* on points in the overlapped region to ensure more accurate *and* faster registration.
- A ***temporal expansion*** technique that expands point clouds over time to create more overlap opportunities resulting in longer duration reconstruction with coverage.
- A ***participant selection*** technique that determines those vehicles whose point clouds are likely to enable high-quality and wider-coverage multi-way registration.

Evaluations (§4) based on both real-world experiments and photo-realistic simulations demonstrate that RECAP can achieve accurate traffic reconstruction for different traffic scenes and traffic density. It can generate highly accurate reconstructions that have 2× or more lower errors compared to several position-based fusion and state-of-art registration algorithms. It generates reconstructions within minutes, and also increases the spatial coverage of reconstruction scenes by nearly 40% in some cases using temporal expansion.

2 Motivation, Approach and Background

In this section, we describe the motivation underlying RECAP, the problem it attempts to solve and its approach, and the shortcomings of alternative approaches.

2.1 Problem and Motivation

Vehicles are starting to have on-board depth perception (or 3D) sensors, such as LiDAR or stereo cameras, for advanced driving assistance (ADAS) or to achieve different levels of autonomy [70]. Most vehicles already have 4G LTE connectivity and will likely include 5G connectivity in the near future.

In this work, we consider a capability based on connected vehicles with 3D sensors that has not, to our knowledge, been explored before in the literature. Consider the scenario shown in Fig. 1 in which three LiDAR-equipped vehicles are moving towards an intersection (Fig. 1(a)). As each vehicle moves, its LiDAR sensor outputs (at 10-20 Hz) 3D views (LiDAR frames) of other vehicles as well as pedestrians and bicyclists at the intersection (Fig. 1(b)). Each LiDAR frame is a point cloud, where each point represents a reflection of one of the LiDAR's laser beams; each point is associated with a 3D position, as well as other attributes such as intensity.

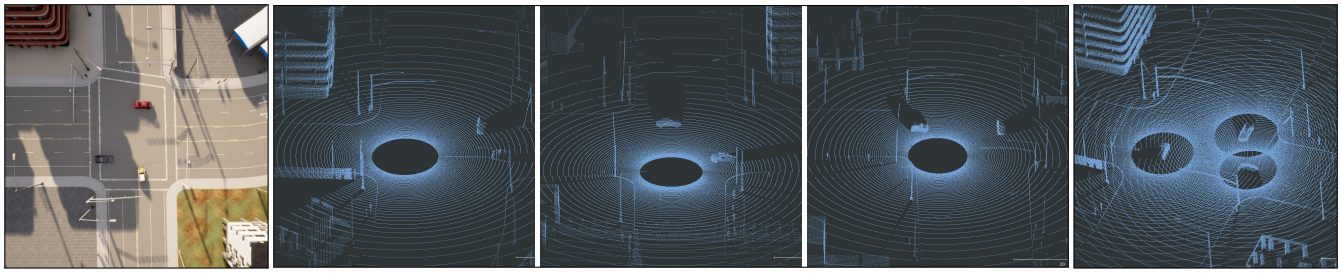
If each vehicle compresses successive LiDAR[1] frames and transmits these point cloud streams to the cloud, servers in the cloud can decompress these streams on the fly. They can then fuse each LiDAR frame from a vehicle with frames captured from other vehicles at the same instant to produce a composite 3D view of traffic dynamics (Fig. 1(c)). A collection of successive 3D views results in a volumetric video that represents the movement of traffic participants in 3D over time.

We call this capability opportunistic 3D traffic scene reconstruction (or 3D traffic reconstruction, for short), because it opportunistically leverages on-board 3D vehicular sensors. Our approach relies on vehicles willing to stream point clouds to cloud infrastructure. There may be several ways to incentivize this. A navigation app (*e.g.,* Waze) can provide advanced navigation assistance, or an insurer can provide a discount in exchange for streamed point clouds. We have left an exploration of incentives to future work.

Applications. Two applications motivate 3D traffic reconstruction.

Accident Reconstruction. Most traffic accidents result in a police report prepared by a police officer at the scene who uses visual inspections and witness statements to produce the report. When an accident involves a legal dispute [24], the disputants sometimes employ a certified [3] accident reconstructionist [47] who attempts to accurately re-create the accident scene and other conditions before and after the accident. The reconstructionist[2] obtains these reconstructions from: an assessment of the damage to vehicles, reports of witnesses at the scene, crash recorders on vehicles that record speed, throttle, position and other internal vehicle sensors [68]. From an assessment of damage, the reconstructionist can use software [31] to estimate the change in speed [39] as a result of the collision. Witness statements provide additional context

[1]In this work, we focus on LiDARs because they generate sparser point clouds than stereo cameras [52]. With 5G, it may be possible to stream denser point clouds from stereo cameras; we leave it to future work to explore this.
[2]Accident reconstruction is a mature field with several textbooks on the subject [6, 22, 28, 67, 68]; we describe it briefly in this paper.



(a) (b) (c)

FIGURE 1: (a) Bird's-eye view of three vehicles in an intersection. (b) Three point clouds from three vehicles. These single views are vulnerable to some occlusions. (c) Fused point cloud. It captures more complete traffic scene than a single vehicle point cloud.

in terms of the presence and behavior of other traffic participants (*e.g.,* the presence of pedestrians or bicyclists, or the erratic behavior of other vehicles). The output of these efforts is often a video [1, 19, 74] that approximately re-creates the entire scene leading up to the accident as well as the aftermath of the accident [74].

Unlike this approach, 3D traffic reconstruction can provide a more ***direct*** reconstruction of the event using vehicular sensors. It is likely to be more accurate than manual reconstruction because: inferring vehicular behavior from damage assessments requires solving an inverse problem that depends on accurate models of vehicle kinematics and structure; not all participants in the crash will have crash recorders (*e.g.,* pedestrians, bicyclists, or motorcyclists); and witness statements can be highly unreliable [68].

Traffic Analytics. The volumetric video produced by 3D traffic reconstruction (a *3D traffic video*) contains, at each frame, the position of all visible participants including vehicles, pedestrians, and bicyclists. From this, it is possible to infer speeds and headings of every participant at each instant in time. A collection of 3D traffic videos is a source of very detailed traffic analytics and can be used, for example, to: (a) test autonomous driving algorithms and neural networks under different conditions; (b) benchmark cooperative perception [52, 83] techniques in autonomous driving; (c) simulate different intersection designs [37] and analyze the impact of signal phase and timing (SPAT [2]).

Goal and Requirements. Consider N vehicles approaching an intersection. At time t, vehicle i's LiDAR generates a frame $f_i(t)$. The goal of 3D traffic reconstruction is to generate $F(t)$, a point cloud *fused* from each $f_i(t)$. A sequence of $F(t)$s over time results in a volumetric video.

Motivated by the applications discussed above, we impose four requirements on 3D traffic reconstruction:

Accuracy. 3D traffic reconstruction must achieve high accuracy for $F(t)$. The accuracy of a point cloud represents how closely it represents the ground-truth scene in 3D. More precisely, we define reconstruction error as an average distance between a point in our reconstructed point cloud and a corresponding point in the ground-truth point cloud [61]. Autonomous vehicles can position themselves using their sensors to within 10-20 cms, so we require that 3D traffic reconstruction also achieves this level of performance. Moreover, accuracy is paramount for accident reconstruction: accurate reconstruction ensures accurate trajectory reconstruction.

Coverage. 3D traffic reconstruction must achieve these reconstruction accuracies over as large a spatial region as possible. Put another way, suppose that there are two solutions to the 3D traffic reconstruction problem, one of which generates an $F(t)$ that spans an area a_1 and another that spans a larger area a_2, the latter solution is strictly preferred. This requirement ensures that participant trajectories are visible over a longer time duration during accident reconstruction and also ensures that analytics applications have as much data on participant movements as possible.

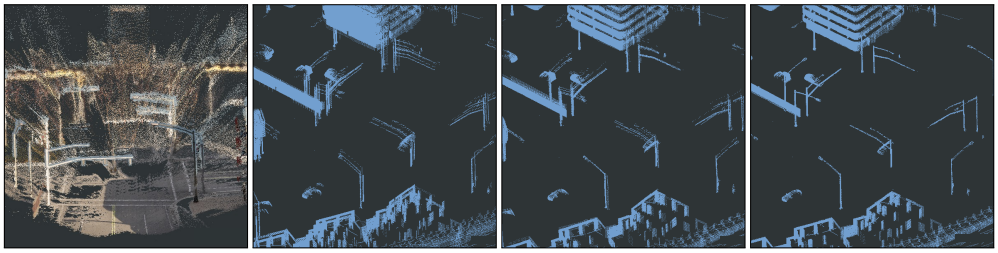
Time-to-reconstruction. The process of reconstruction must complete relatively quickly, so the police officer writing the accident report can use the reconstructed 3D traffic video. While police response times vary widely, average response times are usually under 10 mins [23], so RECAP should target reconstructions within minutes. With this requirement, 3D traffic videos will also be available to accident reconstructionists who work on a longer time scale.

Non-reliance on HD Maps. For a reason discussed below, a solution must not require HD maps but can use them when available to increase reconstruction accuracy.

2.2 Alternative Approaches

In this section, we consider several alternative approaches to 3D traffic reconstruction and demonstrate that they fail to meet at least one of the requirements listed above.

