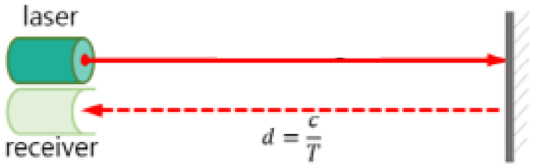
EMP: Edge-assisted Multi-vehicle Perception

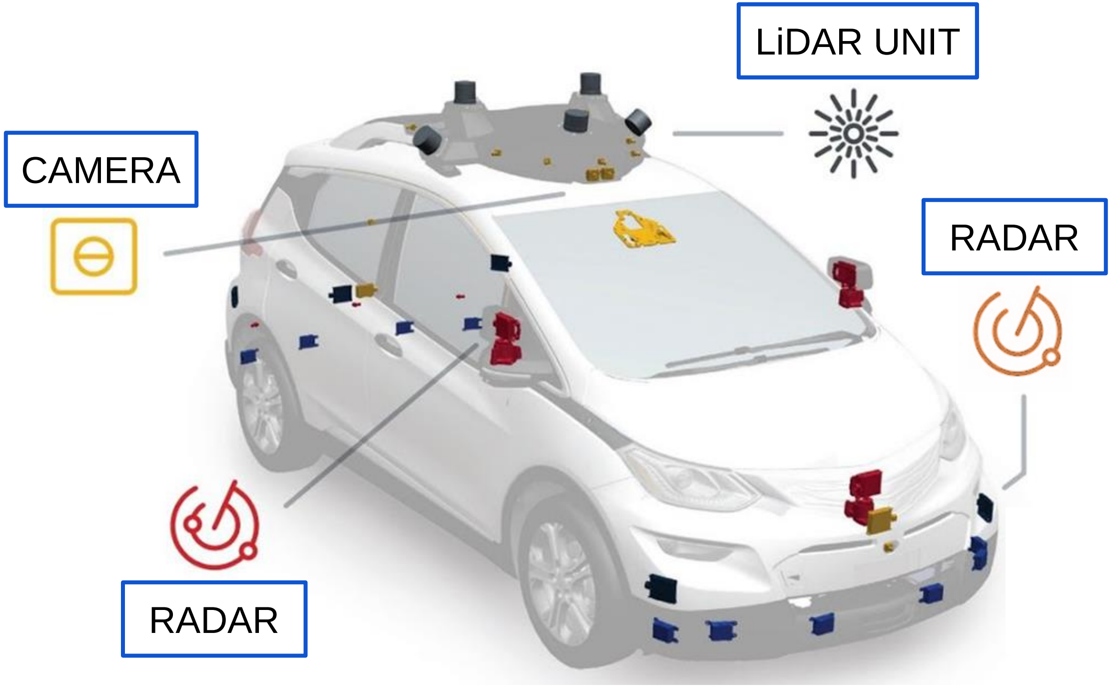
Xumiao Zhang, Anlan Zhang[†], Jiachen Sun, Xiao Zhu,

Yihua Guo[‡], Feng Qian[†], Z. Morley Mao

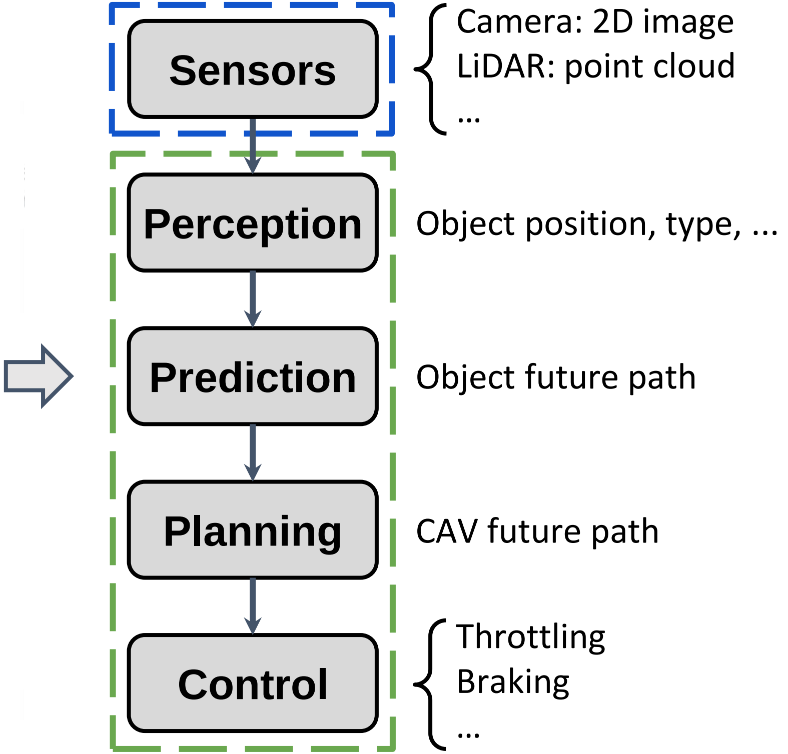
University of Michigan [†]University of Minnesota – Twin Cities [‡]Uber Technologies, Inc.

Connected and Autonomous Vehicle



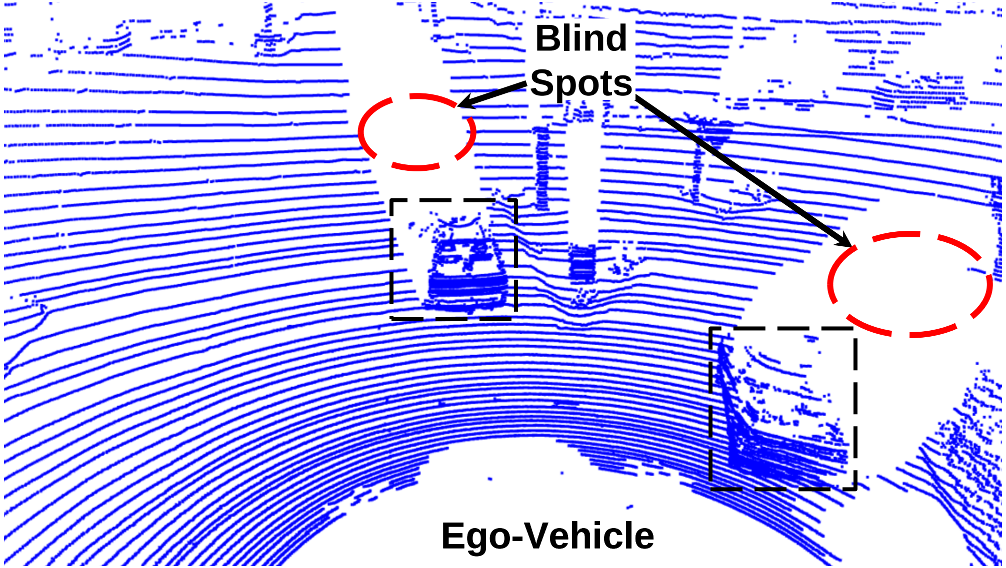


* LiDAR = Light Detection And Ranging



Limitations of On-board Sensors

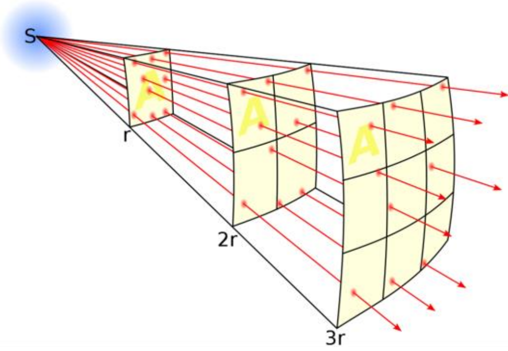
- They are vulnerable to occlusion.



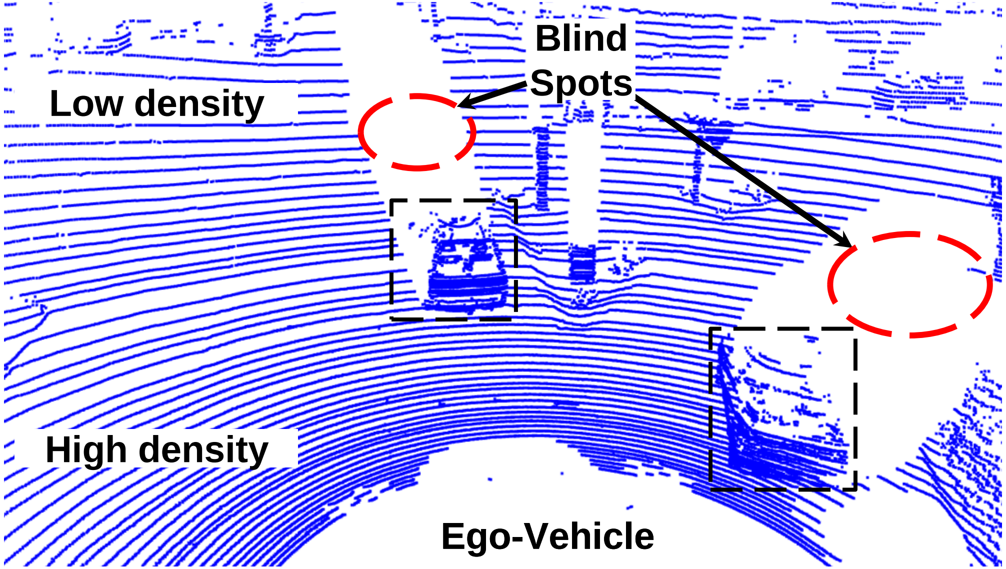
A visualized LiDAR point cloud (blue)

* Ego-vehicle: the vehicle collecting sensor data and perceiving the environment

Limitations of On-board Sensors

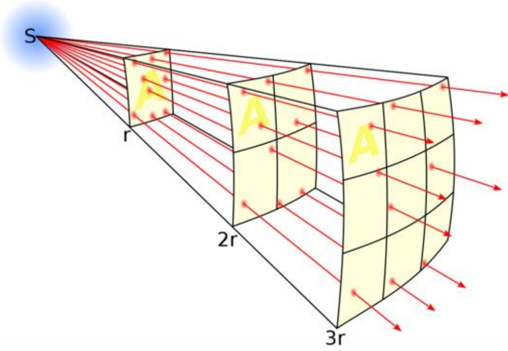


- They are vulnerable to occlusion.
- The farther an object is, the fewer details they can capture.

