Moby: Empowering 2D Models for Efficient Point Cloud Analytics on the Edge

Jingzong Li, Yik Hong Cai, Libin Liu, Yu Mao, Chun Jason Xue, Hong Xu

香港城市大學
City University of Hong Kong

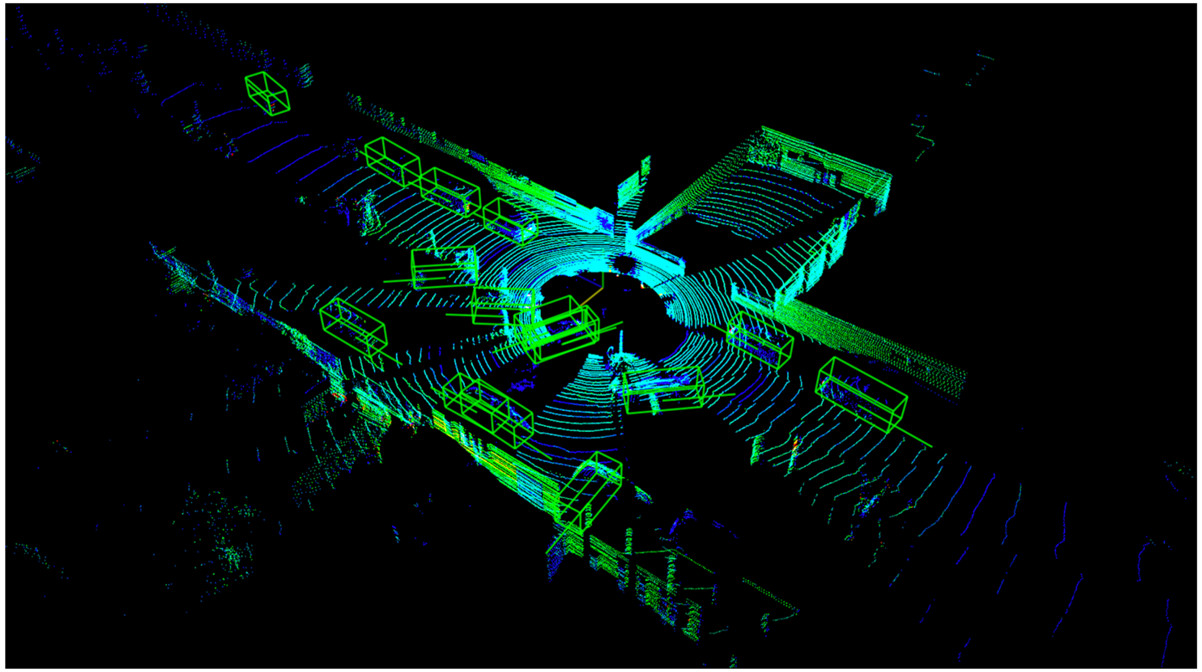
中关村实验室
ZGC Lab

香港中文大學
The Chinese University of Hong Kong

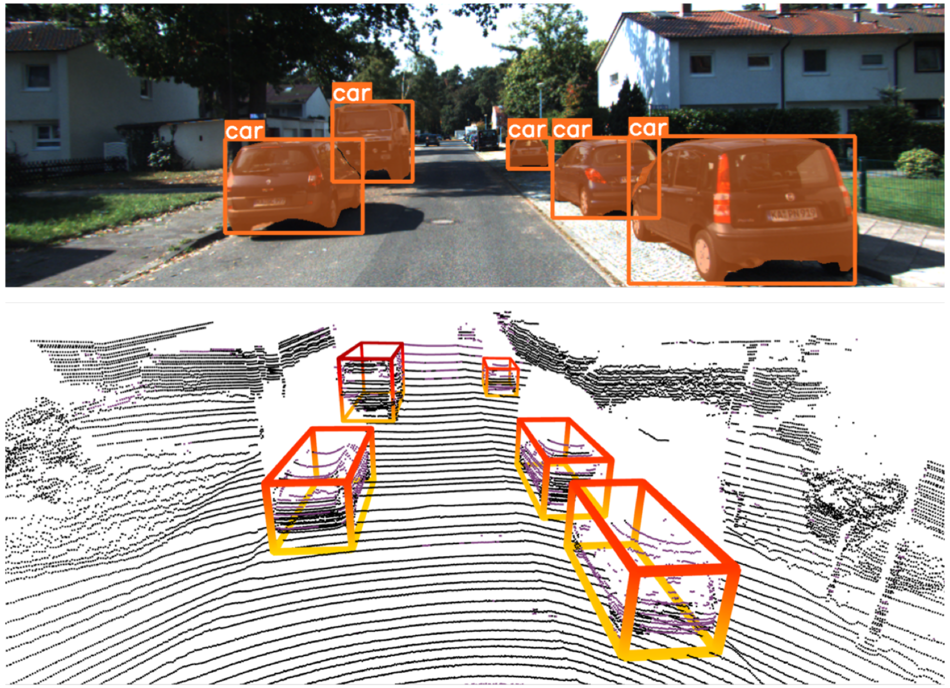
Point cloud data is everywhere

Point cloud analytics is vital to understand the environment

3D object detection is widely used in autonomous driving and robotics applications.



3D object detection

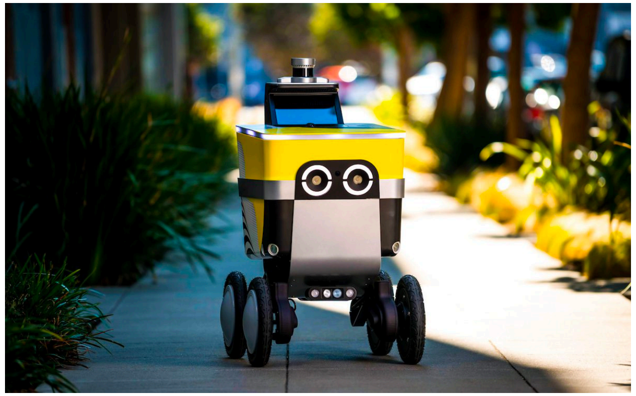


2D VS 3D object detection

The ever-increasing scenarios of edge computing

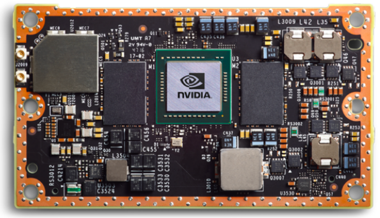
Efficiency is crucial for automotive driving and robotics applications