Photogrammetry. Photogrammetry is a commonly used approach for 3D reconstruction. It infers 3D models of objects using multi-view stereo (MVS) [30], which infers the structure (position and orientation) of the scene from the set of 2D images. Using this structure and stereo correspondence from multiple images, these algorithms estimate the depth of every pixel in each image resulting in a dense 3D reconstruction of



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

FIGURE 2: (a) Point cloud from ColMap. Moving objects (*e.g.,* vehicles or pedestrians) are absent in the point cloud. (b) Point cloud from GPS+IMU pose (error: ***1.13m***). (c) Point cloud from HD map positioning (error: ***20cm***). Both (b) and (c) have blurry alignments while (b) is worse than (c). (d) Ground-truth point cloud.

the entire scene/object. In our setting, photogrammetry might be employed by taking a succession of images from each vehicle v_i, and using these to infer $F(t)$. Unfortunately, photogrammetry is suitable for reconstructing static scenes/objects, not for tracking the position of moving objects, and can be inaccurate when used for the latter. Moreover, photogrammetry often takes several hours to complete.

To validate this, we used the CARLA simulator [25] in which, at a traffic intersection, we spawned several moving vehicles and pedestrians, and collected a sequence of images from vehicles' on-board cameras. For a 3D reconstruction with images, we used ColMap [59], a popular open-sourced photogrammetry implementation. After feeding images from a selected vehicle, the resulting 3D reconstruction (Fig. 2(a)) could capture static objects (*e.g.,* buildings or traffic lights) but it could not capture any moving objects (*e.g.,* vehicles or pedestrians). Thus, photogrammetry-based reconstruction cannot be used for 3D traffic reconstruction. Moreover, time-to-reconstruction was more than 4 hours to process 115 images from one vehicle.

GPS+IMU. In this approach, the vehicle can use its GPS position and IMU orientation to transform all LiDAR points into the GPS coordinate system. Then, it can merge all of the points from each LiDAR frame of each vehicle to obtain $F(t)$. This approach can be extremely fast, but is limited by the accuracy of on-board vehicular GPS.

To validate this, we conducted another experiment in CARLA (more details in §4). We spawned seven vehicles in a simulated traffic intersection and collected point clouds and GPS and IMU readings from each vehicle's sensors. Then, we transformed point clouds from each LiDAR frame of the vehicle to the GPS coordinate system, and merged all point clouds. Because GPS is inaccurate, the fused point cloud (Fig. 2(b)) is misaligned and looks blurry compared to the ground-truth fused point cloud (Fig. 2(d)). Reconstruction error (§2.1) of the point cloud in Fig. 2(b) was ***1.13 m*** but it took only 1 minute to generate a 3D volumetric video consisting of 150 fused point clouds.

HD Maps. A similar approach to estimating $F(t)$ is to use a high-definition map. Autonomous vehicles use this HD map to estimate their own position by comparing their sensor views against the pre-computed on-board map. Each vehicle can then transform each point in its LiDAR frame to the HD map's coordinate frame of reference. If all vehicles have the same HD map, the composite view results by simply merging all points from each LiDAR frame of each vehicle. This approach is both accurate and fast, since HD maps provide accurate localization and support fast lookup.

To validate this, we conducted an experiment in CARLA that uses HDMap-computed positions. Because HD map positioning is more accurate than GPS, the fused point cloud (Fig. 2(c)) shows traffic more clearly than the GPS+IMU approach. Reconstruction error of the point cloud in Fig. 2(c) was ***20 cm*** and it also took around 1 minute to generate a 3D volumetric video consisting of 150 fused point clouds. Even with more accurate positioning, the fused point cloud is visibly misaligned compared to the ground-truth stitched point cloud (Fig. 2(d)).

Practicality of HD Maps. Of these approaches, HD maps come closest to satisfying the requirements described above. However, the use of HD map technology in future autonomous driving is uncertain. The cost of collecting and updating HD maps over large geographical regions is proving to be expensive [21]. For this reason, some vehicle manufacturers reportedly do not use HD maps [40]. Moreover, recent research has demonstrated that it may be possible to do navigation for autonomous vehicles without any maps at all [73], so in the future, vehicles might not need HD maps at all. This motivates our non-reliance on HD maps requirement described in §2.1.

2.3 Approach and Background

Consider two point clouds A and B which represent distinct but overlapping views (respectively, v_A and v_B) of a scene. Point cloud registration is the process of estimating, using the point clouds, the rigid transformation T from v_A to v_B. Using the matrix T, one can transform the positions of points in A

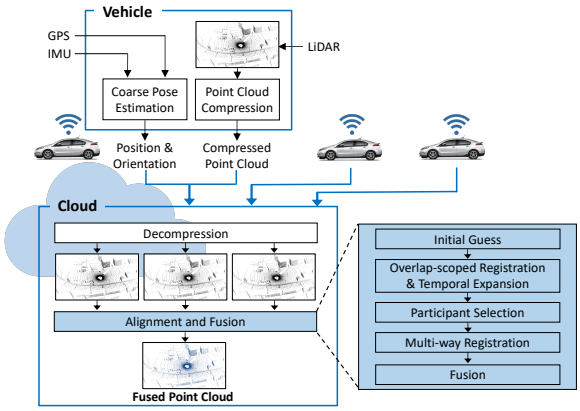


FIGURE 3: RECAP architecture: In RECAP, vehicles continuously transmit compressed point clouds to a cloud that performs 3D traffic reconstruction.

into B's coordinate frame of reference (alignment) and can then simply merge the two point clouds to stitch or fuse the point clouds. We use this basic idea to generate $F(t)$.

The most well-known approach for point cloud registration is Iterative Closest Point (ICP) [9, 12], a technique that, given an initial guess for T, iteratively refines the transformation matrix to minimize the distance between the closest points from the two point clouds within the area of overlap. ICP is known to be sensitive to the choice of initial guess [56].

ICP aligns two point clouds. In our setting, there may be multiple vehicles, each of which generates a point cloud. Prior work [14] has proposed an approach to register and align multiple views into a single global coordinate frame of reference. This multi-way registration first computes pairwise transformations between every pair of point clouds, then optimizes a pose graph whose nodes are the individual views and whose edges are the transformations between them.

3 RECAP Design

Overview. Consider a scenario in which multiple vehicles are on a road segment or at an intersection. Fig. 3 describes how RECAP works. Each vehicle continuously transmits its compressed LiDAR point clouds over the cellular network to a cloud service. The latter (a) extracts individual LiDAR point clouds from the compressed stream from each vehicle, then (b) performs multi-way registration using multi-view ICP (§2.3) at each instant t on point clouds (or frames) $f_i(t)$, $i \leq N$. The output of these steps, at t, is a fused point cloud frame $F(t)$ (Fig. 1(c)). Successive fused point clouds form a volumetric video[3] comprising an accurate 3D traffic reconstruction.

RECAP performs 3D traffic reconstruction using several components on the vehicle and the cloud (Fig. 3).

Vehicle. On-board a vehicle, RECAP runs two components. *Coarse pose estimation* roughly estimates the current position and orientation (*i.e.,* pose) of the vehicle at each instant. *Point cloud compression* compresses each point cloud generated by the vehicle's LiDAR. This is necessary because raw LiDAR data can be voluminous. For instance, Ouster *OS1-64* LiDAR generates 10 point clouds per second requiring almost 240 Mbps without compression. Fortunately, off-the-shelf compression techniques can reduce this by almost two orders of magnitude [58] to about 10 Mbps, well within LTE speeds in today's deployments (§4).

Cloud. RECAP's cloud service decompresses the received stream of point clouds. It uses the coarse pose estimate for *overlap-scoped registration* which registers only points in the overlapped region of two point clouds from two different vehicles. RECAP, if necessary, performs *temporal expansion*, which uses multiple temporal point clouds from a single vehicle to produce a larger point cloud increases reconstruction coverage. Finally, when there are multiple vehicles on a road segment or at an intersection, RECAP's *participant selection* component finds the subset of these vehicles that will ensure reconstruction accuracy.

These components perform three qualitatively different tasks: (a) overlap-scoped registration increases the accuracy and speed of pairwise registration, (b) temporal expansion increases the coverage of registration and (c) participant selection increases the accuracy of multi-way registration. The following subsections discuss these.

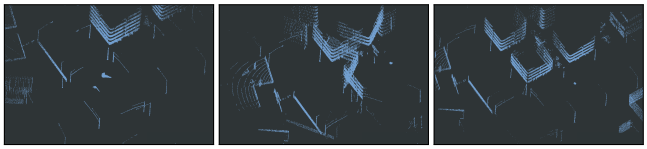
Before doing this, we briefly discuss our choice of building block for registration. As described in §2.3, RECAP uses a well-known ICP algorithm [10]. Recent work has made impressive strides in training neural nets for registration [8, 15, 29, 33, 36, 78]. These are usually trained using point clouds captured from a single moving vehicle (*e.g.,* from the KITTI [32] dataset). A highly accurate neural network for point cloud registration, PointDSC [7], does not generalize to our setting, in which we attempt registration of point clouds from different vehicles captured at the same instant. We demonstrate this experimentally in §4.7.

3.1 Overlap-scoped Registration

Initial Guess. ICP's [10] performance and accuracy depends heavily on the *initial guess* for the transformation [55, 56]. A poor initial guess can lead to local minima resulting in poor reconstruction, and/or take many iterations to converge. In some settings, it is possible to obtain good initial guesses. For example, when aligning successive point clouds obtained from a single moving vehicle, odometry can provide an initial guess, since a vehicle moves relatively short distances between point cloud captures (e.g. 10 Hz). In RECAP, this method does not work because we seek to align point clouds obtained from multiple moving vehicles at the same instant.