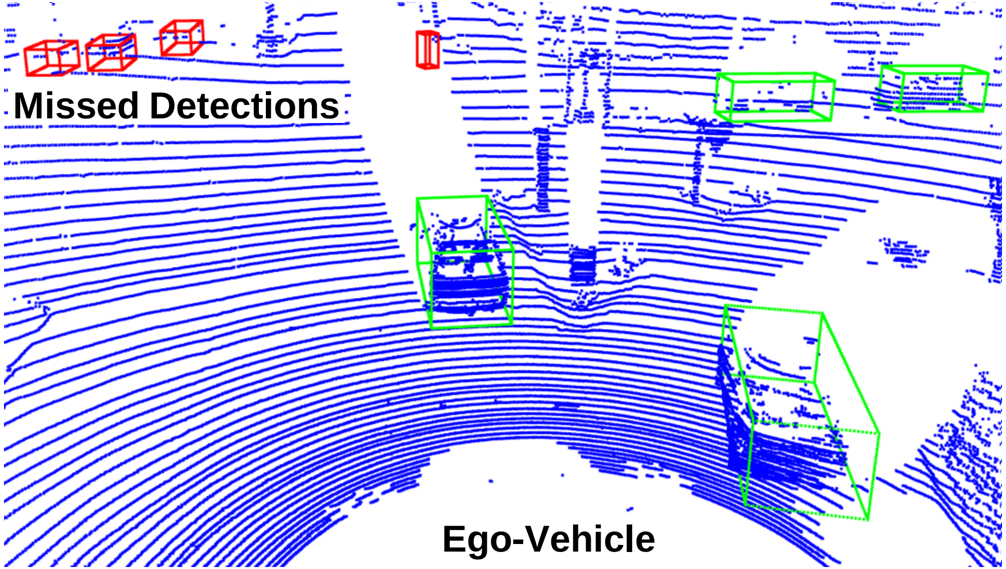


A visualized LiDAR point cloud (blue)

Limitations of On-board Sensors



- They are vulnerable to occlusion.
- The farther an object is, the fewer details they can capture.

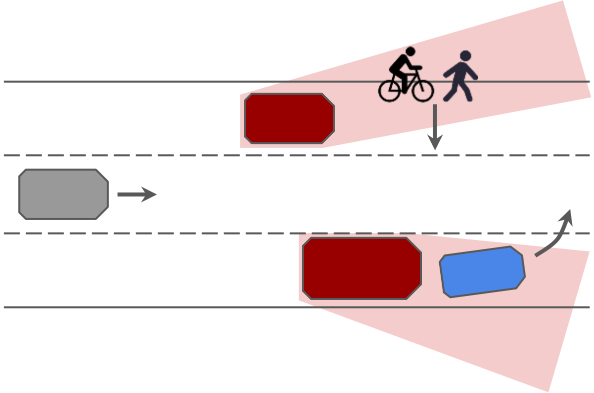


A visualized LiDAR point cloud (blue)

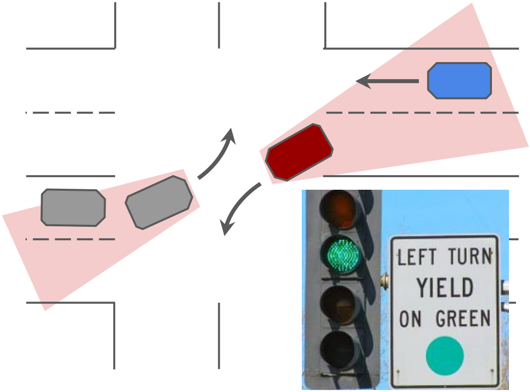
* Ego-vehicle: the vehicle collecting sensor data and perceiving the environment

Benefits of Sensor Data Sharing

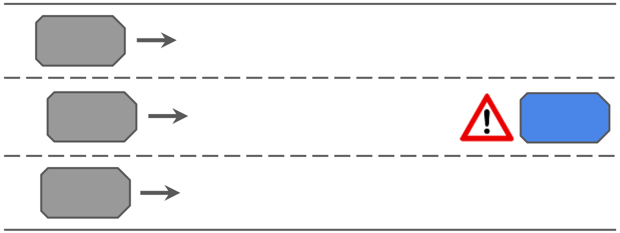
- Different vehicles perceive information from various locations
  - *objects occluded in the views of some vehicles can be easily perceived by others.*
- Driving scenarios where vehicles can benefit from sensor data sharing:



(1) Blind Spots

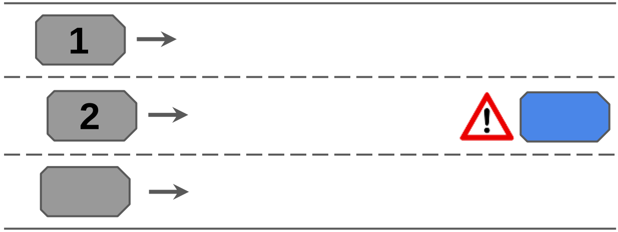


(2) Unprotected Left Turn

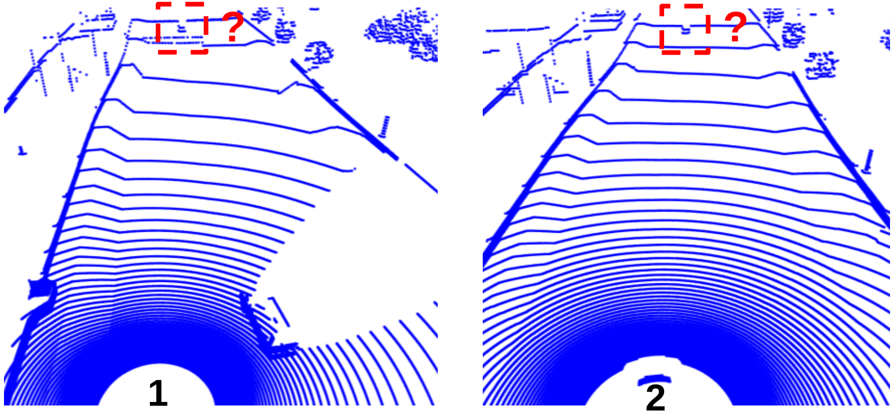


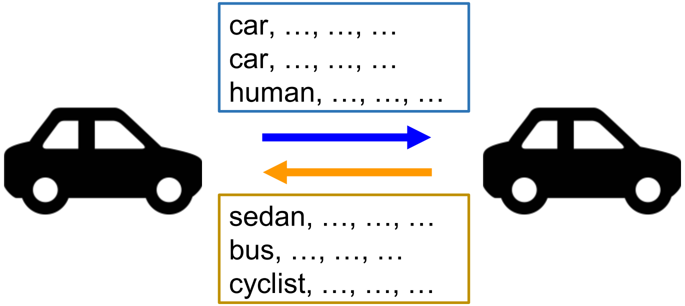
(3) Broken Down Vehicle

Limitations of Existing Solutions

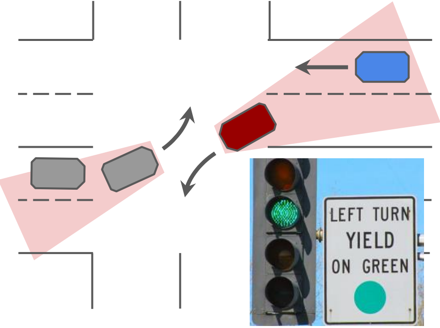


- Sharing processed data [1,2]
  - *Limited data granularity: missed detections will still be missed after sharing*
    - Combining sensor data can lead to a higher resolution
  - *Lack of generality*
    - Raw data has a fundamental and universal format, compatible with various applications



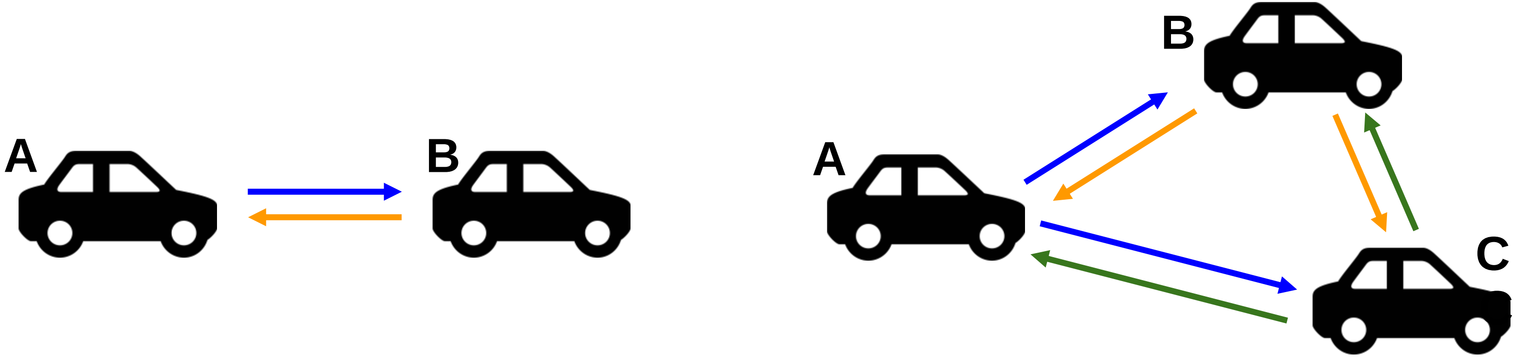


[1] Liu, Hansi, et al. "FusionEye: Perception Sharing for Connected Vehicles and its Bandwidth-Accuracy Trade-offs." IEEE SECON. 2019.
[2] Chen, Qi, et al. "F-cooper: feature based cooperative perception for autonomous vehicle edge computing system using 3D point clouds." ACM/IEEE SEC. 2019.



Limitations of Existing Solutions

- Vehicle-to-vehicle sharing [1,2,3]
  - *Additional <u>network</u> overhead for sharing with different vehicles*
    - N vehicles → N-1 copies, N*(N-1) bandwidth consumption
  - *Additional <u>computational</u> overhead for processing data from others*
    - CAV hardware is originally equipped for processing single-vehicle data



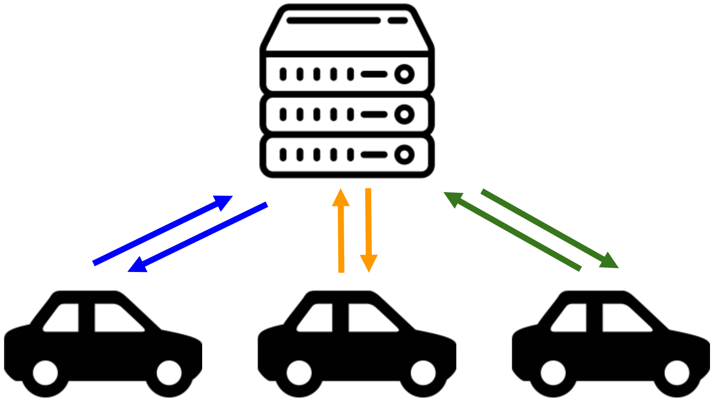
(1) Number of Vehicles = 2

(2) Number of Vehicles ≥ 3

[1] Chen, Qi, et al. "Cooper: Cooperative perception for connected autonomous vehicles based on 3d point clouds." IEEE ICDCS, 2019.
[2] Olaverri-Monreal, Cristina, et al. "The See-Through System: A VANET-enabled assistant for overtaking maneuvers." IEEE Intelligent Vehicles Symposium, 2010.
[3] Qiu, Hang, et al. "Avr: Augmented vehicular reality." Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services. 2018.