Logistics robot



Food delivery robot

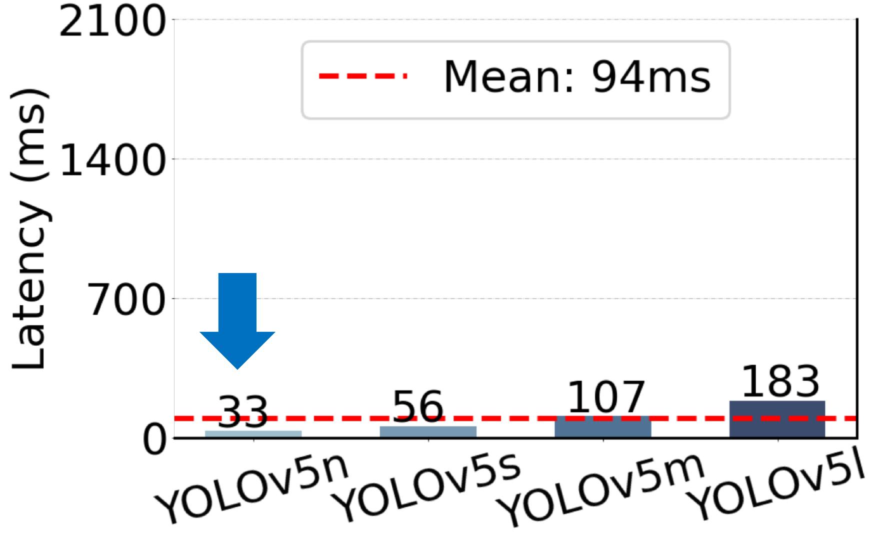
autonomous driving



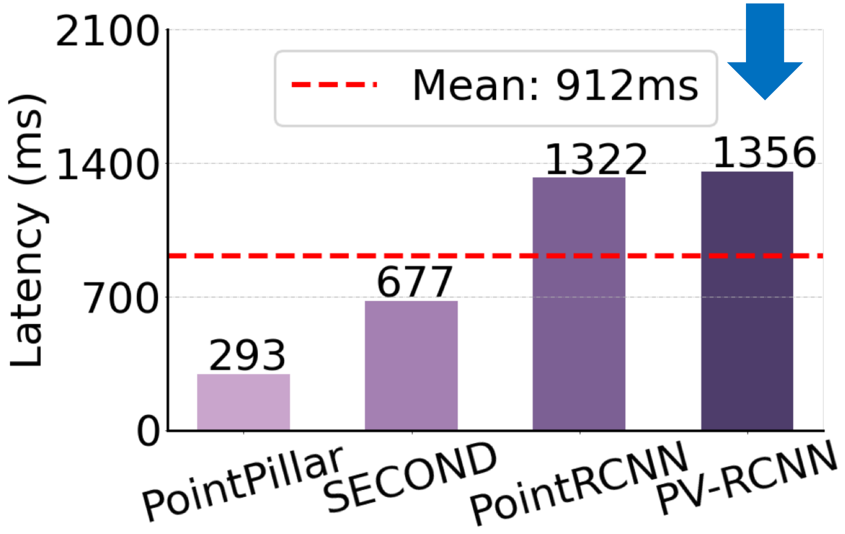
Edge devices

Deploying 3D object detection on edge is challenging

The latency of on-board inference on NVIDIA TX2:



2D model



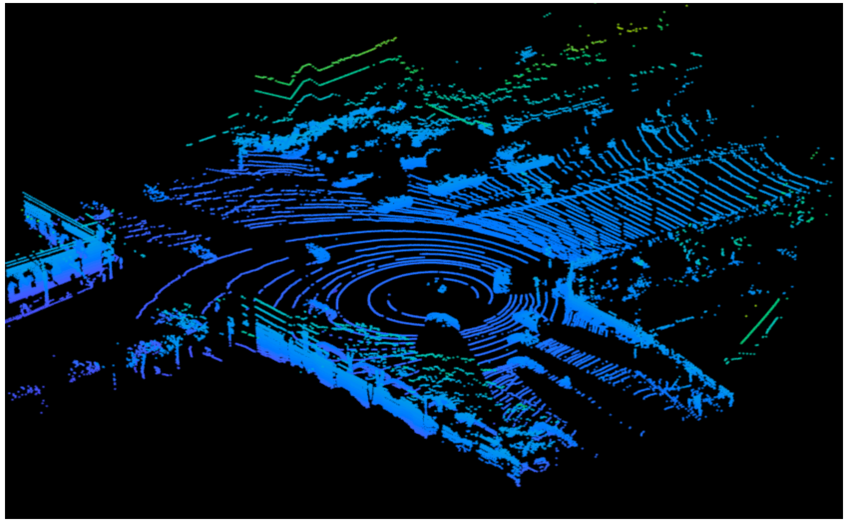
3D model

The average inference latency of 3D model is almost 10X that of 2D model

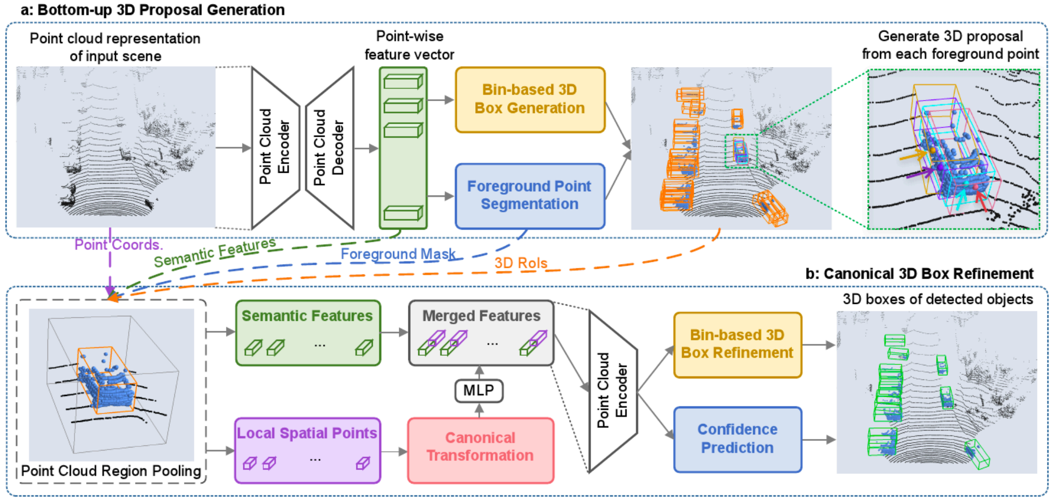
The inference latency of 3D detection model can be up to 41× of the 2D model

Deploying 3D object detection on edge is challenging

3D object detection is much more **compute-intensive** than 2D counterpart



Large amount of highly irregular, sparse, and unstructured data to process



More complicated architecture [1]

[1] Shi et al., PointRCNN: 3D Object Proposal Generation and Detection from Point Cloud, CVPR 2019

What if we offload the task to the cloud server for processing?

What if we offload the task to the server?