RECAP uses positions and orientations (poses) generated by its *pose estimator* (Fig. 3) to obtain an initial guess. For

[3]This video shows a bird's eye view of a 3D traffic scene reconstruction: https://youtu.be/SSTa3OhqiO4



(a) 0.07 m (b) 2.28 m (c) 0.32 m

FIGURE 4: Importance of overlap. (a) ICP applied to two point clouds with high overlap. The number below the figure shows reconstruction error (lower is better). (b) ICP applied to two point clouds with minimal overlap, resulting in high error. (c) ICP applied to two minimally overlapped point clouds after using overlap-scoped registration, resulting in lower error.

this, it can either use GPS and IMU sensors smoothed using a Kalman filter, or by matching its LiDAR output to a HD map using a fast registration algorithm [46]. Both these approaches result in comparable RECAP's reconstruction accuracy (§4).

RECAP's use of pose estimate as initial guess is, by itself, not novel; prior work [82] has suggested this, for example. However, we are not aware of any prior work that performs overlap-scoped registration: using the pose estimate to determine points in the region of overlap between two point clouds, and registering the point clouds using only those points.

Impact of Overlap. ICP and, more generally, registration algorithms that rely only on point cloud inputs work well when there is a significant overlap between the corresponding views. With a small overlap, ICP sometimes fails to produce accurate registration results. Fig. 4(a) shows the fused point cloud obtained using ICP from two LiDARs placed close enough to each other to have significant overlap. When we move the LiDARs far apart to reduce view overlap, the reconstruction is visibly worse (Fig. 4(b)). This occurs because ICP attempts to align every point in both point clouds, and points in non-overlapping parts of the view can confound the alignment. Moreover, ICP by itself has no way of knowing which parts of the two point clouds overlap.

In RECAP, the views of moving vehicles will, more often than not, have little overlap. Consider a LiDAR with a nominal range of 100 m (*e.g.,* Ouster *OS1*), and suppose two cars start from the same spot and drive away from each other at 30 mph. It is easy to show that, within 7 s, the overlap between their LiDAR point clouds reduces to less than 50%.

Overlap-scoped Registration. While vanilla ICP has no way to determine whether two point clouds overlap, we observe that RECAP can use its pose estimates as well as the nominal LiDAR range to identify points that lie in the overlapped region. For two point clouds p_i and p_j, this overlap-scoped registration works as follows:

- Use pose estimates from vehicles i and j to convert each point in p_i and p_j to GPS or HD map coordinates (depending on the pose estimator used).
- Let the nominal LiDAR range be r. For each point in p_i that (a) lies within a distance r of vehicle j ***and*** (b) has at least one point in p_j that is within δ_c of itself, add the point to $q_{i,j}$, i's overlapped point cloud with j. Similarly, compute $q_{j,i}$, j's overlapped point cloud with i.

Intuitively, $q_{i,j}$ consists of points in p_i that fall within the region of overlap between the two point clouds. Moreover, each point in $q_{i,j}$ has at least one point in p_j within a distance δ_c (we call this a *potential* correspondence). Instead of applying ICP to p_i and p_j, RECAP applies ICP to $q_{i,j}$ and $q_{j,i}$. This overlap-scoped registration has two benefits:

- Because $q_{i,j}$ contains only points in the overlapped region that have a potential correspondence with at least one point in j, ICP is likely to be more accurate than using the complete point clouds.
- $q_{i,j}$ is usually smaller than p_i, so ICP between the overlapped point clouds can be faster. ICP computational cost is a function of the point cloud size.

Fig. 4(c) illustrates how overlap-scoped registration can result in better reconstruction even with minimal overlap.

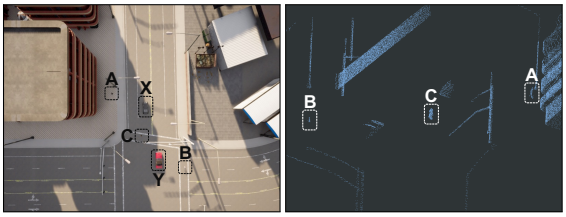
If the overlapped region is too small, ICP might converge to a local optimum, or fail completely. For this reason, we only consider overlaps whose size exceeds a specified threshold; we discuss this below (§3.2).

Removing Small Objects. Small objects in the scene (*e.g.,* pedestrians) can confound ICP. Consider two cars approaching each other while a pedestrian crosses a crosswalk in between them. Each vehicle's view overlaps with the other, but each vehicle captures a non-overlapping part of the pedestrian and ICP may not be able to align these. This occurs rarely, since overlap extraction can trim such objects. To optimize ICP accuracy, RECAP *crops* all points below a height h above the LiDAR before determining overlap and applying ICP. After running ICP, we obtain transformation matrices between vehicles. We generate fused point clouds with these matrices using the original point clouds. This ensures that the fused point cloud includes all objects including small ones.

To empirically determine h, we conducted an experiment in CARLA and explored the coverage, reconstruction error, and reconstruction time for different cropping thresholds h. We randomly chose the number of vehicles (from 3 to 13) and traffic scenes (4-way traffic intersection or T-junction), iterated the 3D traffic reconstruction over 150 frames three times, and averaged the reconstruction accuracy, spatial coverage, and reconstruction time to process 150 frames. As shown in Tbl. 1, a higher h can reduce coverage, but improve accuracy and reconstruction time because overlapped point clouds are

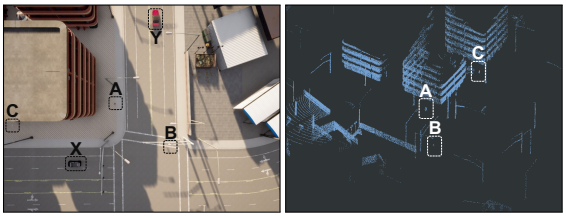
	No Crop	**0.5 m**	**1.5 m**	**2.5 m**
Avg. Coverage (m^2)	10,001	9,065	8,830	8,217
Avg. Reconstruction Error (m)	0.17	0.17	0.16	0.16
Avg. Reconstruction Time (min)	28.33	16.78	13.47	10.97

TABLE 1: Coverage, reconstruction error and reconstruction time as a function of cropping height.



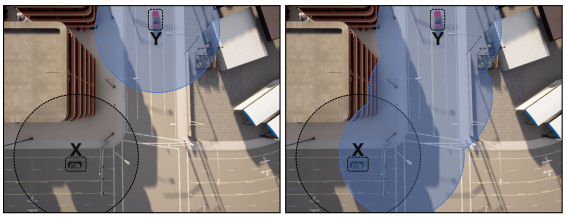
(a) (b)

FIGURE 5: (a) When two vehicles (**X**, **Y**) are near each other, after ICP, RECAP can find the relative position of pedestrians **A**, **B** and **C** and obtain a denser point cloud of **C**. (b) Generated point cloud.



(a) (b)

FIGURE 6: (a) When two vehicles (**X**, **Y**) are far away, RECAP can *expand* **Y**'s point cloud across time, then align it with **X**'s point cloud. (b) In the composite point cloud, RECAP can find the relative positions of **A**, **B** and **C** after temporal expansion.



(a) Before (b) After

FIGURE 7: Before and after temporal expansion of vehicle **Y**.

smaller. h of 0.5 m is a sweet spot in this trade-off space; future work can explore more advanced strategies.

3.2 Temporal Expansion

Limited Duration Visibility. Consider Fig. 5, which shows two vehicles **X** and **Y**, and three pedestrians **A**, **B** and **C**. In Fig. 5(a), the two vehicles are close enough to each other so that their point clouds overlap. **A** is within range of **X**, **B** is within range of **Y**, and **C** is within range of both vehicles. After applying overlap-scoped registration (§3.1), RECAP can: (a) find the relative positions of the three pedestrians; (b) create a denser point cloud for **C** by combining the points corresponding to **C** from each vehicle's point cloud (Fig. 5(b)).

However, as vehicles move away from each other, the degree of overlap decreases to zero very quickly, beyond which point relatively positioning other objects becomes infeasible. We call this *loss of coverage*. For example, a vehicle moving away from another stationary vehicle at 30 mph only has coverage for about 15 s.

Algorithm 1: Efficient Temporal Expansion

Input : A sequence of successive point clouds $\{p_1, \cdots, p_n\}$ from a single vehicle.
Output: A fused point cloud p_F which combines the input point clouds.

```
/* Initialize loop indices & fused point cloud   */
i ← 1, j ← i + 1, p_F ← p_1;
while i < n do
    /* Find j with correspondence count < ρ      */
    if c_{i,j} ≥ ρ then
        j ← j + 1;
        continue;
    end
    /* If j reaches the end, run the last ICP    */
    if j > n then j = n + 1;
    /* Run ICP and update fused point cloud      */
    T = icp(p_F, p_{j-1});
    p_F ← fuse(p_{j-1}, transform(p_F, T));
    i ← j − 1, j ← i + 1;
end
return p_F
```

Temporal Expansion. RECAP recovers loss of coverage using temporal expansion. To illustrate this, consider Fig. 6(a). In this figure, assume that vehicles **X** and **Y** were, at some time t' in the past, near each other and that **Y** has moved away so that, at time t, there is no longer any overlap between the two vehicles' point clouds.