Need for an Edge-assisted System

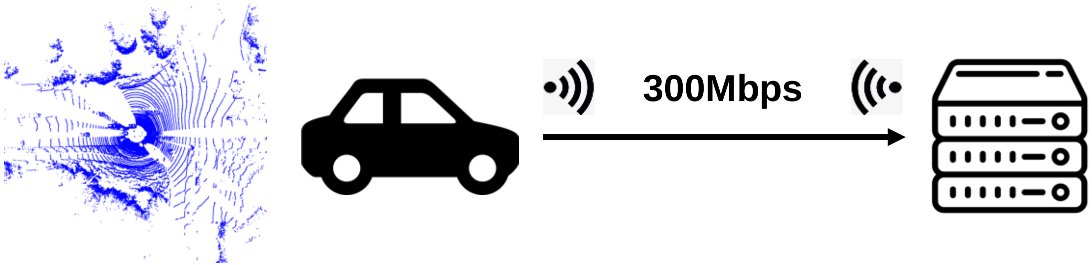
- Offloading heavy computational tasks to an **edge**
  - *Edge: computing resources close to vehicles, providing low network latency*
  - *Advantages of using an edge*
    - Less network overhead: vehicles only need to share their sensor data to the dge
    - More computational resources: compared to a vehicle’s on-board hardware

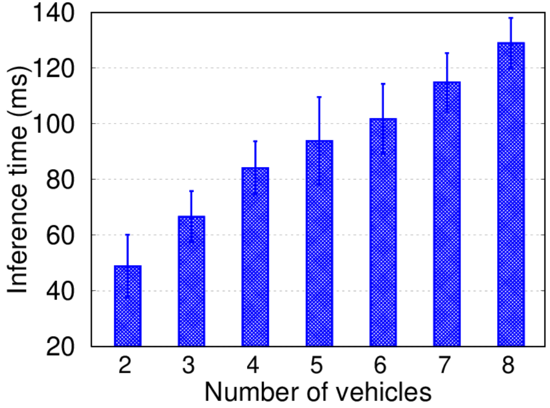


Raw point cloud: ~2.0MB
LiDAR capture rate: 5-20Hz

Challenges

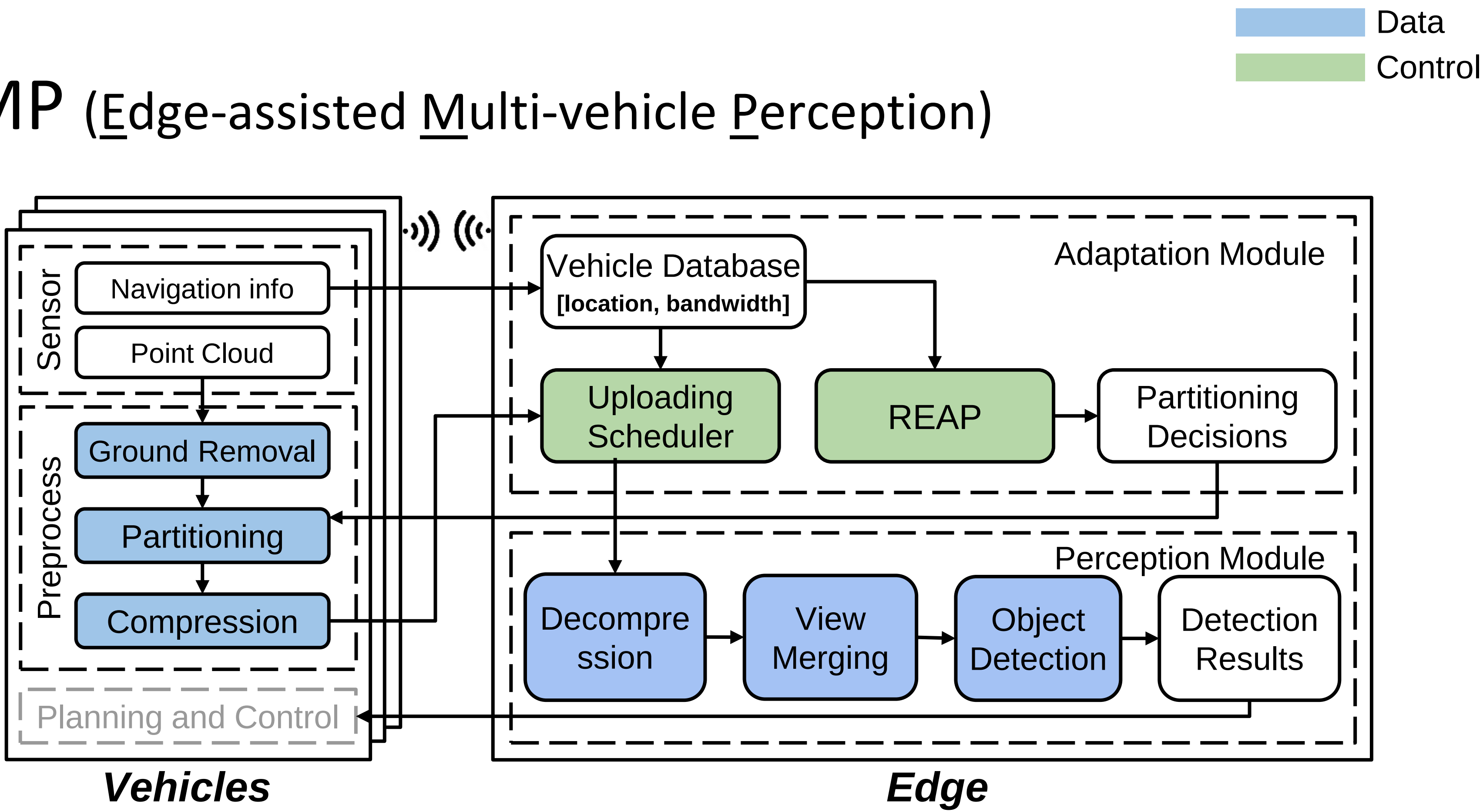
1. Bulky size of raw sensor data
2. Increased latency to process aggregated data
3. Network resource variability
   - *Vehicles have different available bandwidths*.*
   - *Wireless networks fluctuate under high mobility.*
4. Asynchronous data arrival



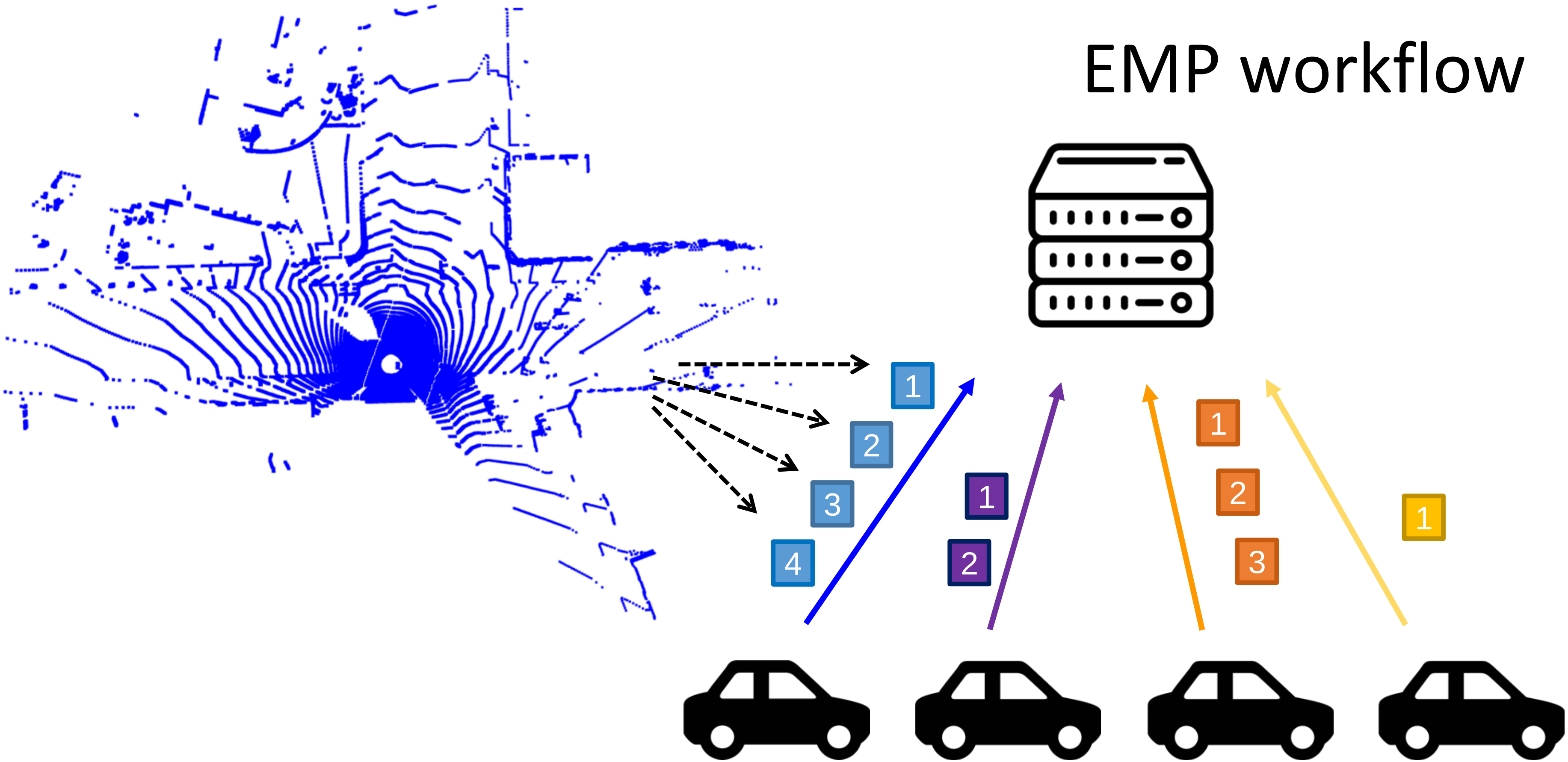


* Available bandwidth: the maximum throughput that an end host can achieve during data transfer