We measure the end-to-end latency of offloading to cloud server

Four representative point cloud-based models:

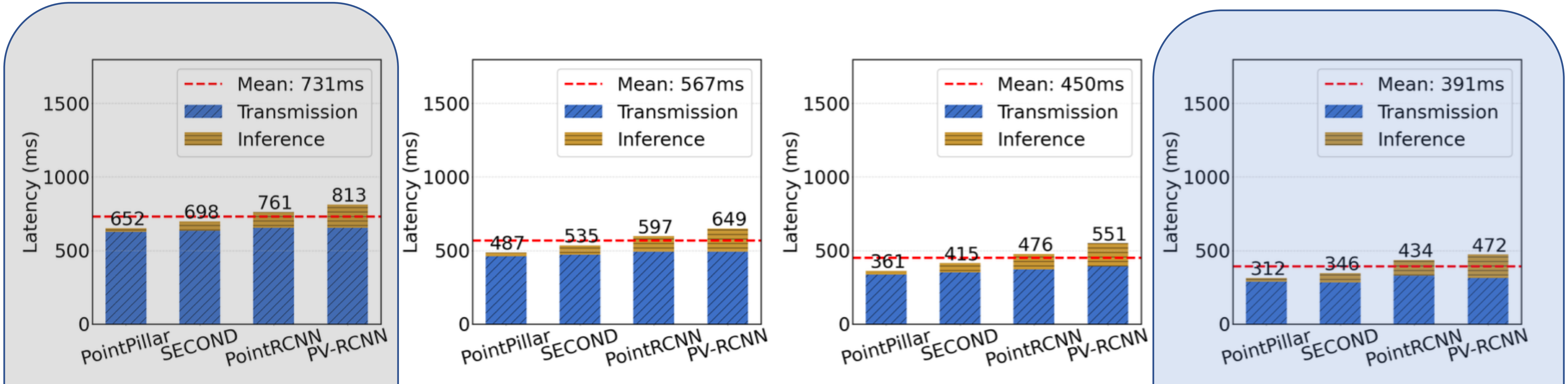
Model	PointPillar	SECOND	PointRCNN	PV-RCNN
Feature Extraction	Voxel based	Voxel based	Point based	Point-voxel based
Network Architecture	One Stage	One Stage	Two Stages	Two Stages

Four real-world 4G/LTE network traces:

Trace (Mbps)	Mean (± Std)	Range	$P_{25\%}$	Median	$P_{75\%}$
FCC-1	11.89 (± 2.83)	[7.76, 17.76]	9.09	12.08	13.42
FCC-2	16.69 (± 4.69)	[8.824, 28.157]	13.91	16.07	19.43
Belgium-1	23.89 (± 4.93)	[16.02, 33.33]	19.84	23.46	27.73
Belgium-2	29.60 (± 4.92)	[20.17, 37.345]	25.18	30.761	32.76

What if we offload the task to the server?

The transmission of point cloud dominates the end-to-end latency.



(a) FCC-1 (Avg. 11.89Mbps) (b) FCC-2 (Avg. 16.69Mbps) (c) Belgium-1 (Avg. 23.89Mbps) (d) Belgium-2 (Avg. 28.60Mbps)

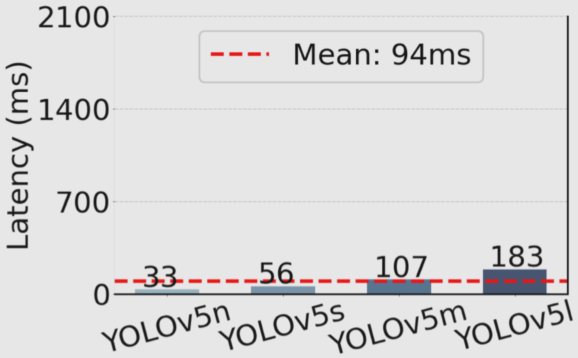
Latency grows ← **Network deteriorates**

Offloading all frames to the cloud for inference is also impractical

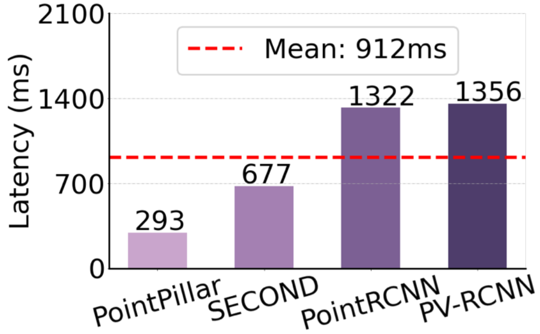
Motivation

Low latency

2D model

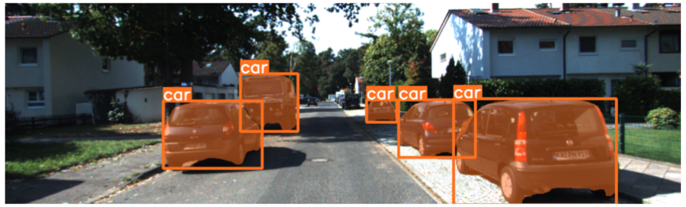


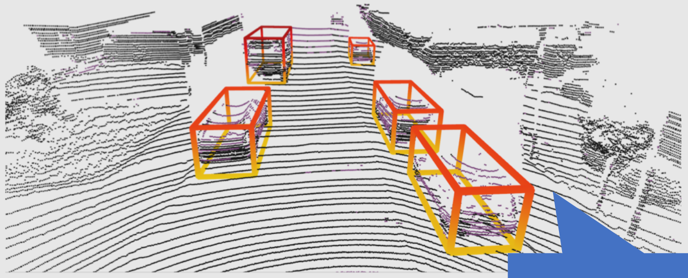
3D model



Significant lower inference time of 2D object detection

Close correspondence between the 2D and 3D bounding boxes





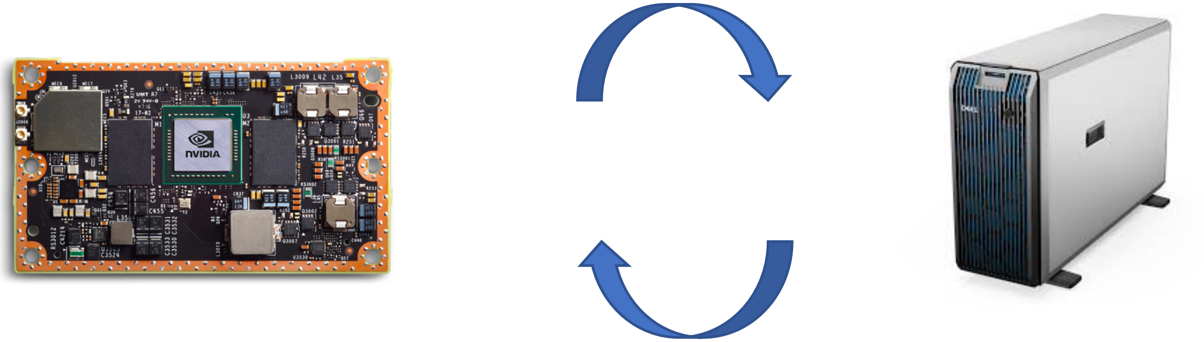
3D output

Can we use 2D detection models to extrapolate the 3D bounding boxes?

Motivation

Both ill-suited for 3D object detection

Edge-only **Cloud-only**



Can we better orchestrate the edge and cloud computation?

Main idea

- Rather than relying on heavy DNN-based 3D detectors, we propose a light-weight **2D-to-3D transformation** approach that generates 3D bounding boxes based on 2D model outputs.
    - Challenge 1: At the frame level, how can Moby **transform** 2D bounding boxes into 3D ones **accurately and efficiently**?
- Evidently, this approach would require DNN-based 3D detection on a few **anchor frames** to provide the 3D information.
    - Challenge 2: Across frames, as the error of transformation accumulates over time, how can Moby **monitor** the accuracy drop and **decide** the offloading timing?