Temporal expansion aligns each point cloud of **Y** between t' and t ($p_{Y-t'}$, ..., p_{Y-t}) to create a *temporally expanded point cloud* (Fig. 7). This expanded point cloud will have sufficient overlap with **X**'s point cloud at t so RECAP can apply overlap-scoped registration to find the relative positions of **A**, **B** and **C** (Fig. 6(b)). In a more general scenario, RECAP might apply temporal expansion for **X** as well. During temporal expansion also we exclude small objects as discussed in §3.1.

Because it uses point clouds retroactively, the above example illustrates backward expansion. RECAP also can perform forward expansion. At time t, if **Y** is moving towards **X** and meets (or is near to) it at time t', RECAP can use **Y**'s point clouds between t and t' to proactively expand **Y**'s point clouds. RECAP does not attempt expansion when two vehicles never meet in the past or the future.

To determine t', RECAP finds the time in the past (backward expansion) or the time in the future (forward expansion), where the number of points in the overlap between **X** and **Y**'s point cloud is the highest. To do this, it continuously tracks overlaps between each pair of vehicles at every instant.

Expansion can be computationally intensive. LiDARs generate point clouds at 10 Hz, and aligning every point cloud using ICP can increase time-to-reconstruction (§2.1). To reduce this, RECAP minimizes the number of ICP invocations in the temporal expansion, as described next.

Efficient Expansion. To minimize ICP invocations, RECAP relies on an empirical observation: as long as the number of points in the overlapped point cloud ($c_{i,j}$= $|q_{i,j}|$ in §3.1) exceeds a threshold ρ, ICP can accurately align the corresponding point clouds. Below this threshold, ICP often fails (§4.8). Thus, for each vehicle, RECAP tracks, for each successive pair of point clouds p_k and $p_{k'}$, $c_{k,k'}$[4], the number of points in the overlapped point cloud between the k-th point cloud and the k'-th point cloud from the *same* vehicle.

Now, suppose we are trying to expand a vehicle **Y** between time t' (in the past) and t. RECAP starts at t' and finds the furthest time $t_1 < t$ such that $c_{t',t_1} \geq \rho$. In other words, it finds the furthest t_1 such that ICP can register the point cloud at t' with that at t_1. It then fuses the two point clouds, and then repeats this step: *i.e.,* it finds the furthest $t_2 < t$ from t_1 such that $c_{t_1,t_2} \geq \rho$, and so on. Algorithm 1 formalizes this. The input to this algorithm is a sequence of successive LiDAR frames p_i from a vehicle, and its output is a single temporally expanded frame p_F that fuses these frames with as few ICP invocations as possible. The algorithm works as follows:

- Initially, p_F is assigned p_1, the first frame in the sequence (line 1). Variables i and j track the frames between which RECAP performs ICP.
- In the `while` loop (lines 2-11), RECAP increments j to the first point where an ICP between i and j would likely fail because of too few correspondences. At that point, it performs `icp()` (line 8) between p_F and p_{j-1} (the furthest frame likely to result in accurate ICP), and fuses p_F with p_{j-1}, updating p_F with the result (line 9).
- The `while` loop terminates when i reaches n (line 7).

Backward and Forward Expansion. As described earlier, RECAP can expand both backward and forward in time. It can do this by feeding point cloud frames to Alg. 1 in the appropriate order. For example, to expand backwards at time t to t', RECAP would use as p_1 the point cloud at t' and as p_n the point cloud at time t. To expand forwards at time t to t', RECAP would use as p_1 the point cloud at t and as p_n the point cloud at time t'.

Why Not SLAM? RECAP could have also used SLAM (*e.g.,* FAST-LIO2 [77]) to align temporal point clouds from a single vehicle. However, for accident scene reconstruction, SLAM is less accurate, as demonstrated in §4. FAST-LIO2 uses scan-matching to iteratively register incoming 3D point clouds using ICP. For real-time operation, FAST-LIO2 limits the number of ICP iterations. Thus, it can align point clouds faster but at the cost of registration accuracy. RECAP can run ICP for longer, so has relatively higher registration accuracy.[5]

[4]For ease of exposition, we abuse notation. Subscripts sometimes refer to vehicles, and sometimes to LiDAR frames across time. The correct interpretation will be evident from the context.

[5]For short traces, as is the case for expansion. Over longer traces, FAST-LIO2 will outperform RECAP because it has sophisticated drift mitigation.

	All Reliable	All Unreliable	RECAP
Reconstruction Error (m)	1.38	1.38	0.10

TABLE 2: Reconstruction error from pose graph estimation.

Pairwise ICP with Expansion. RECAP combines overlap-scoped registration and temporal expansion between each pair of vehicles, at each instant in time. Consider a pair of vehicles (i, j) at time t. Let p_i and p_j be the point clouds for i and j at time t. RECAP combines these two steps as follows:

1. Extract overlapped point clouds $q_{i,j}$ and $q_{j,i}$ (§3.1).
2. If the count of potential correspondences is more than ρ, run ICP between $q_{i,j}$ and $q_{j,i}$ (ICP without expansion).
3. Else if i and j met at time t', expand p_i and p_j till time t' using Alg. 1, and, if the expanded point clouds have sufficient potential correspondences, run overlap-scoped registration (§3.1) on the expanded point clouds.

At this point, at each instant t, RECAP runs pairwise ICP to generate the transformation matrix $T_{i,j}$ for each vehicle pair.

3.3 Improving Multi-way Registration

Pose Graph Optimization. Each transformation matrix obtained from pairwise ICP describes how to transform the pose of one vehicle to the coordinate frame of reference of the other vehicle. However, RECAP needs to transform the pose of each of these vehicles into a *global* coordinate frame of reference: it needs to find, for each vehicle i, a transformation matrix T_i that converts i's pose to this global frame of reference. Prior work by Choi *et al.* [14] has developed an optimization technique which searches for the best T_i that minimizes overall distance between correspondences. This work has applied this problem to reconstructing indoor scenes from RGB-D images captured by a single camera traversing the scene over time. We adapt this algorithm to work in our setting: multi-way registration of point clouds captured at the same instant in time from spatially distributed vehicles.

Impact of ICP Error. To determine T_i, we adapt [14] to model the results of pairwise ICP (§3.2) as a pose graph, in which node i represents a vehicle, and edge i, j represents $T_{i,j}$. However, pose graph optimization results in high error if the input transformations have high error [15, 33]. To deal with this, [14] uses a heuristic: it assumes that ICP between successive point clouds in a single camera is reliable (has low error), but ICP between other point clouds may be unreliable. For these unreliable edges, their approach heuristically estimates the confidence in the edge, and prunes out low confidence edges. Because it obtains point clouds from different LiDARs, RECAP cannot use their heuristic for reliability. Simpler heuristics (assuming either all edges are reliable or all are unreliable) also result in poor accuracy (Tbl. 2).

Participant Selection. RECAP uses a different approach. ICP returns the number of actual correspondences found. If this number on edge i, j is higher than a threshold (μ, learnt from simulations), RECAP inserts the edge into the pose graph,

else it does not. Because pose graph optimization requires a clique, RECAP then finds the maximum clique in the resulting pose graph before running the optimization. Effectively, this approach finds the largest set of participants whose views are likely to result in low reconstruction error, so we call this *participant selection*. In doing so, RECAP trades-off a little spatial coverage (because it may use fewer participants) for accurate reconstruction.

4 Evaluation

We evaluate RECAP's performance using real-world experiments as well as traces collected from a simulator.

4.1 Methodology

Implementation. Our RECAP implementation uses the Point Cloud Library [57] for pairwise ICP, and Open3D [86] for pose graph optimization. Our implementation is 8,100 lines of code, not including the above libraries.

Experiments. We collect traces, then run RECAP on these on an Intel i9-9900K CPU (16 cores, 3.60 GHz).

Real-world Traces. To evaluate RECAP, we collect traces using sensors mounted on two vehicles. Each vehicle has either an Ouster *OS0-64* or an *OS1-64* and a mobile phone with GPS and gyroscope. We use SensorLog [62] to collect GPS and gyroscope data. To collect the ground-truth pose of each vehicle, we mount an Xsens GNSS *MTi-680G RTK*. We use this pose to transform each vehicle's point cloud to the GPS coordinate system, and fuse the point clouds to obtain the *ground-truth point cloud*.

Simulator Traces. We also collect traces using the CARLA [25] simulator which models realistic traffic environments, behaviors of traffic participants (*e.g.,* vehicles and pedestrians), and supports various sensors including GPS and IMU sensors. To simulate realistic LiDARs, we use CARLA's built-in model for the Ouster *OS1-128* LiDAR.

In CARLA, we spawn vehicles and pedestrians in three different intersection types: a ***4-way intersection***, a ***T-junction***, and a ***roundabout***. We focus on intersections for two reasons. First, at intersections or roundabouts, traffic dynamics are more complicated than on road segments, with vehicles arriving or departing in different directions, making reconstruction harder. Second, 50% of all serious accidents and 25% of fatalities [26, 27] occur at intersections. Since accident reconstruction is a key use case for RECAP, we chose these intersection types to demonstrate RECAP's performance in these settings. In each scene, as vehicles and pedestrians move, we collect on-board LiDARs, GPS, and IMU sensor data at 10 Hz. We also collect ground-truth poses from the simulator for computing ground-truth point clouds.

For each intersection type, we collect traces for different *scenarios* by varying the number of vehicles from 3 to 13; beyond this, the capacity of the intersection is exceeded and vehicles incur significant wait times. We average results over three runs that randomize the traffic flow (*e.g.,* initial positions, speeds, trajectories and types of vehicles).