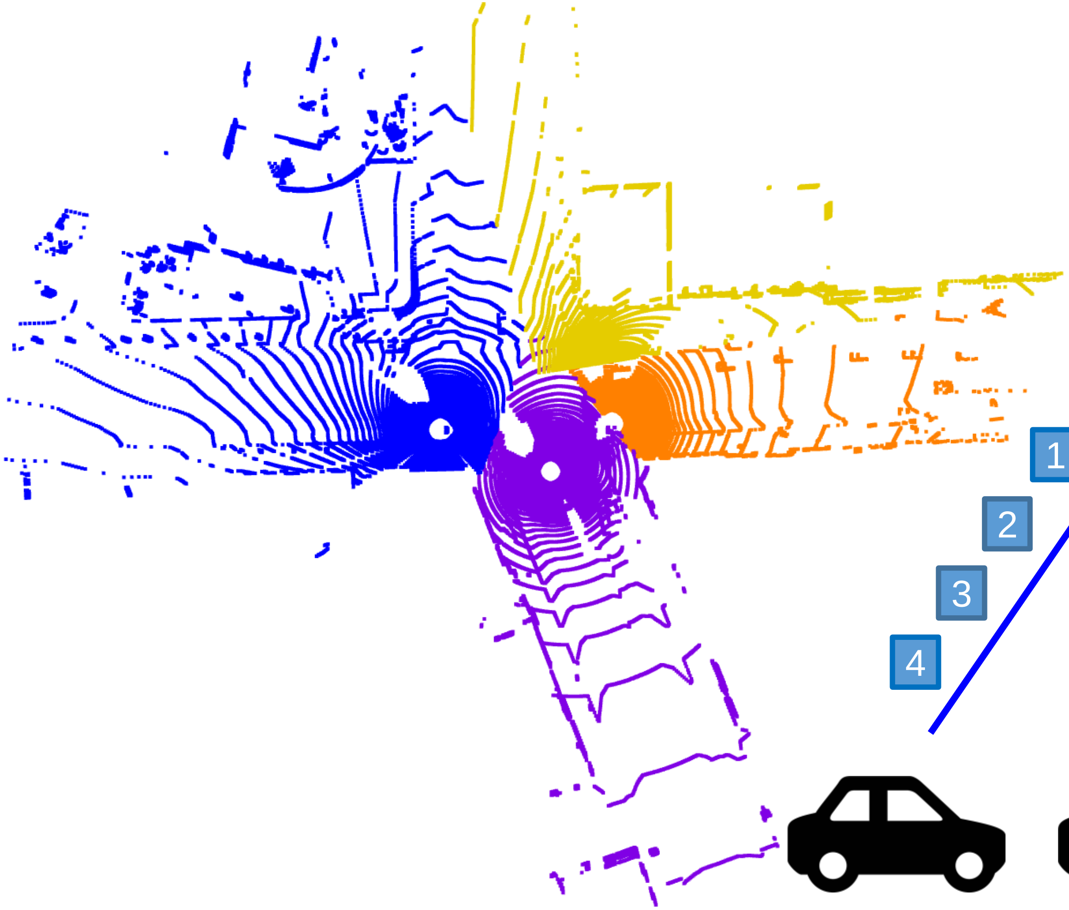
EMP (Edge-assisted Multi-vehicle Perception)