Main idea

- Challenge 1: At the frame level, how can Moby **transform** 2D bounding boxes into 3D ones **accurately and efficiently**?

→ Tracking-based association

→ 2D-to-3D transformation

- Challenge 2: Across frames, as the error of transformation accumulates over time, how can Moby **monitor** the accuracy drop and **decide** the offloading timing?

→ Frame offloading scheduler

Moby's system workflow

Preparation

1 2

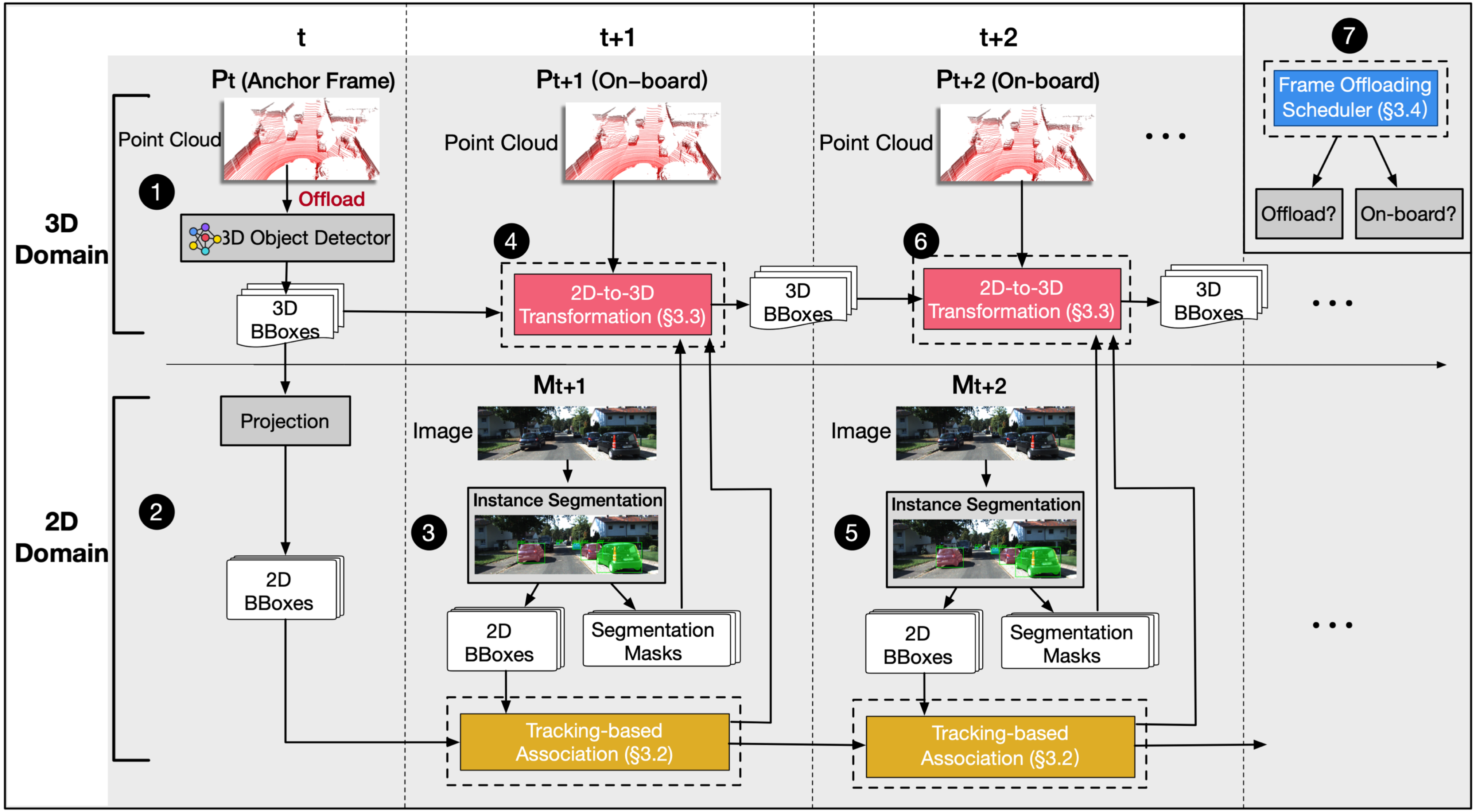
Transformation

3 4

5 6

Scheduling

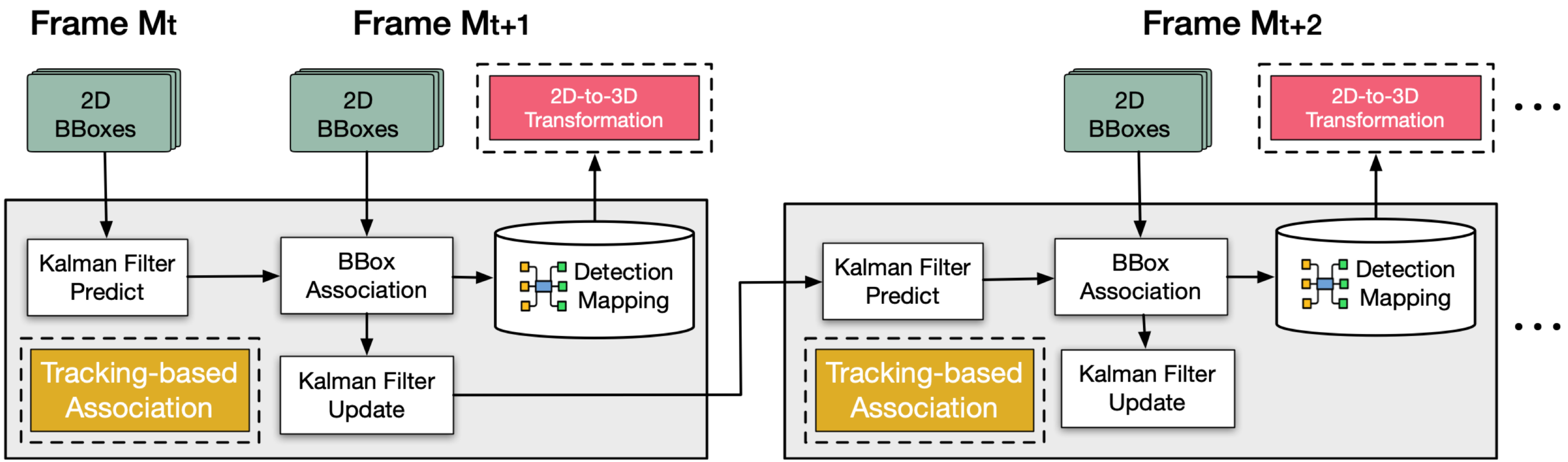
7



Tracking-based Association

Utilizing tracking in the 2D domain to build the mapping between results in two adjacent frames.