	# of Vehicles					
Scheme	**3**	**5**	**7**	**9**	**11**	**13**
SAC-IA	2.59	8.16	8.08	8.37	8.70	9.02
FGR	3.22	6.89	6.54	7.84	7.81	9.47
Go-ICP	2.82	7.76	7.47	7.68	7.97	9.27
GPS+IMU	0.95	1.10	1.14	1.18	1.14	1.23
GPS+IMU+KF	0.62	0.68	0.69	0.71	0.68	0.83
HDMap	0.14	0.19	0.20	0.27	0.22	0.40
RECAP	0.07	0.10	0.12	0.13	0.12	0.15

TABLE 3: Avg. reconstruction error (m) of RECAP and baselines.

Metrics. We evaluate RECAP using three metrics (§2.1):

Reconstruction Error of a fused point cloud is the average distance between a point in the point cloud and its nearest point in the ground-truth point cloud. This is commonly used for evaluating reconstruction accuracy [61].

Spatial Coverage is the area of the projection of a fused point cloud on the ground plane. We use this metric in some experiments to demonstrate the importance of expansion.

Time-to-reconstruction measures the time RECAP requires to process a trace, assuming elastic cloud resources (since most RECAP components execute on the cloud, Fig. 3). Specifically, many steps in RECAP are parallelizable (*e.g.,* computing pairwise ICP or extracting the overlap between vehicle point clouds). For such steps, we estimate the time to run these in parallel, and our time-to-reconstruction metric measures the total time taken by RECAP for the trace.

Baselines. We compare RECAP against:

GPS+IMU uses GPS and IMU to fuse point clouds (§2.2).

GPS+IMU+KF applies a Kalman filter to GPS+IMU readings before fusing point clouds.

HDMap obtains a pose estimate by matching each point cloud with a pre-built HD map. In our implementation, we use NDT (Normal Distribution Transformation [46]), a variant of ICP to match a point cloud to the HD map. NDT is the most widely-used point cloud matching algorithm for LiDAR-based localization [13, 43, 69, 75]. Since each vehicle's pose estimate is already in the HD map's coordinate system, we simply merge all point clouds to obtain a fused point cloud.

SAC-IA [56] uses no extraneous sensor or map information, but finds correspondences between two point clouds using unique features of points (*e.g.,* point normals or the number of neighbors). For this reason, it can be robust to the degree of overlap. Additionally, it finds an optimal transformation between two point clouds using a subset of the point cloud, so registration accuracy is less impacted by non-overlapping regions (§3.1). We use SAC-IA to align pairwise point clouds and run pose graph optimization to align multiple point clouds.

Real-world Trace	A	B	C	D	E
Avg. Reconstruction Error (m)	0.17	0.26	0.21	0.15	0.23

TABLE 4: Avg. reconstruction error (m) from five real-world traces.

FGR [85] (or Fast Global Registration) relies on feature correspondences only, like SAC-IA. Unlike SAC-IA, it can be potentially faster, because it does not include RANSAC-based model proposal and evaluation. For this baseline, we run FGR to align pairwise point clouds and run pose graph optimization for multiple point clouds.

Go-ICP [79] is a variant of ICP that avoids local minima. Because it finds the globally optimal alignment, it is known to be more robust to initial guesses. For this baseline, we run Go-ICP for pairwise registration and run pose graph optimization for multiway registration.

4.2 Accuracy Comparison Against Baselines

The primary requirement for RECAP is high reconstruction accuracy (§2). So, we compare RECAP against other baselines (§4.1) on reconstruction error and show that it outperforms the baselines. For this experiment, we use a large 4-way intersection in CARLA.

Results. Tbl. 3 shows the average reconstruction error of RECAP and other baselines across all of these scenarios. *SAC-IA*, *FGR* and *Go-ICP*, because they do not use any extraneous information (*e.g.,* a pose guess), perform poorly with average errors on the order of nearly 9 m. Error standard deviations for these three schemes range from 1.10 to 2.72. Moreover, they have a time-to-reconstruction of more than 6 hours (even with elastic cloud resources, §4.1).

GPS+IMU incurs a reconstruction error of about 1 m. We model a GPS device in CARLA with a 2 m error, consistent with the best reported accuracy of commodity GPS devices installed on vehicles [11, 80], so 1 m error is not surprising. Applying a Kalman filter (*GPS+IMU+KF*) reduces the error by around 25% to around 75 cm. Recall that RECAP also uses Kalman-filtered GPS readings, but only as an initial guess. By contrast, *GPS+IMU+KF* uses the position estimate directly to fuse point clouds. Error standard deviations of GPS-based schemes are between 0.19 and 0.37.

The most accurate baseline, *HDMap*, achieves a reconstruction error of about 25 cm. It uses the position estimate directly to fuse point clouds, but is more accurate because HDMap-based positioning is more accurate than GPS. Its standard deviation of error ranged from 0.09 to 0.26.

RECAP improves on *HDMap* accuracy by about 2×. Across the different scenarios, RECAP's average accuracy is in the range of 10-15 cm with standard deviations of 0.07-0.17. As such, for accident reconstruction, where positioning accuracy is crucial, RECAP is clearly the most preferred alternative.

	# of Vehicles					
Scene Type	3	5	7	9	11	13
4-way Intersection	0.07	0.10	0.12	0.13	0.12	0.15
T-junction	0.12	0.10	0.14	0.15	0.11	0.11
Roundabout	0.06	0.10	0.12	0.10	0.09	0.07

TABLE 5: Avg. reconstruction error (m) for different traffic scenes.

Tbl. 3 shows the impact of traffic density on reconstruction error. RECAP's careful participant selection enables it to have low error even at high traffic density. Other approaches are independent of density, except for *SAC-IA* and FGR.

4.3 Real-world Experiments

We also collected five traces from on-vehicle LiDARs (§4.1) to verify that RECAP works on real-world traffic scenes. In trace **A**, collected on a university campus, two vehicles drive one behind the other. In **B**, collected on the same campus, the two vehicles drive towards each other. In **C**, collected in a neighborhood off-campus, the vehicles also drive towards each other. **D** and **E** are from a different neighborhood off-campus while **D** has two vehicles driving with same direction and **E** has two vehicles with opposite direction. We used the *OS0-64* for trace A, B and C and *OS1-64* for the rest.

Results. Tbl. 4 shows the average reconstruction error from above five real-world traces.[6] **A** and **D** exhibit 15-17 cm error. In these traces, since the vehicles are traveling in the same direction, RECAP does not need to invoke expansion. For traces **B**, **C** and **E**, in which vehicles move in opposite directions, the error is slightly higher (26 cm, 21 cm and 22 cm respectively, standard deviations range from 0.08 to 0.24). This is because expansion increases error slightly while increasing spatial coverage (§4.6). While these errors are in the ballpark of reconstruction errors from simulation (§4.2), they are consistently a few centimeters higher. We attribute this to errors in GNSS-RTK for ground-truth poses — commercial receivers are known to exhibit 5-10 cm in urban environments [38].

4.4 Different Intersection Types

To verify that RECAP can achieve consistent performance across different intersection types, we compute, from simulator traces, reconstruction error for the three types of intersections listed in §4.1 and shown in These three types of intersections exhibit qualitatively different traffic patterns.

Results. Tbl. 5 shows the average reconstruction error across different numbers of vehicles. Despite the differences in traffic dynamics, RECAP exhibits low reconstruction error across these intersection types (of less than 15 cm). This suggests that it is robust to variations in the way traffic approaches and navigates an intersection. The standard deviations of error were 0.07-0.17 for the 4-way Intersection, 0.08-0.15 for the T-junction, and 0.03-0.14 for the Roundabout.

[6] Generated 3D traffic videos at https://www.youtube.com/@RECAP-o2p

Component	# of Vehicles		
	3	7	13
Overlap + Expansion (p)	1004.51	2709.54	7100.13
Pairwise ICP (p)	345.19	441.78	556.24
Decompression (v)	76.10	81.29	88.75
Initial Guess + Crop (v)	1.99	2.29	2.55
Selection + Pose Optimization	1.14	1.50	2.55
Total	1428.94	3236.40	7750.22

TABLE 6: Avg. latency per frame for each components (ms).

	# of Vehicles		
Scene Type	3	7	13
4-way Intersection	3.65 (2.24)	10.13 (2.84)	15.33 (3.11)
T-junction	4.77 (2.30)	8.47 (2.99)	21.39 (6.32)
Roundabout	2.30 (2.17)	5.66 (3.26)	14.97 (5.61)

TABLE 7: Avg. time-to-reconstruction (minutes) for 150 frames.

4.5 Time-to-reconstruction

In these and subsequent experiments, we show results only for 3, 7 and 13 vehicles for brevity. We begin by exploring the breakdown of latency for different components of RECAP.

Latency Breakdown. The breakdown of average latency for the time to process a single frame by components is shown in Tbl. 6. In this table, some components execute once per vehicle (**v**) during a frame. Examples include point cloud decompression, or initial guess estimation. Each instance of execution can be parallelized, and the time shown is the average across all frames of the maximum time to execute the component for one vehicle. Other components execute once per pair of vehicles (**p**). These include overlap extraction and pairwise ICP. For these too, we calculate the latency as the average across all frames of the maximum per pair execution time. Finally, some components are entirely sequential, such as participant selection and pose optimization.

Tbl. 6 shows that the dominant components (overlap extraction/expansion and pairwise ICP) are the ones most easily parallelizable. Other components are minuscule by comparison. In particular, the sequential components that operate across all vehicles (participant selection and pose optimization) together take a few milliseconds.

Average Time-to-reconstruction. This is time to process each trace of the different intersection types with different number of vehicles.