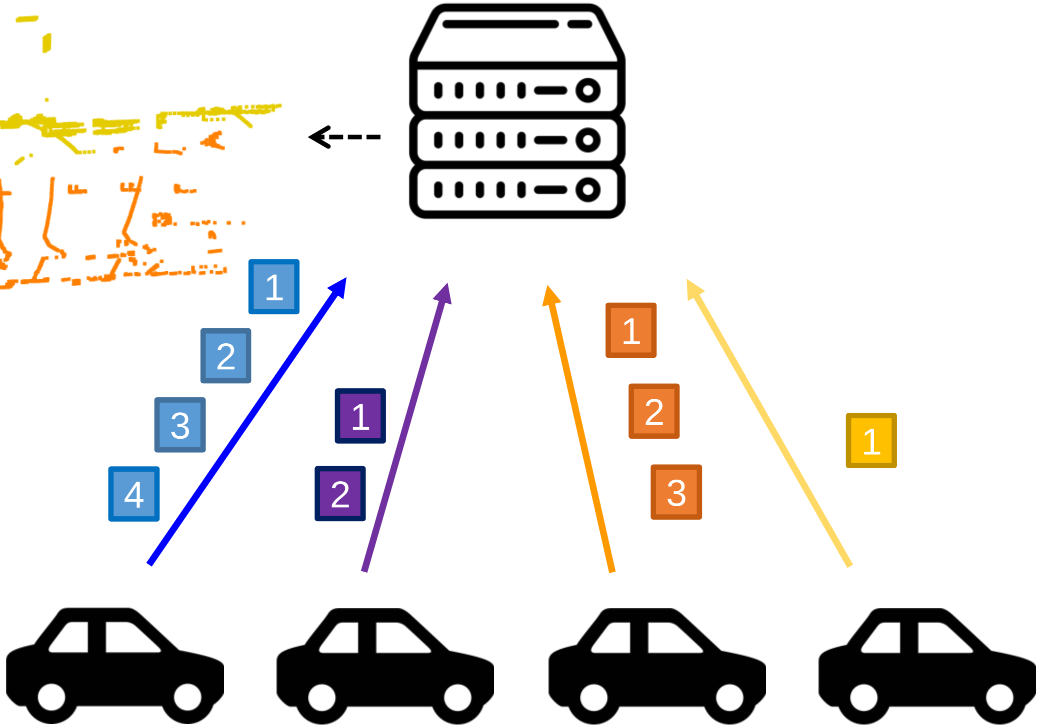


EMP workflow

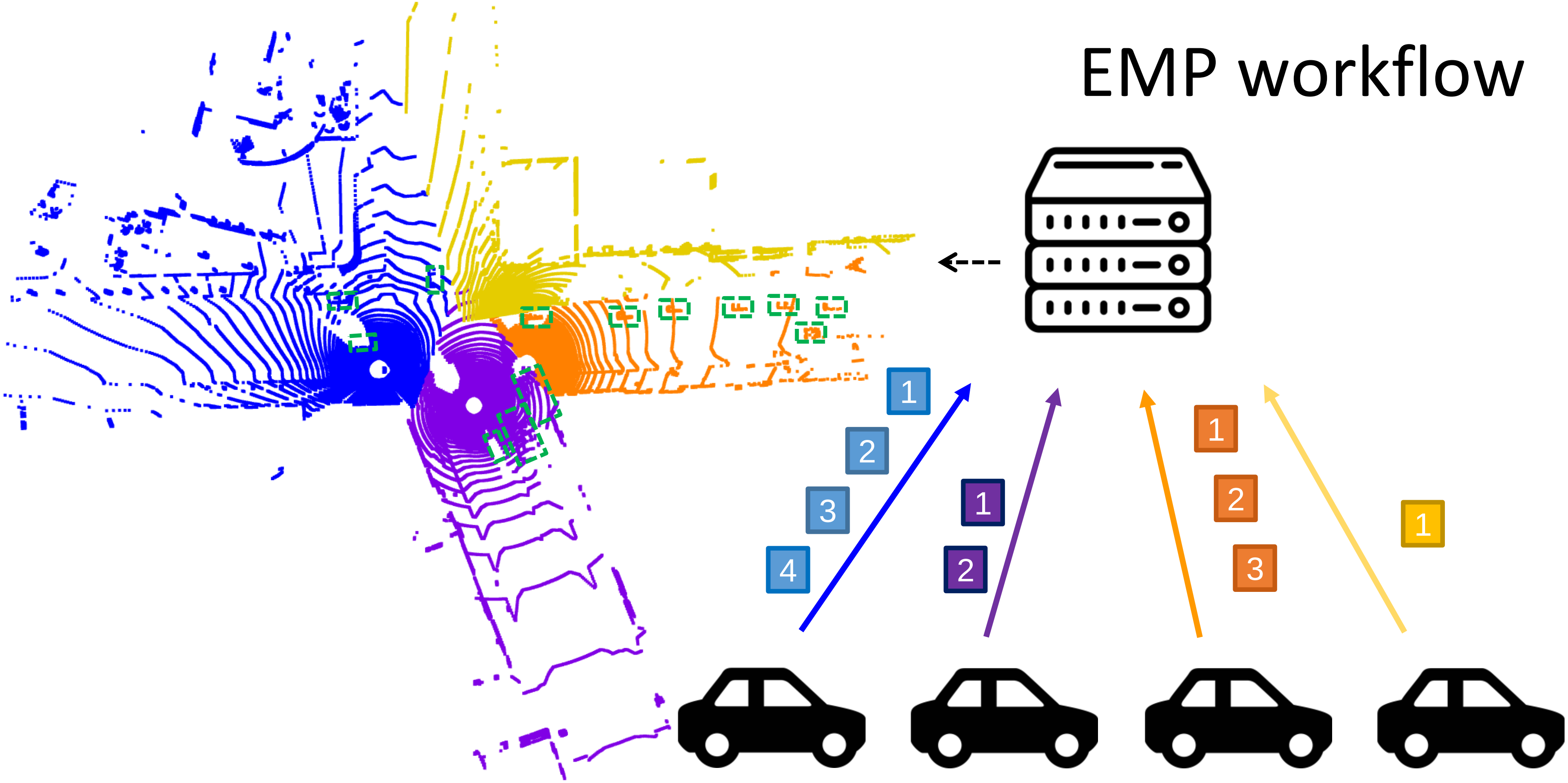


EMP workflow

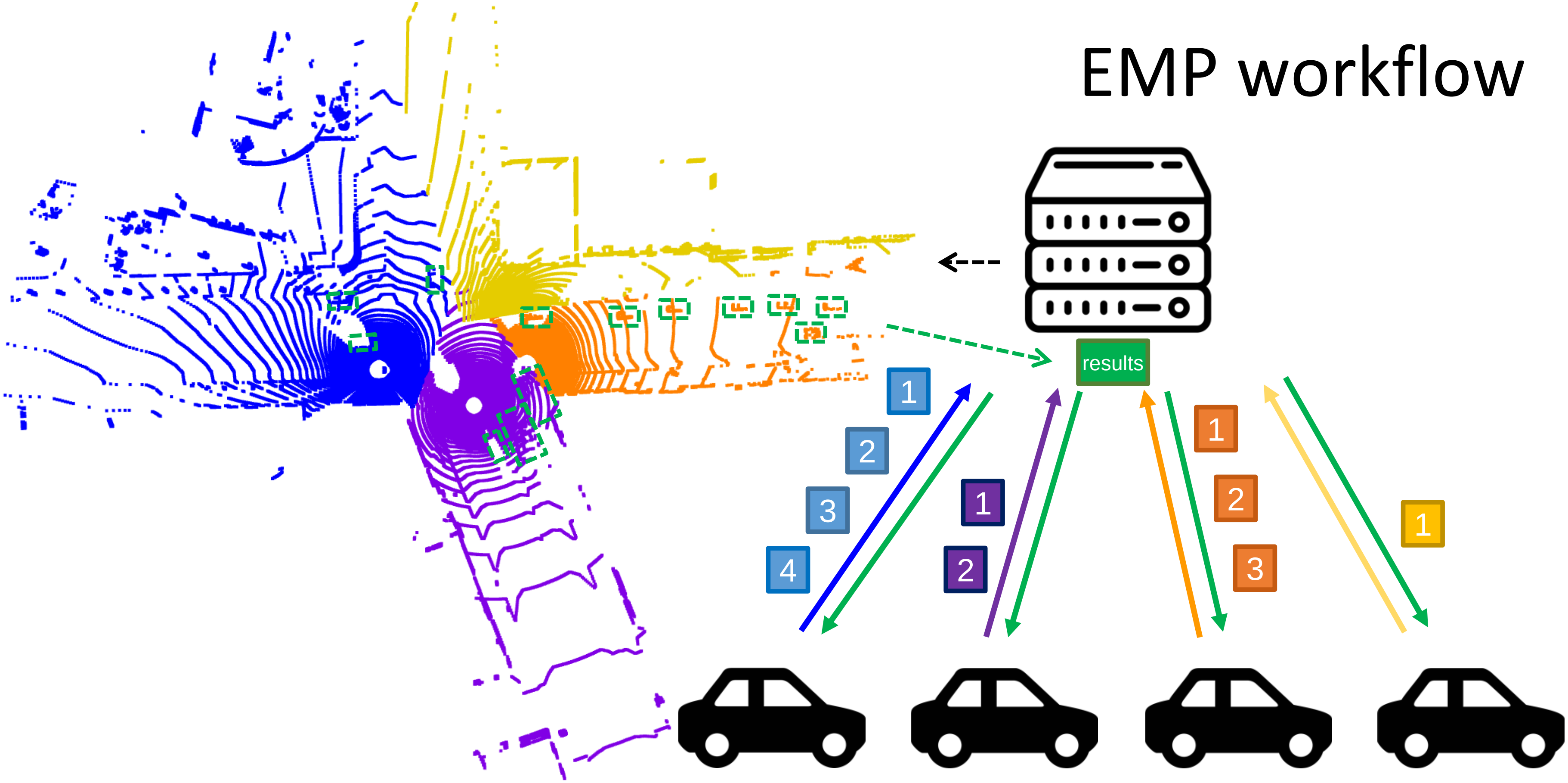


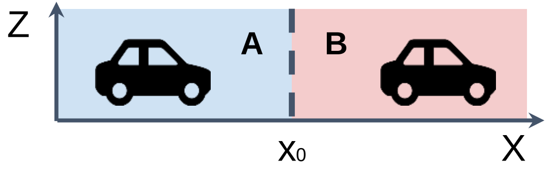


EMP workflow



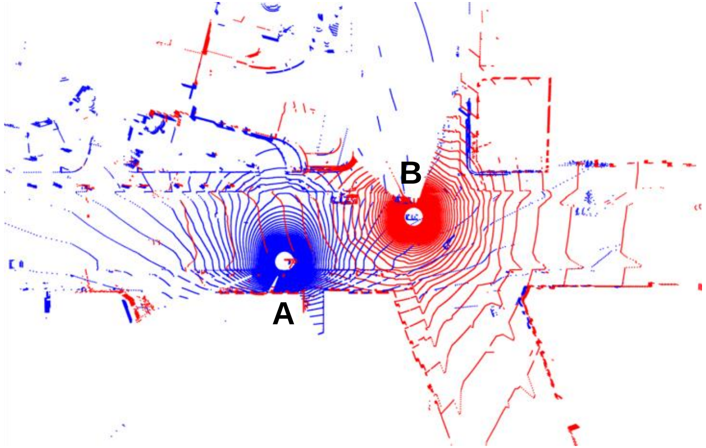
EMP workflow



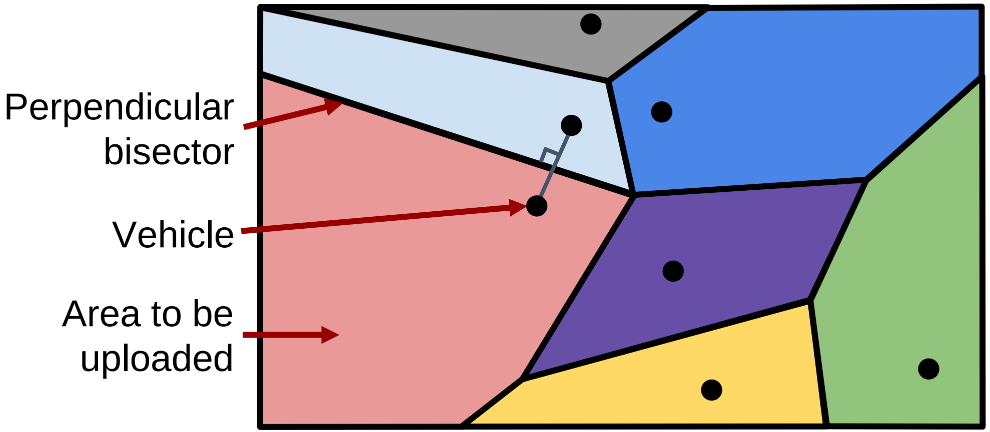


Point Cloud Partitioning

- Partitions the whole area into non-overlapping regions
  - *Key idea: assigns each point to the closest vehicle*
  - ***Voronoi diagram**: partitioned by the perpendicular bisectors of connections between every two neighboring vehicles.*



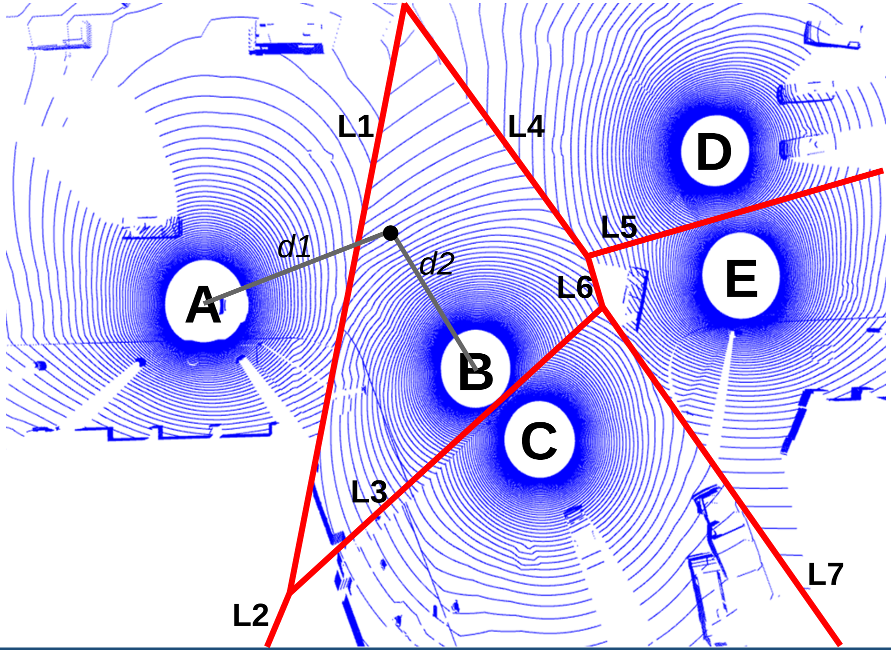
Point clouds of two vehicles (bird-eye view)



Point Cloud Partitioning

- Naive partitioning of point cloud through Voronoi diagram

d1 > d2



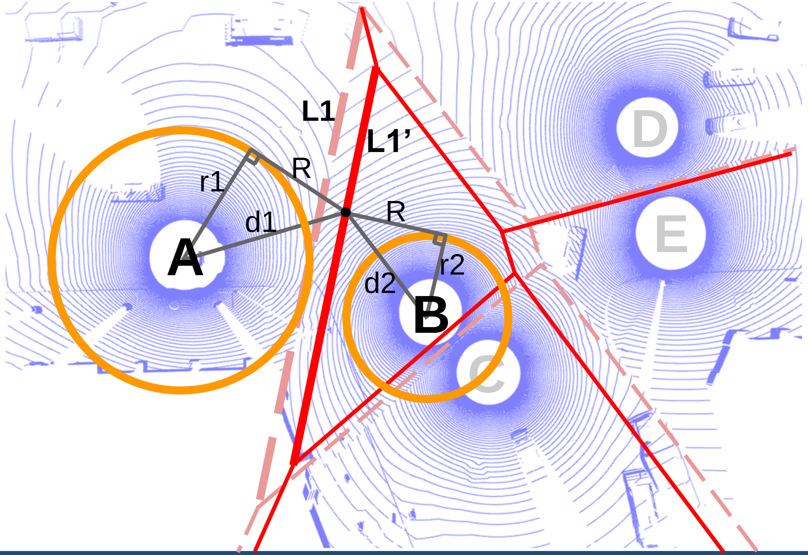
What if A's bandwidth is much lower than B's?