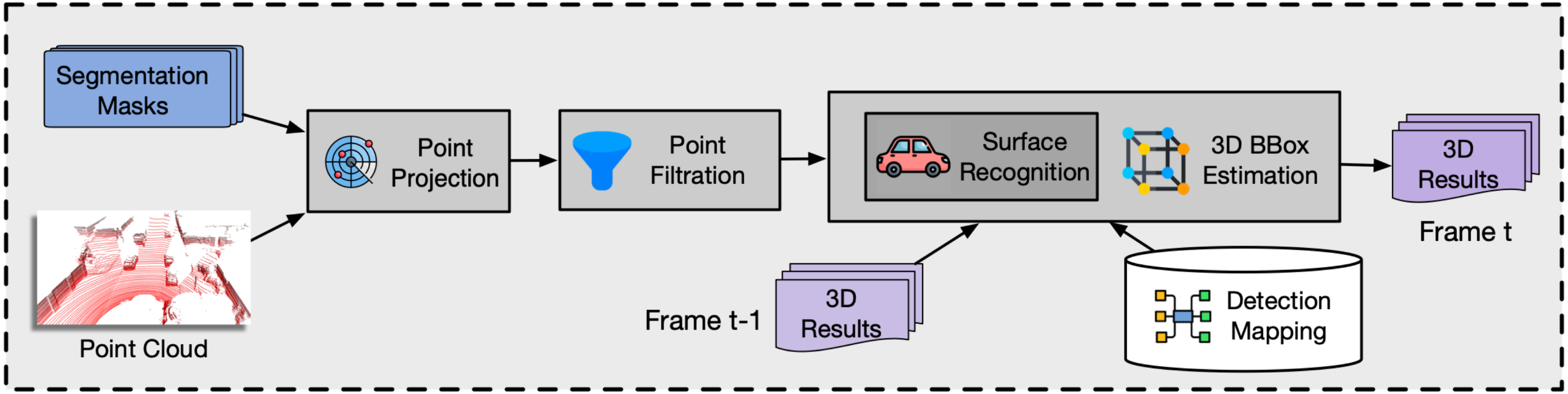
- On-device 2D Inference
- Kalman Filter-based Tracking



2D-to-3D Transformation

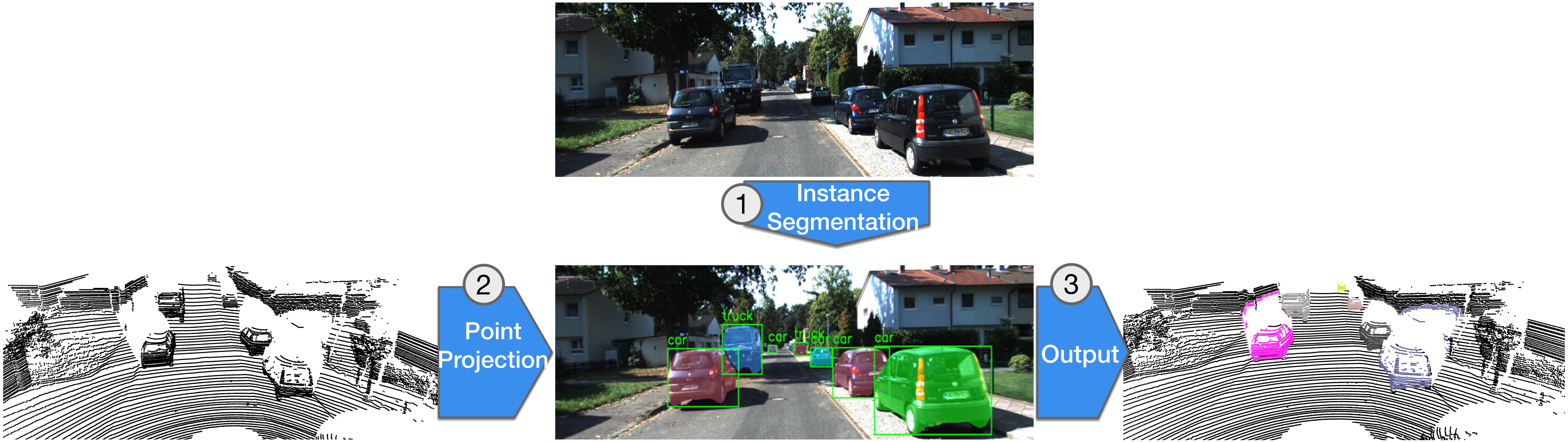
Transform bounding box from 2D domain to 3D domain

- Point Projection
- Point Filtration
- 3D bounding box estimation

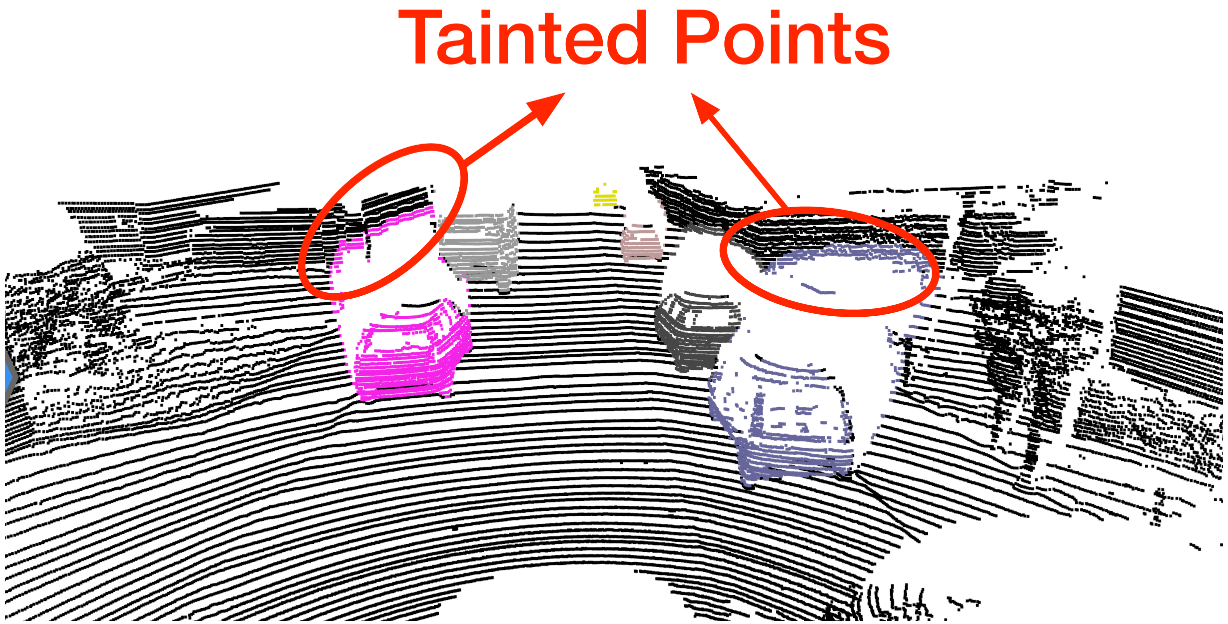


Point Projection

Transfer 2D semantics to 3D point cloud and obtain point clusters



Point Projection



Reasons for tainted points:

- Segmentation result is **imperfect**;
- The projection from point cloud to pixels is **many to one**.