Tbl. 7 shows that reconstruction time is less than 22 minutes across all of the scenarios we have explored (assuming that all parallelizable components are actually executed in parallel, §4.1). With more vehicles, RECAP needs higher time-to-reconstruction because expansion is invoked more frequently because of much more complex traffic dynamics. This demonstrates that, at least for the cases we have explored, RECAP can generate a reconstruction at the timescales required for a police officer to respond to and process an accident. For instance, at least one major city in the US aims to respond to emergencies within 10 minutes [23]. Thereafter, police likely need 10-15 minutes to process the scene (interview witnesses, take photographs, redirect traffic *etc.*). The standard deviations of time-to-reconstruction were 1.60-6.46 for 4-way Intersection, 0.70-11.43 for T-junction, and 0.22-1.96 for Roundabout.

For faster reconstruction, RECAP can also generate a reconstruction *without* expansion. This has lower coverage (§4.6), but can be more accurate and run *much* faster (within 5 minutes in most cases, numbers in parentheses in Tbl. 7). Finally, RECAP can be optimized further (*e.g.,* by using the GPU to run ICP, or by minimizing expansion in dense traffic); we have left this to future work.

# of Vehicles	3	7	13
With Overlap	3.65	10.13	15.33
Without Overlap	5.47	17.48	19.54

TABLE 8: Avg. time-to-reconstruction (minutes) with and without overlap extraction.

# of Vehicles	3	7	13
With Expansion	7,387	10,568	12,434
Without Expansion	5,928	6,647	7,754

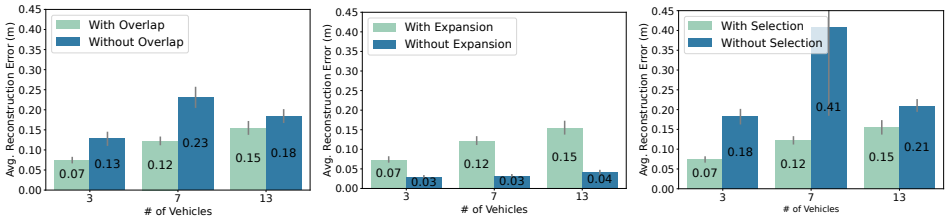
TABLE 9: Avg. spatial coverage (m^2) with and without expansion.

4.6 Ablations

Overlap-scoped Registration. To demonstrate the benefits of overlap-scoped registration, we compare reconstruction error (Fig. 8(a)) and time-to-reconstruction (Tbl. 8) with and without this technique. Overlap-scoped registration consistently reduces error, sometimes by up to 2×; it also reduces time-to-reconstruction by 1.27-1.7× in our experiments.

Temporal Expansion. To quantify the benefit of expansion, we compare reconstruction error (Fig. 8(b)) and spatial coverage (Tbl. 9) with and without expansion. We compute coverage (§4.1). RECAP is able to cover 19.7%, 37.1% and 37.6% more space with expansion enabled. However, expansion can increase reconstruction error; without expansion, reconstruction error can be below 5 cm, 2× or more lower than with expansion. Whether the increase in spatial coverage is worth the impact in accuracy can depend on the use-case for the reconstructed data. A user of RECAP can selectively enable expansion when greater coverage is desired.

Participant Selection. Fig. 8(c) shows that without participant selection, reconstruction error can be as high as 41 cm in some scenarios. For the three scenarios that we evaluate, RECAP selects 2.5, 4.5 and 6.9 vehicles respectively on average per frame. Thus, without participant selection, RECAP can be adversely impacted by traffic density, so selection is crucial for ensuring reconstruction accuracy.



(a) (b) (c)

FIGURE 8: Avg. reconstruction error (m) with and without (a) overlap-scoped registration, (b) temporal expansion, and (c) participant selection.

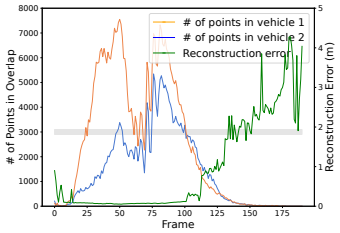


FIGURE 9: Impact of overlap on reconstruction error, ρ=3000.

	Initial Guess	Cropped	Overlapped	Expanded
≤ 80 cm	5	2	7	41
≤ 50 cm	1	2	4	39
≤ 30 cm	1	2	2	37

TABLE 10: Frames with reconstruction error less than a given limit.

Avg. Distance (m)	4.64	8.55	11.74	16.39	18.19	29.37
PointDSC	0.12	0.17	0.26	0.95	1.27	2.08
RECAP	0.02	0.01	0.02	0.04	0.01	0.06

TABLE 11: Avg. error (m) of PointDSC and RECAP at different offsets.

4.7 Design Alternatives

Using Deep Learning Based Registration. Instead of ICP, we could have used a deep neural network for spatial and temporal registration. The state-of-the-art, PointDSC [7], runs a conventional feature extraction algorithm (FPFG [56] or FCGF [16]), then matches features and extracts the transformation matrix using a neural network. To evaluate PointDSC, we used a model trained with KITTI data [32].

Spatial Registration. Tbl. 10 shows the results of an experiment in which we attempted to register 150 pairs of frames using PointDSC, at several stages in the RECAP pipeline (with the initial guess only, after cropping, after overlap extraction, and after expansion). PointDSC is able to register 25% of the frames or less within 30 cm. In contrast, RECAP is able to register ***all*** 150 frames to within a few centimeters. We conjecture (but cannot verify since PointDSC lacks explainability) that this is because PointDSC is trained with frames obtained from a single moving vehicle, so cannot generalize to settings where frames are from different vehicles.

Temporal Registration. Tbl. 11 shows PointDSC's reconstruction error for temporal registration used during expansion. In this experiment, we attempted to register point clouds from a single vehicle at different distances from each other. When point clouds are within 10 m of each other, PointDSC is reasonably accurate (but RECAP's overlap-scoped registration is much more accurate). Beyond 10 m, the performance of PointDSC degrades significantly for a reason we cannot determine because the model lacks explainability. We have left it to future work to design more generalizable deep learning approaches for registration.

μ	2000	3500	5000	6500	8000
Reconstruction Error (m)	0.15	0.13	0.11	0.09	0.08
# of Participants	7	6	6	4	4

TABLE 12: Reconstruction error and the number of participants varying the threshold μ. We use 5000 for μ.

SLAM for Temporal Expansion. We could have used SLAM to align temporal point clouds (§3.2). We compared our temporal expansion with FAST-LIO2 [77], the state-of-the-art in LiDAR-inertial odometry. We obtained fused point clouds from both our temporal expansion and FAST-LIO2, and measured the reconstruction error of the temporally fused point clouds. We averaged the error over 7 sequences each of 100 temporal frames. FAST-LIO2 incurs an average error of 6 cm while our temporal expansion incurs an error of 4 cm. SLAM trades-off some accuracy for speed (§3.2); FAST-LIO2 requires 64 ms to process a frame on average, while RECAP takes 2.5 seconds.

4.8 Sensitivity

Threshold for Expansion. The threshold ρ in §3.2 decides when to invoke temporal expansion. After about 3000 points (about roughly 12% of raw point cloud size), reconstruction error starts to rise (Fig. 9) significantly, so we use this as the value of ρ in our experiments. This result is for one trace; other traces show similar behavior, so we omit these for space.

Threshold for Participant Selection. The threshold μ determines number of actual correspondences used in participant selection. Tbl. 12 shows that when we set a higher threshold for participant selection, RECAP selects fewer participants leading to lower reconstruction error. We use μ = 5000, a sweet-spot between accuracy and the number of participants.

HD Map Positioning for Initial Guesses. Instead of smoothed GPS estimates, RECAP can also use position estimates generated using HD maps (§3.1). This approach results in comparable reconstruction accuracy to using GPS (Tbl. 13). Thus, RECAP satisfies our fourth requirement in §2.2 — it does not rely on HD maps for accuracy.

# of Vehicles	3	7	13
HD Map Positioning	0.09	0.15	0.18
GPS+KF Positioning	0.07	0.12	0.15

TABLE 13: Reconstruction error using HD maps as an initial guess.

Point Cloud Compression. RECAP compresses point clouds using octree compression [58]. Two parameters, point resolution (coding precision of points) and octree resolution (minimum voxel size), impact its compression efficacy. RECAP uses 0.0001 m for point resolution and 0.01 m for octree resolution, and achieves average bandwidth of 20.33 Mbps per vehicle and maximum 34 Mbps per vehicle. This is well within LTE supported transfer speeds; *e.g.,* average 34 Mbps in North America [18]. Except for the highest compression ratio, reconstruction error is relatively unaffected by changes to these parameters (results omitted for brevity).

5 Related Work

Traffic Reconstruction. Traffic reconstruction or traffic estimation using various sensors has a long history [76]. Traditionally, traffic flow estimation has relied on roadway sensors (*e.g.,* loop detectors) [20, 44, 49, 71]. Because static sensors can be expensive [35], more recent methods have explored using mobile devices either exclusively [17, 81], or in conjunction with static sensors [35, 50]. Other work has explored using simulators [42, 65] for 3D traffic visualization using sensor data [63, 64, 76]. In contrast, RECAP reconstructs the traffic scene in 3D from vehicular on-board sensors, providing direct information on traffic dynamics.

Collaborative Perception in Autonomous Vehicles. Using V2X connectivity, autonomous vehicles with on-board sensors can share their sensor data with each other to improve traffic safety [4, 52, 53, 83]. Unlike these approaches, RECAP seeks to reconstruct driving scenes post-facto.