Bandwidth-aware Partitioning

- Partition based on the vehicle locations and the estimated bandwidths
  - *Key idea: uploaded area positively correlated to the estimated bandwidths*
  - ***Power diagram** (weighted Voronoi diagram)*

Weights: r1 ∝ BW_A, r2 ∝ BW_C

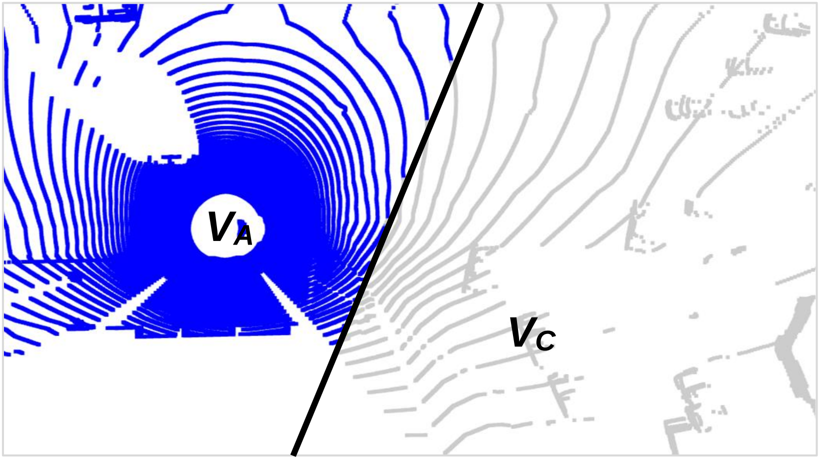
$R^2 = d1^2 - r1^2 = d2^2 - r2^2$



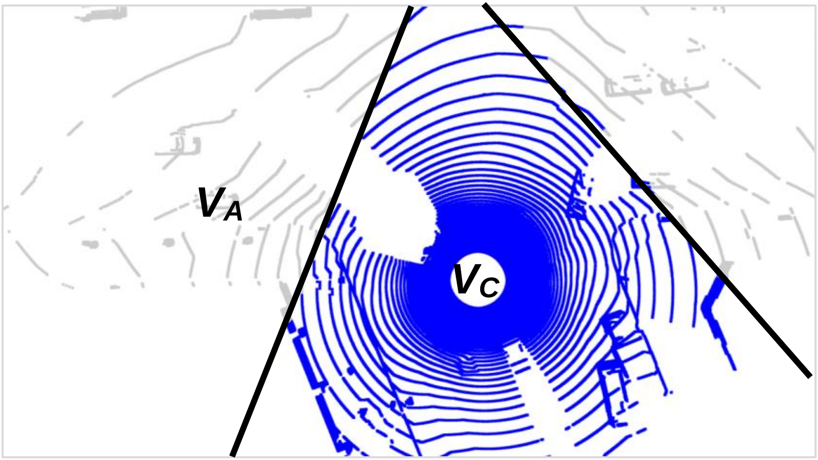
What if A's bandwidth becomes lower than B's?

Adaptation to Bandwidth Fluctuation

- Partition the data into multiple chunks with two additional boundaries
  - *Consider Accurate/Overestimated/Underestimated bandwidth*



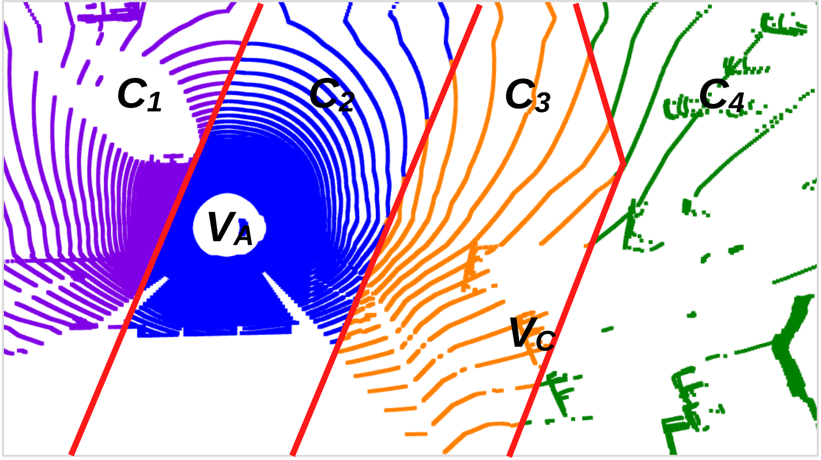
(1) Vehicle A's point cloud



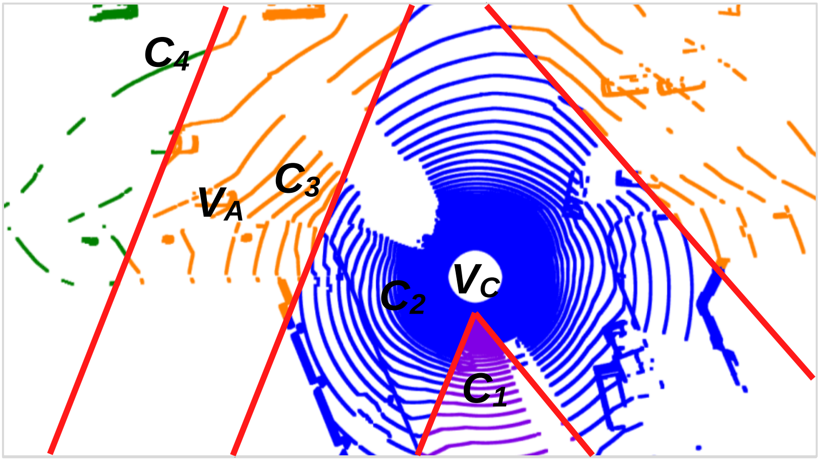
(2) Vehicle C's point cloud

Adaptation to Bandwidth Fluctuation

- Partition the data into multiple chunks with two additional boundaries
  - *Consider Accurate/Overestimated/Underestimated bandwidth*
  - *Each vehicle sequentially uploads from chunk 1 to chunk 4*



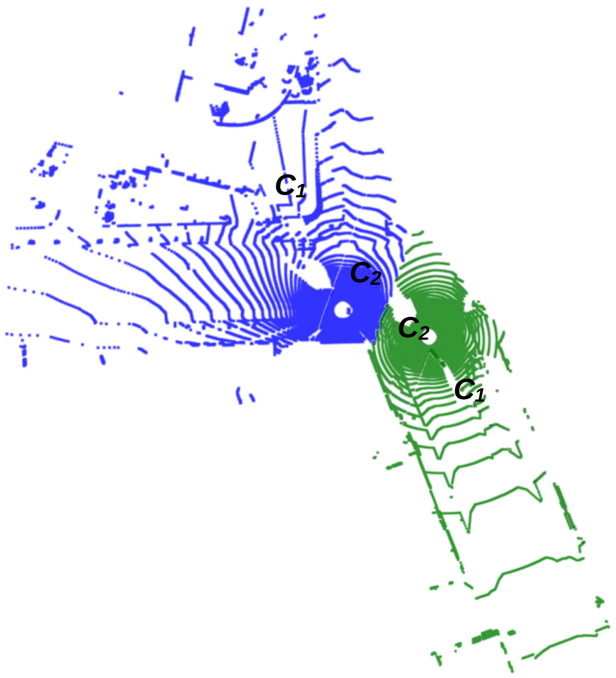
(1) Vehicle A's point cloud



(2) Vehicle C's point cloud

Upload Scheduling

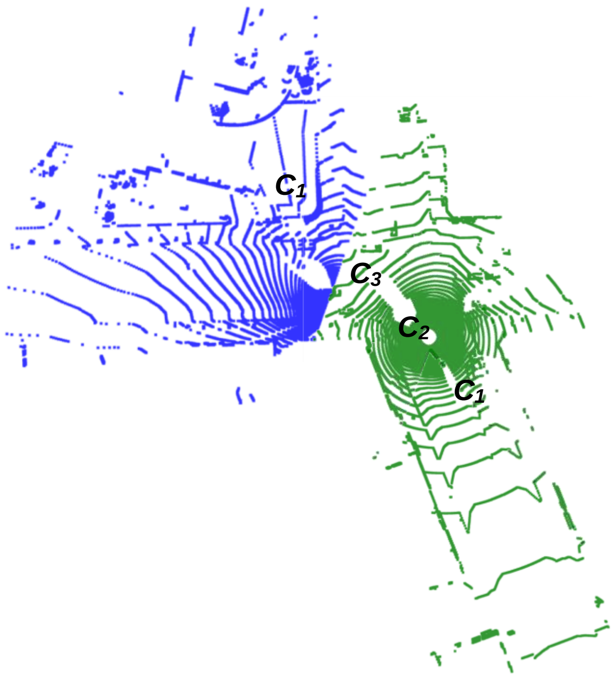
- Upload finish conditions
  - *C_1 & C_2*



* C_1: chunk 1

Upload Scheduling

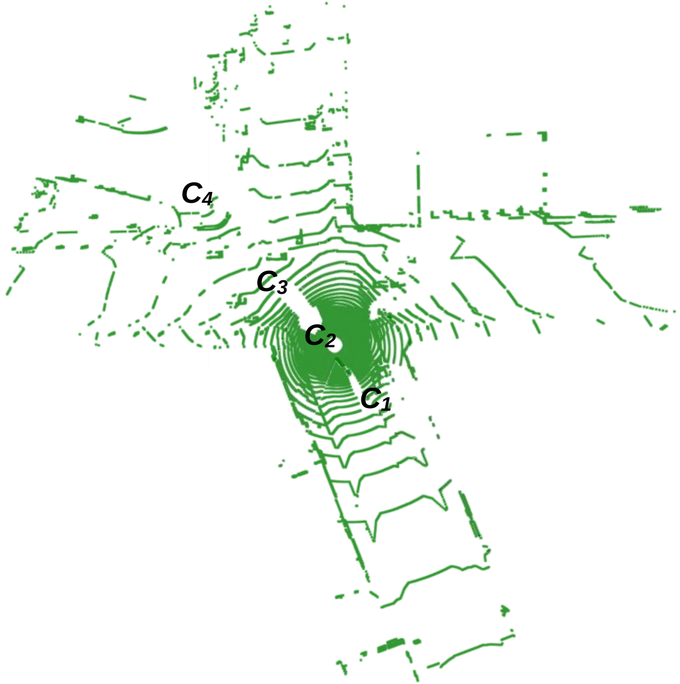
- Upload finish conditions
  - *C_1 & C_2*
  - *C_1 + neighbors' C_3*



* C_1: chunk 1

Upload Scheduling

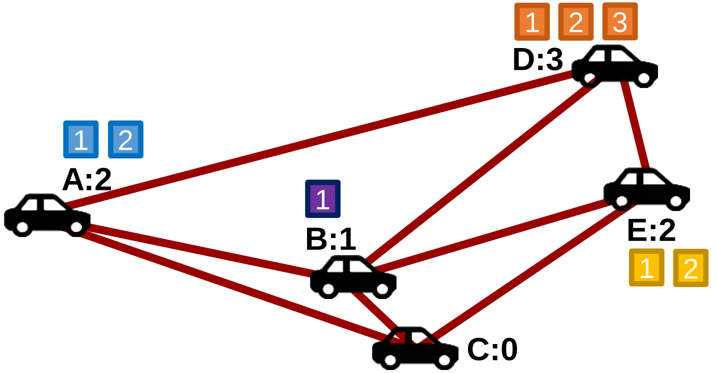
- Upload finish conditions
  - *C_1 & C_2*
  - *C_1 + neighbors' C_3*
  - *neighbors' C_3 & C_4*



* C_1: chunk 1

Upload Scheduling

- Upload finish conditions
  - *C_1 & C_2*
  - *C_1 + neighbors' C_3*
  - *neighbors' C_3 & C_4*
- Check chunk delivery status upon receiving each chunk