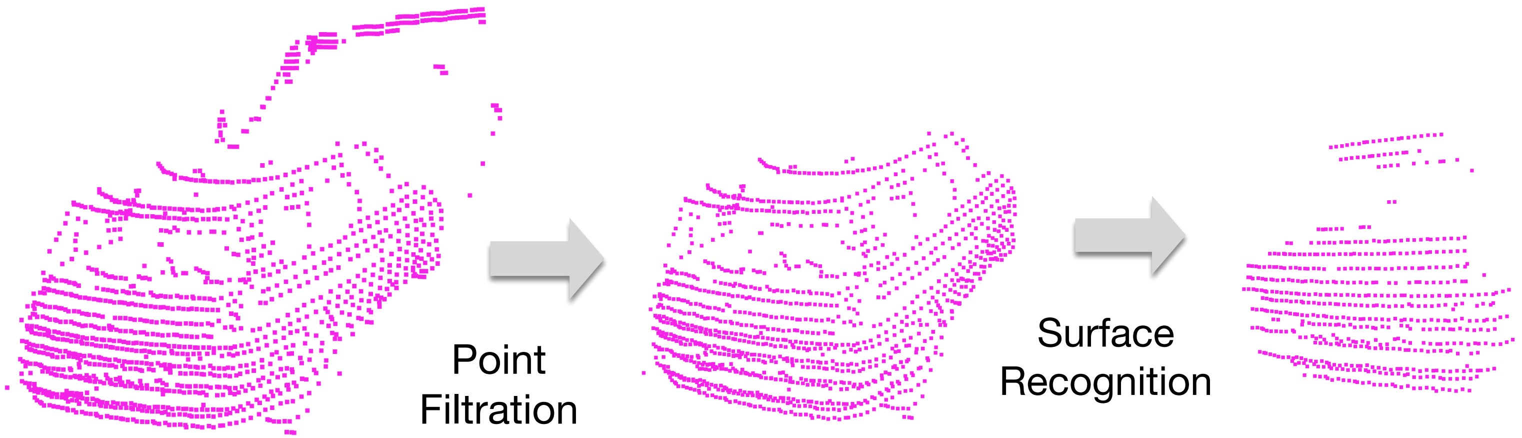
3D Bounding Box Estimation

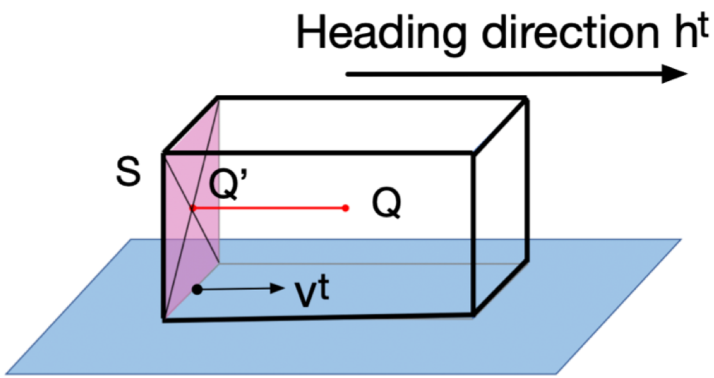
Estimate each object's 3D bounding box based on its point cluster

3D bounding box: **[x, y, z, l, w, h, θ]**

Heading angle

θ





3D Bounding Box Estimation

Estimate each object's 3D bounding box based on its point cluster

3D bounding box: **[x, y, z, l, w, h, θ]**

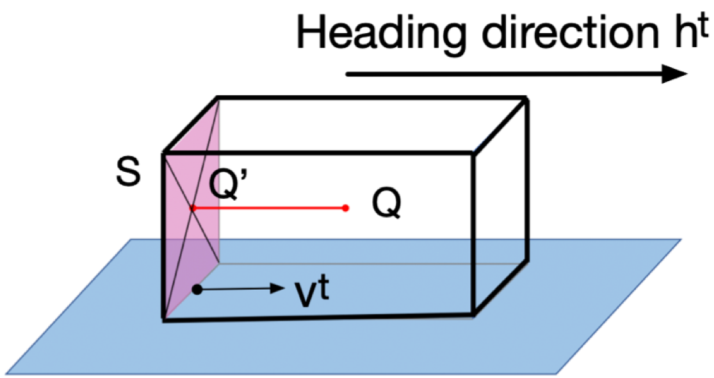
Heading angle

θ

BBox size

l, w, h

Tracking-based Association (§3.2)



3D Bounding Box Estimation

Estimate each object's 3D bounding box based on its point cluster

3D bounding box: **[x, y, z, l, w, h, θ]**

Heading angle

θ

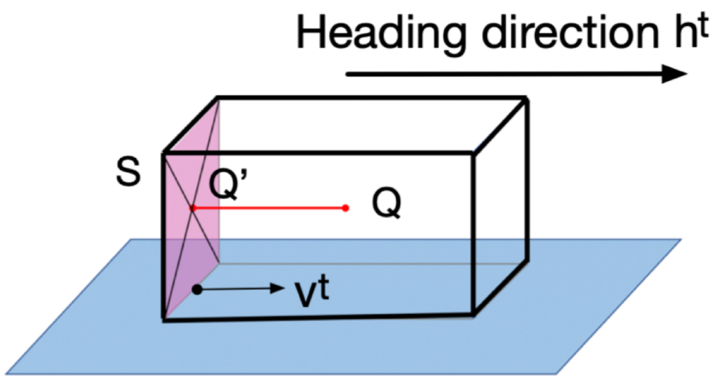
BBox size

l, w, h

Object center

x, y, z

Tracking-based Association (§3.2)