Point Cloud Registration. ICP pioneered point cloud registration [10, 72]. However, accurate ICP requires a good initial guess and sufficient overlap between point clouds [54]. To address these, prior work has proposed RANSAC-based algorithms [5, 48, 51, 72, 84]. However, these are known to be computationally heavy and tend to be less accurate compared to ICP. RECAP ensures faster, more accurate, registration using overlap-scoped registration. [45] scopes RANSAC to features in the overlapped region, but this is too slow (4-6 seconds per frame) for RECAP. Most prior work on ICP focuses on accuracy [60], registers point clouds from a single vehicle/robot [60, 82], and uses ICP to reconstruct static parts of the environment. Instead, RECAP uses ICP as a building block for 3D traffic reconstruction from multiple vehicles. To our knowledge, RECAP is the first end-to-end system towards this. Recent work has explored deep-learning based registration [8, 15, 29, 33, 36, 78], but these do not generalize well to our settings (§4.6).

Recent work, VI-Eye [34] and VIPS [66], devises novel *pairwise* registration between point clouds from a vehicle and a road-side LiDAR; in contrast, RECAP performs *multi-way* registration between point clouds from multiple vehicles. VI-Eye and VIPS extract and match features instead of raw 3D points which trades off accuracy for latency. For this reason, they require a large overlap without which registration would fail. RECAP can register point clouds even if there is little or no overlap. Like RECAP, VI-Eye and VIPS use registration accuracy and computation overhead as metrics. Unlike RECAP, they do not consider coverage, an essential metric for our target application, reconstruction. More generally, because RECAP is designed for offline reconstruction, it can operate in a different point in the trade-off space: it optimizes for accuracy by registering raw point clouds instead at the cost of higher latency.

6 Limitations and Future Work

Time Synchronization. RECAP uses Network Time Protocol (NTP) which can still result in clock synchronization error of a few milliseconds, and these may contribute to the errors we observe in RECAP. For a vehicle travelling at 40 mph, a 10 ms clock offset translates to about 18 cm displacement error. That said, these synchronization errors are one reason we have a localization target of 10-20 cm. Future work can explore more accurate time synchronization methods.

Memory and Storage Overhead. Memory and storage have not been an issue running RECAP on modest desktop resources in our experimentation settings §4.1. However, a practical service based on RECAP will need to address memory usage, as well as storage and retention policies. These will depend on service pricing, operations costs etc., so are beyond the scope of our paper.

Near Real-time Reconstruction. RECAP is optimized for post-facto traffic accident reconstruction, so it improves accuracy and coverage while ensuring reconstruction within minutes. Future work can consider adapting RECAP to permit near real-time reconstruction — which may be useful for live traffic analytics — by exploiting accelerators (our current implementation does not exploit GPUs), or by trading off some accuracy and coverage for speed.

7 Conclusions

RECAP opportunistically reconstructs 3D traffic scenes, achieves high reconstruction accuracy, and generates a volumetric video from a sequence of stitched LiDAR point clouds in minutes. Beyond future work discussed in §6, other future work can explore better ways to filter small objects in the scene to improve registration accuracy, exploit infrastructure sensors, or use photogrammetry to generate a static map to obtain initial guesses for registration.

References

[1] 3D Accident Simulations in Traffic Accident Reconstruction. https://www.atlantaeng.com/3daccidentsimulations.html.

[2] Bruce Abernethy, Scott Andrews, and Gary Pruitt. Signal phase and timing (SPaT) applications, communications requirements, communications technology potential solutions, issues and recommendations. Technical Report FHWA-JPO-13-002, US Department of Transportation, 2012.

[3] The Accreditation Commission for Traffic Accident Reconstructionists. https://actar.org.

[4] Fawad Ahmad, Hang Qiu, Ray Eells, Fan Bai, and Ramesh Govindan. CarMap: Fast 3d feature map updates for automobiles. In *17th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, pages 1063–1081, 2020.

[5] Dror Aiger, Niloy J. Mitra, and Daniel Cohen-Or. 4-points congruent sets for robust pairwise surface registration. In *ACM SIGGRAPH*, 2008.

[6] E Aycock. Accident Reconstruction Fundamentals. *A Guide for Understanding Vehicle Collisions*, 2015.

[7] Xuyang Bai, Zixin Luo, Lei Zhou, Hongkai Chen, Lei Li, Zeyu Hu, Hongbo Fu, and Chiew-Lan Tai. Pointdsc: Robust point cloud registration using deep spatial consistency. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15859–15869, 2021.

[8] Dominik Bauer, Timothy Patten, and Markus Vincze. Reagent: Point cloud registration using imitation and reinforcement learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14586–14594, 2021.

[9] Ben Bellekens, Vincent Spruyt, Rafael Berkvens, Rudi Penne, and Maarten Weyn. A benchmark survey of rigid 3d point cloud registration algorithms. *Int. J. Adv. Intell. Syst*, 8:118–127, 2015.

[10] Paul J Besl and Neil D McKay. Method for registration of 3d shapes. In *Sensor fusion IV: control paradigms and data structures*, volume 1611, pages 586–606. International Society for Optics and Photonics, 1992.

[11] Eli Brosh, Matan Friedmann, Ilan Kadar, Lev Yitzhak Lavy, Elad Levi, Shmuel Rippa, Yair Lempert, Bruno Fernandez-Ruiz, Roei Herzig, and Trevor Darrell. Accurate visual localization for automotive applications. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.

[12] Yang Chen and Gérard Medioni. Object modelling by registration of multiple range images. *Image and vision computing*, 10(3):145–155, 1992.

[13] Chungjae Choe, Sangmin Ahn, Nakju Doh, and Changjoo Nam. Reduction of LiDAR Point Cloud Maps for Localization of Resource-Constrained Robotic Systems. *IEEE Systems Journal*, 2022.

[14] Sungjoon Choi, Qian-Yi Zhou, and Vladlen Koltun. Robust reconstruction of indoor scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5556–5565, 2015.

[15] Christopher Choy, Wei Dong, and Vladlen Koltun. Deep Global Registration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2514–2523, 2020.

[16] Christopher Choy, Jaesik Park, and Vladlen Koltun. Fully Convolutional Geometric Features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[17] Lianyu Chu, S Oh, and Will Recker. Adaptive kalman filter based freeway travel time estimation. In *84th TRB Annual Meeting, Washington DC*. Citeseer, 2005.

[18] Cisco Global Cloud Index: Forecast and Methodology. https://www.cisco.com/c/en/us/solutions/executive-perspectives/annual-internet-report/index.html.

[19] Courtroom animation. https://courtroomanimation.com.

[20] Carlos F Daganzo. The cell transmission model: A dynamic representation of highway traffic consistent with the hydrodynamic theory. *Transportation Research Part B: Methodological*, 28(4):269–287, 1994.

[21] T. Dahlstrom. The Road to Everywhere: Are HD maps for Autonomous Driving Sustainable? https://www.autonomousvehicleinternational.com/features/the-road-to-everywhere-are-hd-maps-for-autonomous-driving-sustainable.html.

[22] John Daily, Nathan S Shigemura, and Jeremy Daily. *Fundamentals of traffic crash reconstruction*, volume 2. Institute of Police Technology & Management, 2006.

[23] Dallas Police Department: Response Time Summary Report. https://www.dallaspolice.net/resources/CrimeReports/New%20Response%20Time%20Report.pdf, June 2022.

[24] Patrick Daniel. What is an Accident Reconstructionist? https://www.patrickdaniellaw.com/blog/accident-reconstructionist/.

[25] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.

[26] Statistics on Intersection Accidents. Federal Highway Administration, 2021.

[27] US Department of Transportation Federal Highway Administration - Intersection Safety, Jan 2022.

[28] Lynn B Fricke. *Traffic accident reconstruction*. Northwestern Univ Center for public, 1990.

[29] Kexue Fu, Shaolei Liu, Xiaoyuan Luo, and Manning Wang. Robust point cloud registration framework based on deep graph matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8893–8902, 2021.

[30] Yasutaka Furukawa and Carlos Hernández. Multi-view stereo: A tutorial. *Found. Trends Comput. Graph. Vis.*, 9(1-2):1–148, 2015.

[31] Hampton Gabler, Carolyn Hampton, and Nicolas Johnson. Development of the WinSMASH 2010 Crash Reconstruction Code. Technical Report 811546, National Highway Traffic Safety Administration, 2012.

[32] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets Robotics: The KITTI Dataset. *International Journal of Robotics Research (IJRR)*, 2013.

[33] Zan Gojcic, Caifa Zhou, Jan D Wegner, Leonidas J Guibas, and Tolga Birdal. Learning multiview 3d point cloud registration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pages 1759–1769, 2020.

[34] Yuze He, Li Ma, Zhehao Jiang, Yi Tang, and Guoliang Xing. 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 (MobiCom)*, page 573–586, 2021.

[35] Juan C Herrera and Alexandre M Bayen. Traffic flow reconstruction using mobile sensors and loop detector data. *UC Berkeley: University of California Transportation Center*, 2007.

[36] Shengyu Huang, Zan Gojcic, Mikhail Usvyatsov, Andreas Wieser, and Konrad Schindler. Predator: Registration of 3d point clouds with low overlap. In *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition (CVPR)*, pages 4267–4276, 2021.

[37] Primer on Intersection Control Evaluation (ICE). https://safety.fhwa.dot.gov/intersection/ice/fhwasa18076.pdf.