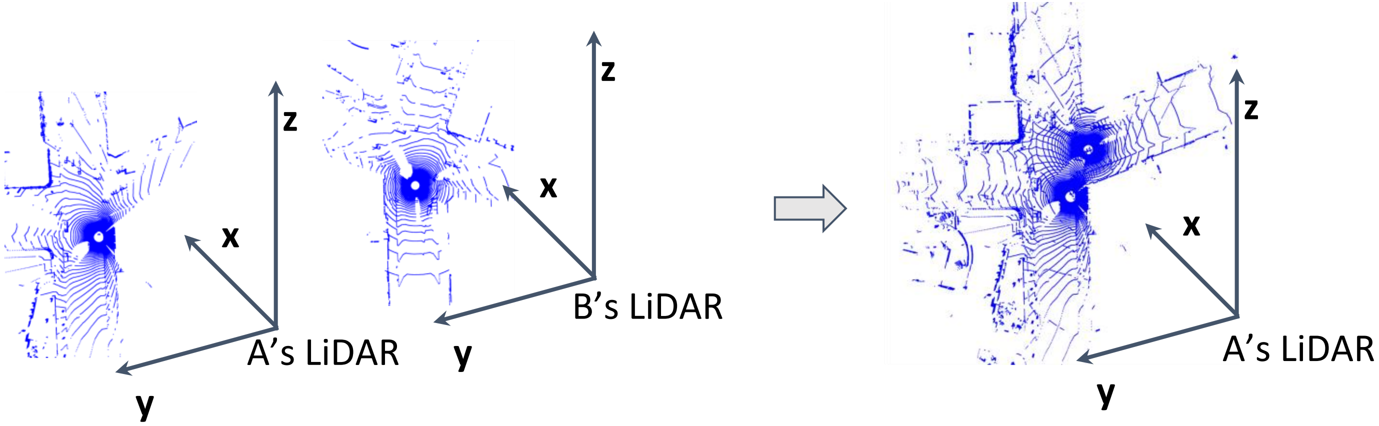


A visualized LiDAR point cloud (blue)

* C_1: chunk 1

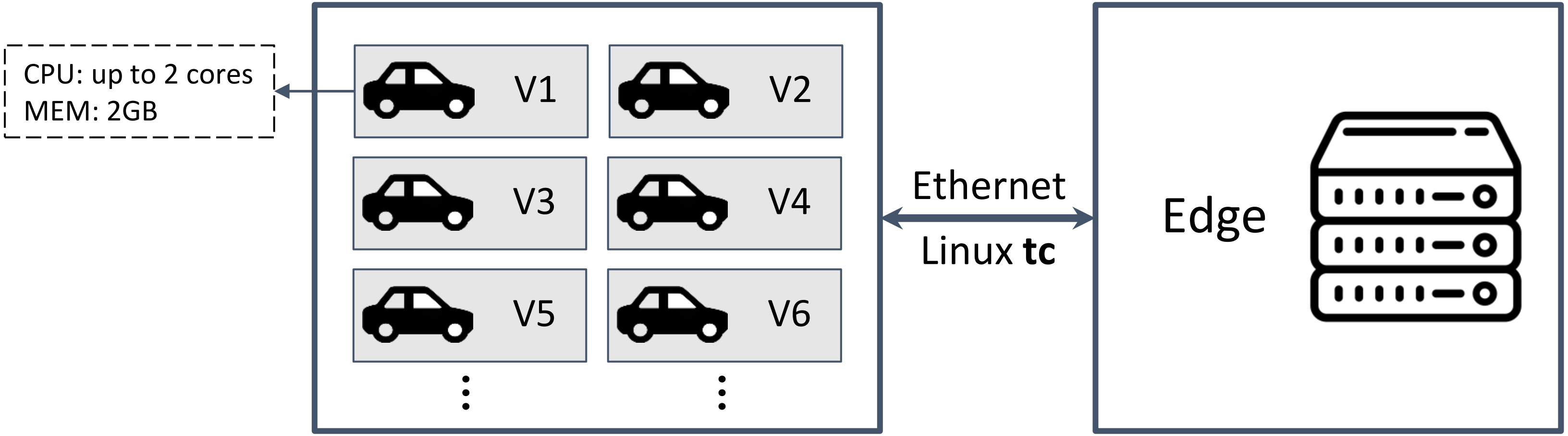
View Merging

- A point cloud is generated from the perspective of the detecting vehicle
  - *The origin is the LiDAR sensor mounted atop the vehicle.*
  - *Point clouds collected by different vehicles have different coordinate systems.*
- The edge merges the views of different vehicles



Evaluation - Experimental Setup

- EMP prototype in Java: https://github.com/Shawnxm/EMP
- Emulation testbed: EMP-edge instance + multiple EMP-vehicle instances



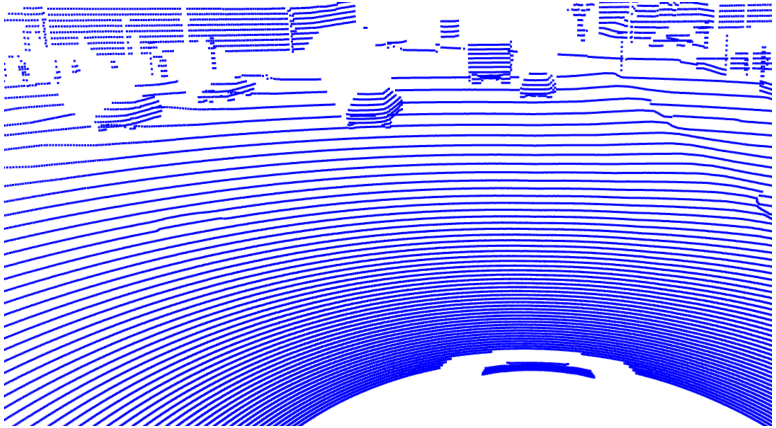
Evaluation - Experimental Setup

- Network conditions
  - *Trace collection*
    - Saturate the link with UDP data upload when driving at urban and rural areas
    - Measure the actual network throughput
  - *Network types*
    - LTE cellular networks (AT&T)
    - 60GHz WiFi networks (802.11ad, also considered in [1])
  - *Replay traces over Ethernet with Linux* **tc** *throttling the bandwidth*

[1] Qiu, Hang, et al. "Avr: Augmented vehicular reality." Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services. 2018.

Evaluation - Experimental Setup

- Sensor (LiDAR)
  - *Modify an existing tool* for generating driving data in a video game (GTA V)*
  - *Collect the **first** <u>multi-vehicle</u> dataset with <u>panoramic</u> LiDAR point clouds*



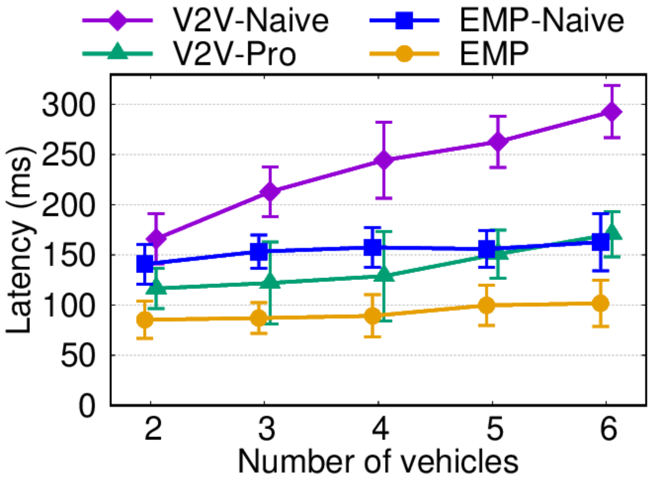
LiDAR point cloud

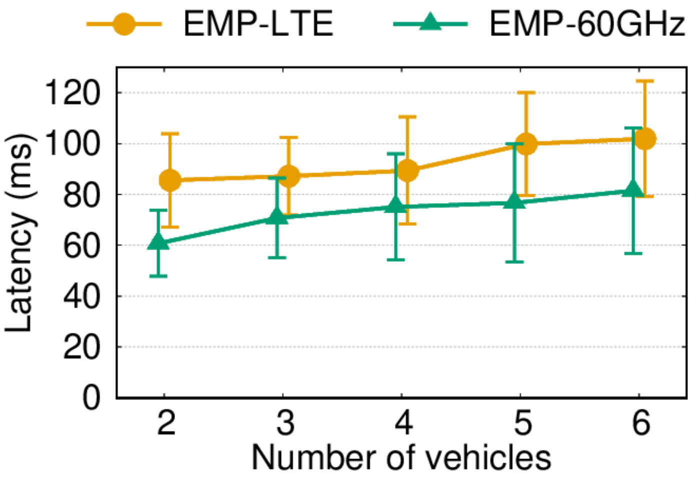
Camera image

* DeepGTAV-PreSIL: https://github.com/Shawnxm/DeepGTAV-PreSIL/tree/modified_for_emp

System Scalability

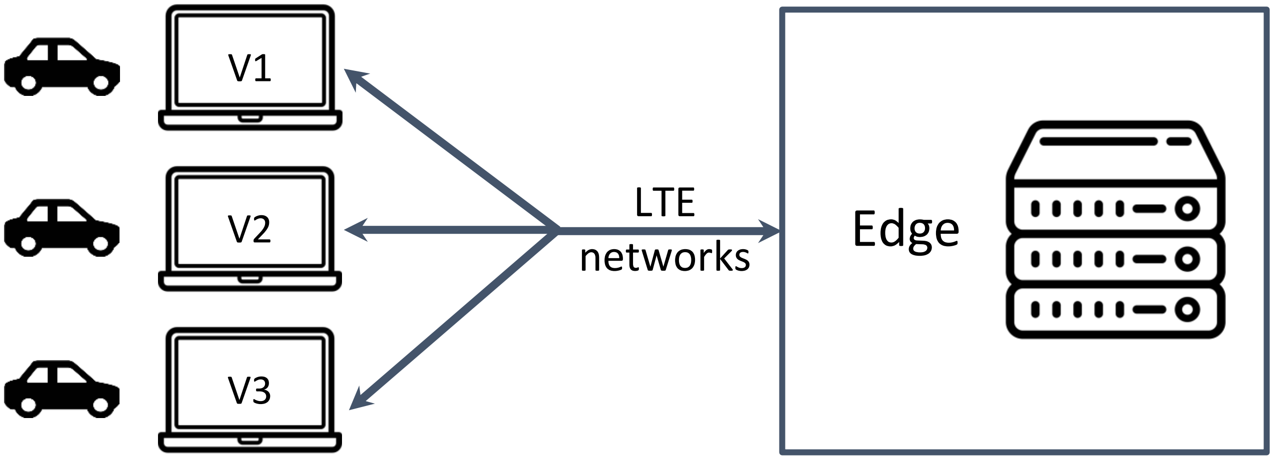
- Compare the end-to-end latency of four schemes
  - *EMP outperforms V2V sharing schemes by 49-65% in end-to-end overhead*
  - *Partitioning and scheduling effectively reduces latency*