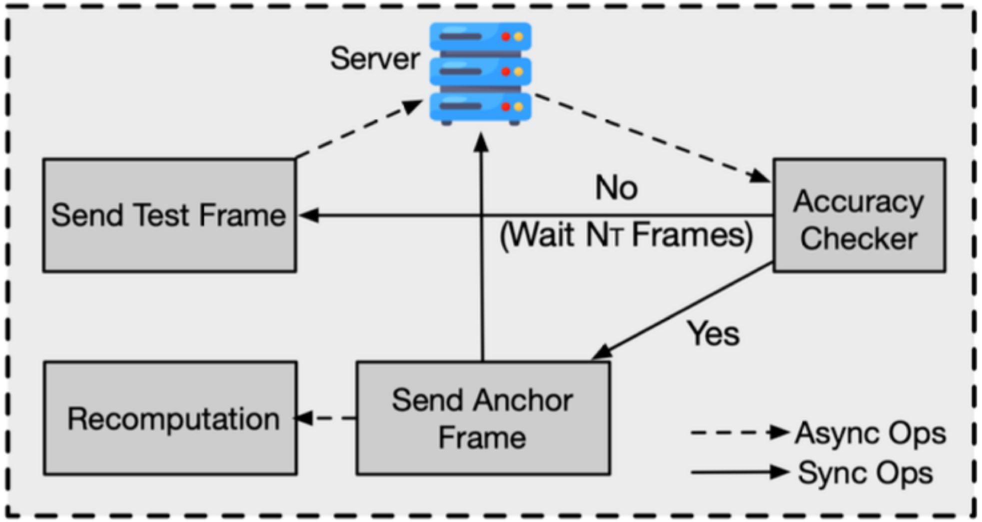


Frame Offloading Scheduler

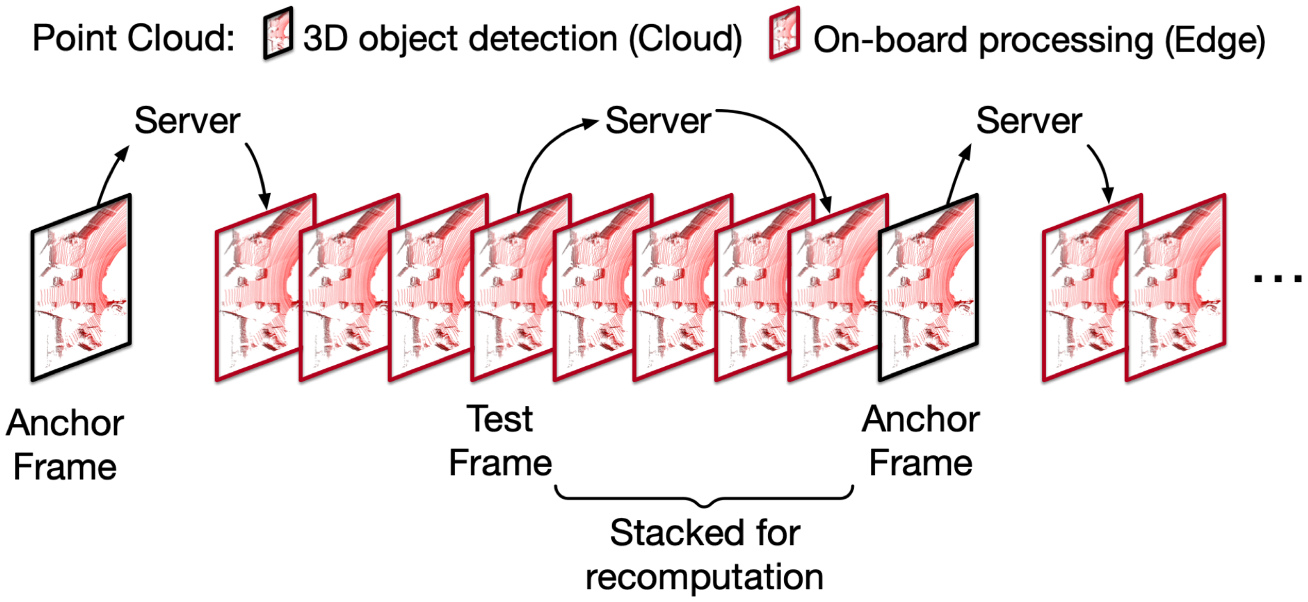
Decide when to offload a new anchor frame to the cloud for processing

It must: 1) introduce little overhead, and 2) efficiently detecte error accumulation

Our solution: send a test frame to the cloud every N frames



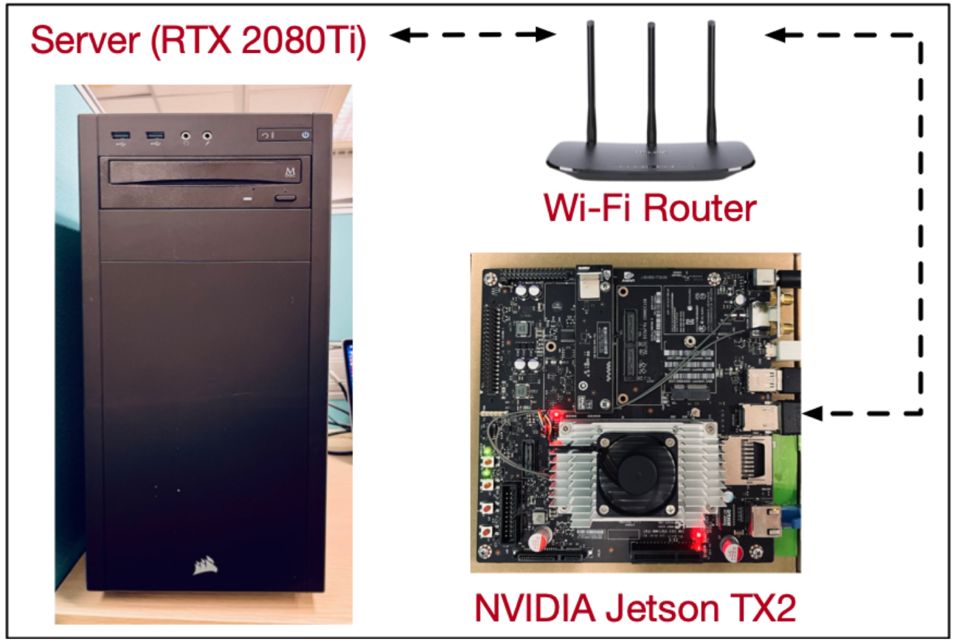
Overview



Example

Evaluation – Experiment Settings

Testbed: We run our experiments using a Jetson TX2 as the edge device and a desktop equipped with an Intel i7-9700K CPU and an RTX 2080Ti GPU as the server.



Dataset: KITTI dataset [1], a real-world autonomous driving benchmark.

Models: use YOLOv5n as Moby's default instance segmentation model, and the same 3D object detection model as the baseline systems.

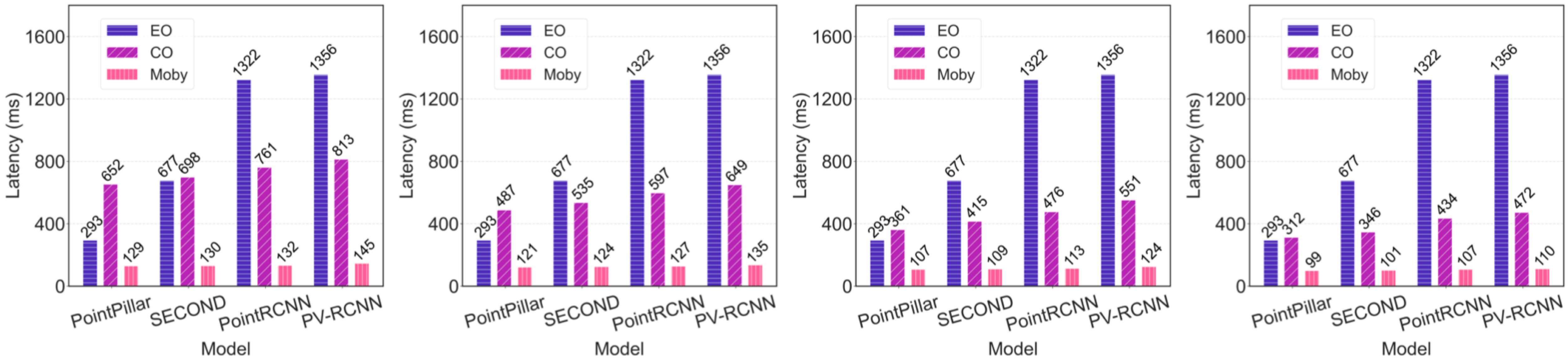
Metrics:

- End-to-end latency
- 3D Detection Accuracy (F1)

[1] Geiger et al., Are We Ready for Autonomous Driving? The KITTI Vision Benchmark Suite, CVPR 2012

Evaluation - Deployment Approaches

Two deployment approaches:
- **Edge Only (EO):** 3D models are deployed on the edge device only to run inference.
- **Cloud Only (CO):** fully offloads point cloud over 4G/LTE networks to the server for inference.