[38] John Jackson, Ricardo Saborio, Syed Anas Ghazanfar, Demoz Gebre-Egziabher, and Brian Davis. Evaluation of Low-Cost, Centimeter-Level Accuracy OEM GNSS Receivers. Technical report, University of Minnesota, 2018.

[39] Nicholas S. Johnson and Hampton C. Gabler. Accuracy of a Damage-Based Reconstruction Method in NHTSA Side Crash Tests. *Traffic*

Injury Prevention, 13(1):72–80, 2012.

[40] Andrej Karpathy. CVPR'21 Workshop on Autonomous Driving: Andrej Karpathy Keynote.

[41] L. A. Klein, M. K. Mills, and D. R. P. Gibson. *Traffic Detector Handbook*. US Federal Highway Safety Administration, 3rd edition, 2006.

[42] Robert S Laramee, Helwig Hauser, Helmut Doleisch, Benjamin Vrolijk, Frits H Post, and Daniel Weiskopf. The state of the art in flow visualization: Dense and texture-based techniques. *Computer Graphics Forum*, 23(2):203–221, 2004.

[43] Liang Li, Ming Yang, Hao Li, Chunxiang Wang, and Bing Wang. Robust Localization for Intelligent Vehicles Based on Compressed Road Scene Map in Urban Environments. *IEEE Transactions on Intelligent Vehicles*, 2022.

[44] Wei-Hua Lin and Dike Ahanotu. Validating the basic cell transmission model on a single freeway link. *PATH technical note; 95-3*, 1995.

[45] Wenbo Liu, Wei Sun, Shuxuan Wang, and Yi Liu. Coarse registration of point clouds with low overlap rate on feature regions. *Signal Processing: Image Communication*, 98:116428, 2021.

[46] Martin Magnusson. *The three-dimensional normal-distributions transform: an efficient representation for registration, surface analysis, and loop detection*. PhD thesis, Örebro universitet, 2009.

[47] Brian McHenry. Introduction: Crash Reconstruction Techniques and Accuracy. https://mchenrysoftware.com/wp/publicationspresentations/accident-reconstruction-techniques-and-accuracy/.

[48] Nicolas Mellado, Dror Aiger, and Niloy J Mitra. Super 4pcs fast global pointcloud registration via smart indexing. *Computer graphics forum*, 33(5):205–215, 2014.

[49] Laura Muñoz, Xiaotian Sun, Roberto Horowitz, and Luis Alvarez. Traffic density estimation with the cell transmission model. In *Proceedings of the 2003 American Control Conference*, volume 5, pages 3750–3755, 2003.

[50] Chumchoke Nanthawichit, Takashi Nakatsuji, and Hironori Suzuki. Application of probe-vehicle data for real-time traffic-state estimation and short-term travel-time prediction on a freeway. *Transportation research record*, 1855(1):49–59, 2003.

[51] Chavdar Papazov, Sami Haddadin, Sven Parusel, Kai Krieger, and Darius Burschka. Rigid 3D geometry matching for grasping of known objects in cluttered scenes. *The International Journal of Robotics Research*, 31(4):538–553, 2012.

[52] Hang Qiu, Fawad Ahmad, Fan Bai, Marco Gruteser, and Ramesh Govindan. AVR: Augmented Vehicular Reality. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys)*, pages 81–95, 2018.

[53] Hang Qiu, Po-Han Huang, Namo Asavisanu, Xiaochen Liu, Konstantinos Psounis, and Ramesh Govindan. Autocast: Scalable infrastructure-less cooperative perception for distributed collaborative driving. In *Proceedings of the 20th Annual International Conference on Mobile Systems, Applications and Services (MobiSys)*, page 128–141, 2022.

[54] Carolina Raposo and Joao P Barreto. Using 2 point+ normal sets for fast registration of point clouds with small overlap. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5652–5658, 2017.

[55] R. B. Rusu, N. Blodow, Z. C. Marton, and M. Beetz. Aligning point cloud views using persistent feature histograms. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3384–3391, 2008.

[56] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast point feature histograms (FPFH) for 3d registration. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3212–3217, 2009.

[57] Radu Bogdan Rusu and Steve Cousins. 3D is here: Point Cloud Library (PCL). In *IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[58] Ruwen Schnabel and Reinhard Klein. Octree-based Point-Cloud Compression. In *PBG@ SIGGRAPH*, pages 111–120, 2006.

[59] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016.

[60] Aleksandr Segal, Dirk Haehnel, and Sebastian Thrun. Generalized-icp. In *Robotics: science and systems*, volume 2, page 435, 2009.

[61] Steven M Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 519–528, 2006.

[62] SensorLog. SensorLog. http://sensorlog.berndthomas.net/.

[63] Jason Sewall, Jur Van Den Berg, Ming Lin, and Dinesh Manocha. Virtualized traffic: Reconstructing traffic flows from discrete spatiotemporal data. *IEEE transactions on visualization and computer graphics*, 17(1):26–37, 2010.

[64] Jason Sewall, David Wilkie, and Ming C Lin. Interactive hybrid simulation of large-scale traffic. In *Proceedings of the 2011 SIGGRAPH Asia Conference*, pages 1–12, 2011.

[65] Jason Sewall, David Wilkie, Paul Merrell, and Ming C Lin. Continuum traffic simulation. *Computer Graphics Forum*, 29(2):439–448, 2010.

[66] Shuyao Shi, Jiahe Cui, Zhehao Jiang, Zhenyu Yan, Guoliang Xing, Jianwei Niu, and Zhenchao Ouyang. Vips: Real-time perception fusion for infrastructure-assisted autonomous driving. In *Proceedings of the 28th annual international conference on mobile computing and networking (MobiCom)*, pages 133–146, 2022.

[67] Timothy Staab. *The Pocket Traffic Accident Reconstruction Guide*. Lawyers & Judges Pub Co, 2017.

[68] Donald E Struble and John D Struble. *Automotive accident reconstruction: practices and principles*. CRC Press, 2020.

[69] Zhongxing Tao, Jianru Xue, Di Wang, Gengxin Li, and Jianwu Fang. An Adaptive Invariant EKF for Map-Aided Localization Using 3D Point Cloud. *IEEE Transactions on Intelligent Transportation Systems*, 2022.

[70] SAE Taxonomy. Definitions for terms related to on-road motor vehicle automated driving systems. *Society of Automotive Engineers (SAE): Troy, MI, USA*, 2014.

[71] Martin Treiber and Dirk Helbing. Reconstructing the spatio-temporal traffic dynamics from stationary detector data. *Cooperative Transportation Dynamics*, 1(3):3–1, 2002.

[72] Emanuele Trucco, Andrea Fusiello, and Vito Roberto. Robust motion and correspondence of noisy 3D point sets with missing data. *Pattern recognition letters*, 20(9):889–898, 1999.

[73] Chi-Yi Tsai, Humaira Nisar, and Yu-Chen Hu. Mapless LiDAR Navigation Control of Wheeled Mobile Robots Based on Deep Imitation Learning. *IEEE Access*, pages 1–1, 08 2021.

[74] Virtual Crash Accident Reconstruction Software. https://www.vcrashusa.com/.

[75] Huayou Wang, Changliang Xue, Yanxing Zhou, Feng Wen, and Hongbo Zhang. Visual semantic localization based on HD map for autonomous vehicles in urban scenarios. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 11255–11261, 2021.

[76] David Wilkie, Jason Sewall, and Ming Lin. Flow Reconstruction for Data-Driven Traffic Animation. *ACM Trans. Graph.*, 32(4), jul 2013.

[77] Wei Xu, Yixi Cai, Dongjiao He, Jiarong Lin, and Fu Zhang. Fast-lio2: Fast direct lidar-inertial odometry. *IEEE Transactions on Robotics*, 38(4):2053–2073, 2022.

[78] Heng Yang, Jingnan Shi, and Luca Carlone. Teaser: Fast and certifiable point cloud registration. *IEEE Transactions on Robotics*, 37(2):314–333, 2020.

[79] Jiaolong Yang, Hongdong Li, and Yunde Jia. Go-icp: Solving 3d registration efficiently and globally optimally. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1457–1464, 2013.

[80] Jui-An Yang and Chung-Hsien Kuo. Integrating Vehicle Positioning and Path Tracking Practices for an Autonomous Vehicle Prototype in Campus Environment. *Electronics*, 10(21):2703, 2021.

[81] Jean-Luc Ygnace, Chris Drane, YB Yim, and Renaud De Lacvivier. Travel Time Estimation on the San Francisco Bay Area Network Using Cellular Phones as Probes. *Institute of Transportation Studies, UC Berkeley, Institute of Transportation Studies, Research Reports, Working Papers, Proceedings*, 01 2000.

[82] Wenan Yuan, Daeun Choi, and Dimitrios Bolkas. Gnss-imu-assisted colored icp for uav-lidar point cloud registration of peach trees. *Computers and Electronics in Agriculture*, 197:106966, 2022.

[83] Xumiao Zhang, Anlan Zhang, Jiachen Sun, Xiao Zhu, Y Ethan Guo, Feng Qian, and Z Morley Mao. EMP: edge-assisted multi-vehicle perception. In *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking (MobiCom)*, pages 545–558, 2021.

[84] Zhengyou Zhang. On local matching of free-form curves. In *BMVC92*, pages 347–356. Springer, 1992.

[85] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Fast global registration. In *European Conference on Computer Vision (ECCV)*, pages 766–782, 2016.

[86] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018.