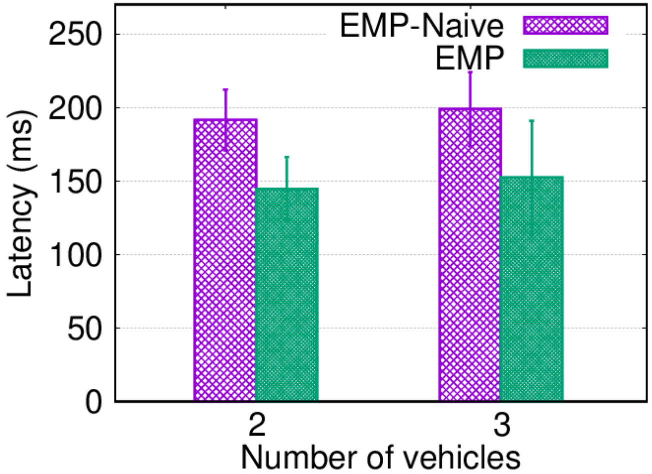
Evaluation - Experimental Setup

- Real-world driving test
  - *One machine runs the EMP-edge instance*
  - *Multiple vehicles each carries a laptop running EMP-vehicle instances*



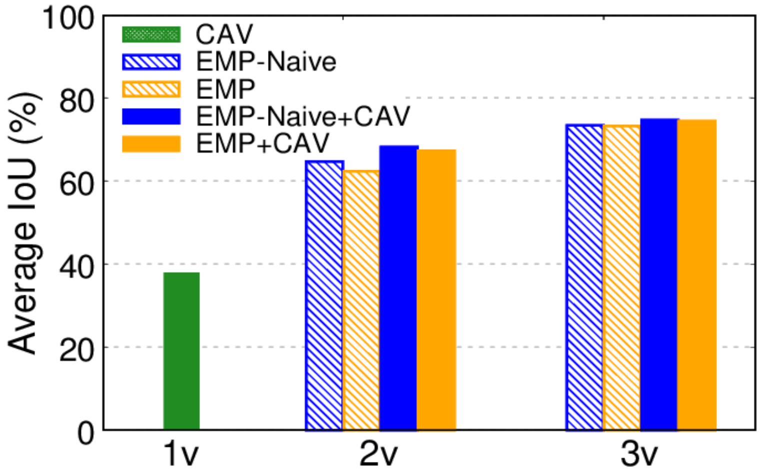
System Scalability

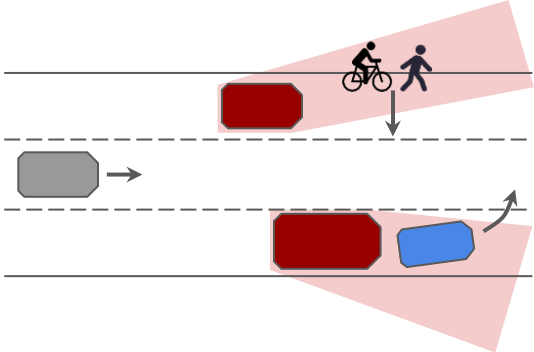
- Real-world driving tests
  - *The latency does not inflate when increasing the number of vehicles*
  - *REAP helps reduce the processing delay*



Perception Enhancement

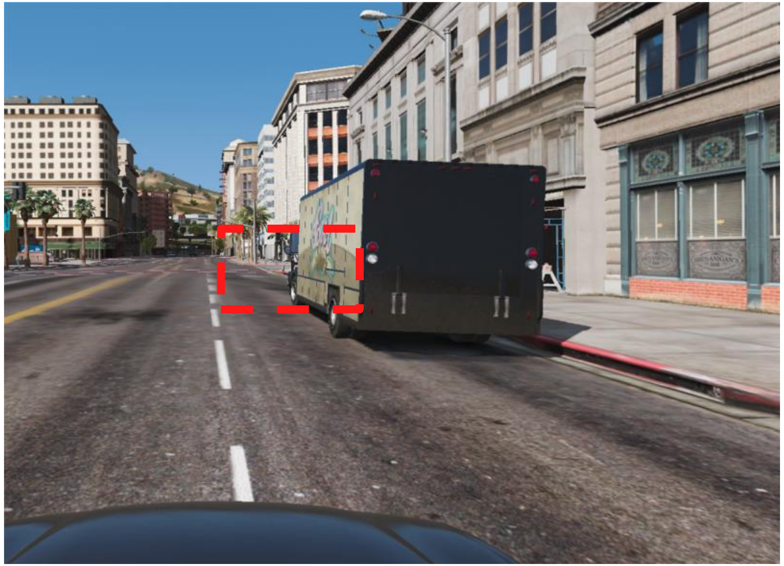
- Object detection accuracy
  - *Single-CAV (CAV) < Multi-CAV (EMP) < Combined (Edge+CAV)*
  - *REAP introduces negligible performance degradation while saving bandwidth*



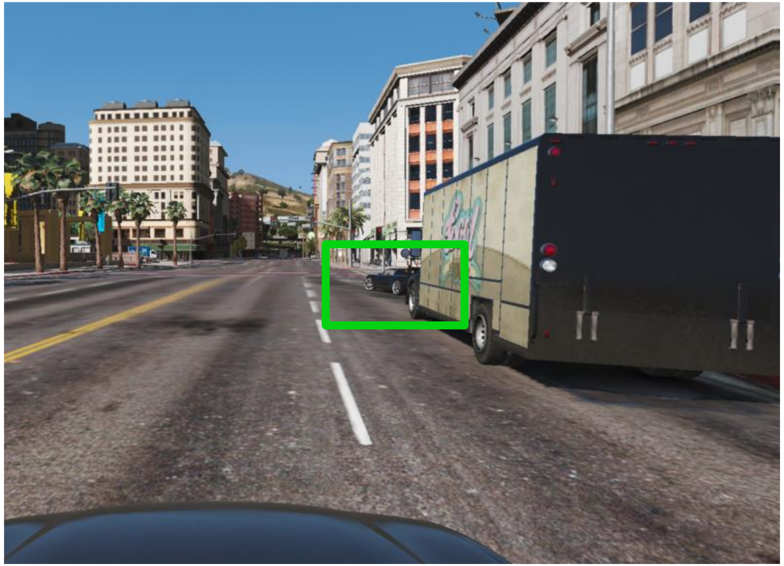


Road Hazard Avoidance

- Blind Spots (camera images)

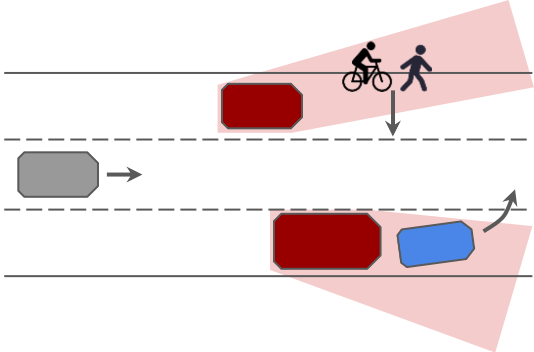


Frame 0

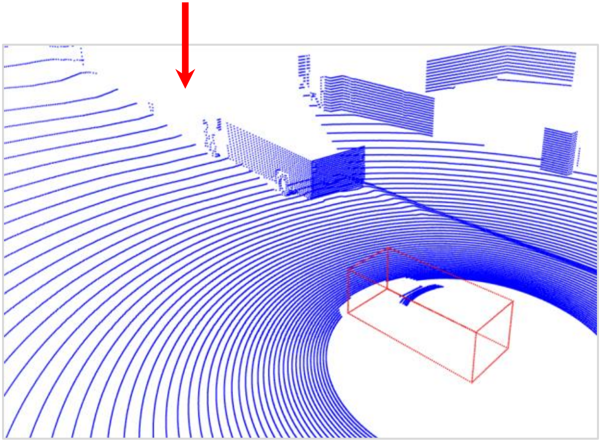


Frame 8

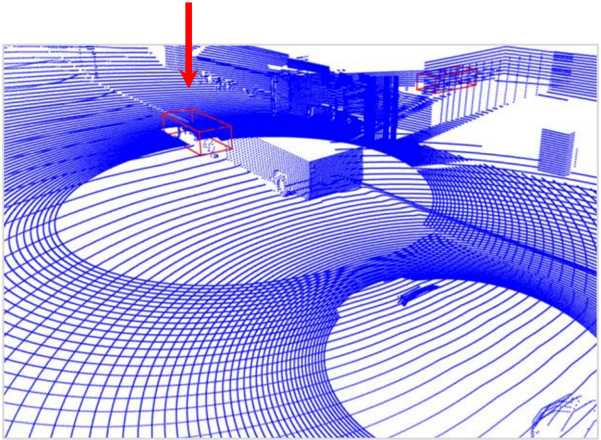
Road Hazard Avoidance



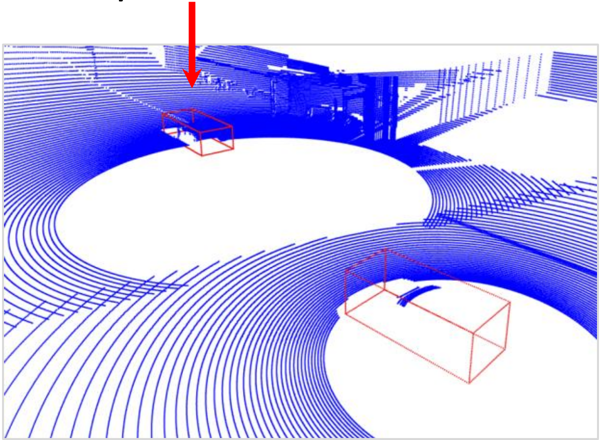
- Blind Spots (visualized point clouds): save 0.6s
    - *The blocked vehicle can be detected in both 2-vehicle setups*



Frame 0: 1-vehicle



Frame 0: 2-vehicle



Frame 0: 2-vehicle (REAP)

* 0.1*8 - (0.2 processing - 0.063 inference + 0.051 transmission) ≈ 0.6s

Conclusion

Thank you!

- Propose EMP, an edge-assisted multi-vehicle perception framework
- Develop robust algorithms for scalable, adaptive, and resource-efficient sensor data sharing under fluctuating network conditions
  - *A point cloud partitioning algorithm with bandwidth adaptation*
  - *A graph-based upload scheduling algorithm*
- Implement the *first* LiDAR-based cooperative perception system
  - *Outperforms V2V sharing schemes by 49-65% in end-to-end overhead*
  - *Reduce network bandwidth by 36-43% by adaptively uploading sensor data*
  - *Demonstrates its benefits of improved perception in realistic driving scenarios*