(a) FCC-1 (Avg. 11.89Mbps) (b) FCC-2 (Avg. 16.69Mbps) (c) Belgium-1 (Avg. 23.89Mbps) (d) Belgium-2 (Avg. 28.60Mbps)

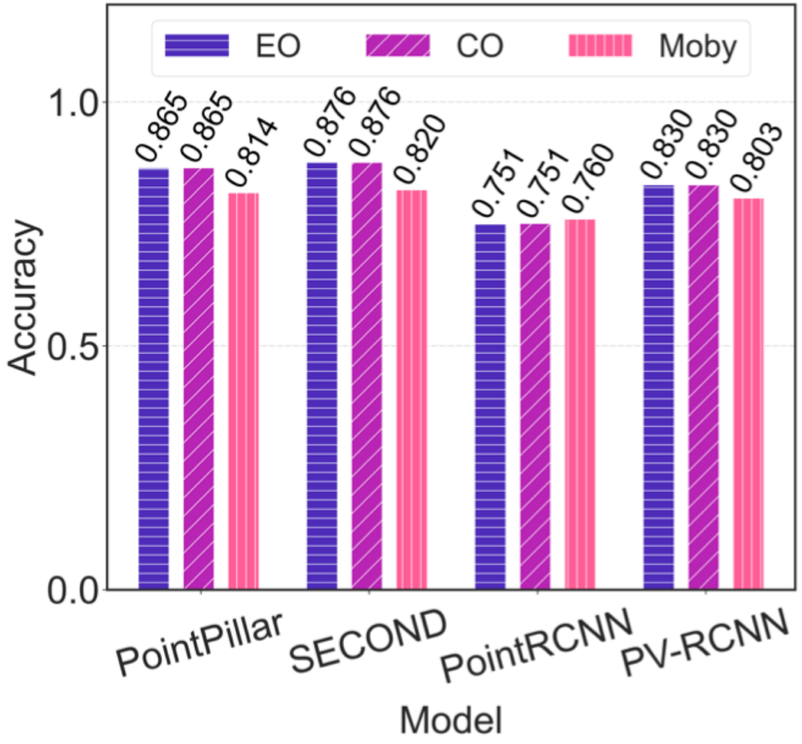
Latency

The latency reduction ranges from 56.0% to 91.9%.

Evaluation - Deployment Approaches

Two deployment approaches:

- Edge Only **(EO)**
- Cloud Only **(CO)**



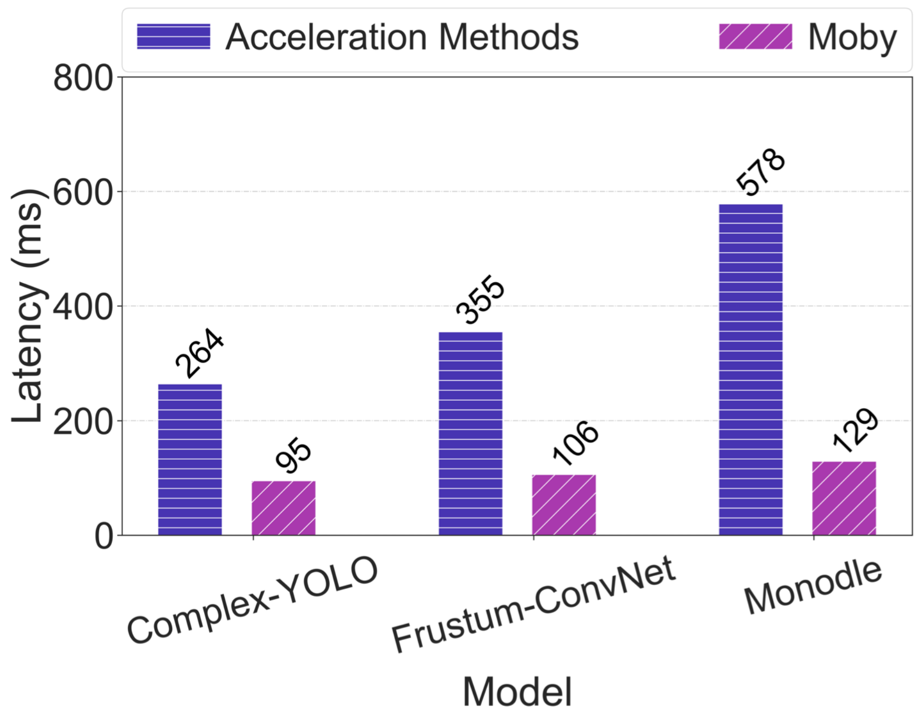
Accuracy

Accuracy drops slightly between 0.027 to 0.056, which is negligible.

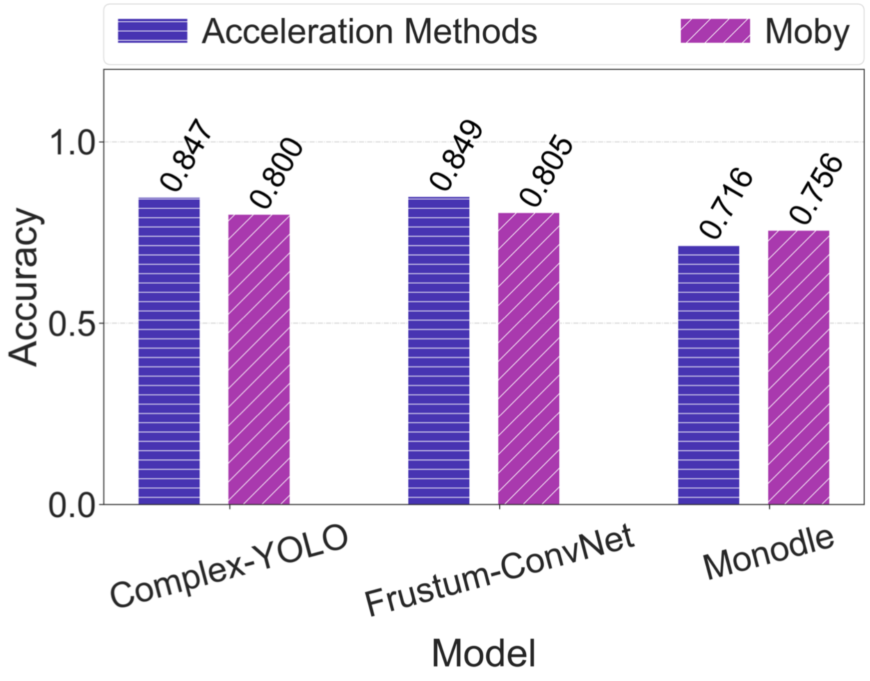
Evaluation – Acceleration Methods

Comparison of Moby and three **acceleration methods:**

- **Complex-YOLO:** converts point cloud data to birds-eye-view RGB maps
- **Frustum-ConvNet:** utilizes 2D region proposals to narrow down the 3D space
- **Monodle:** State-of-the-art image-based 3D detection approach



Latency



Accuracy

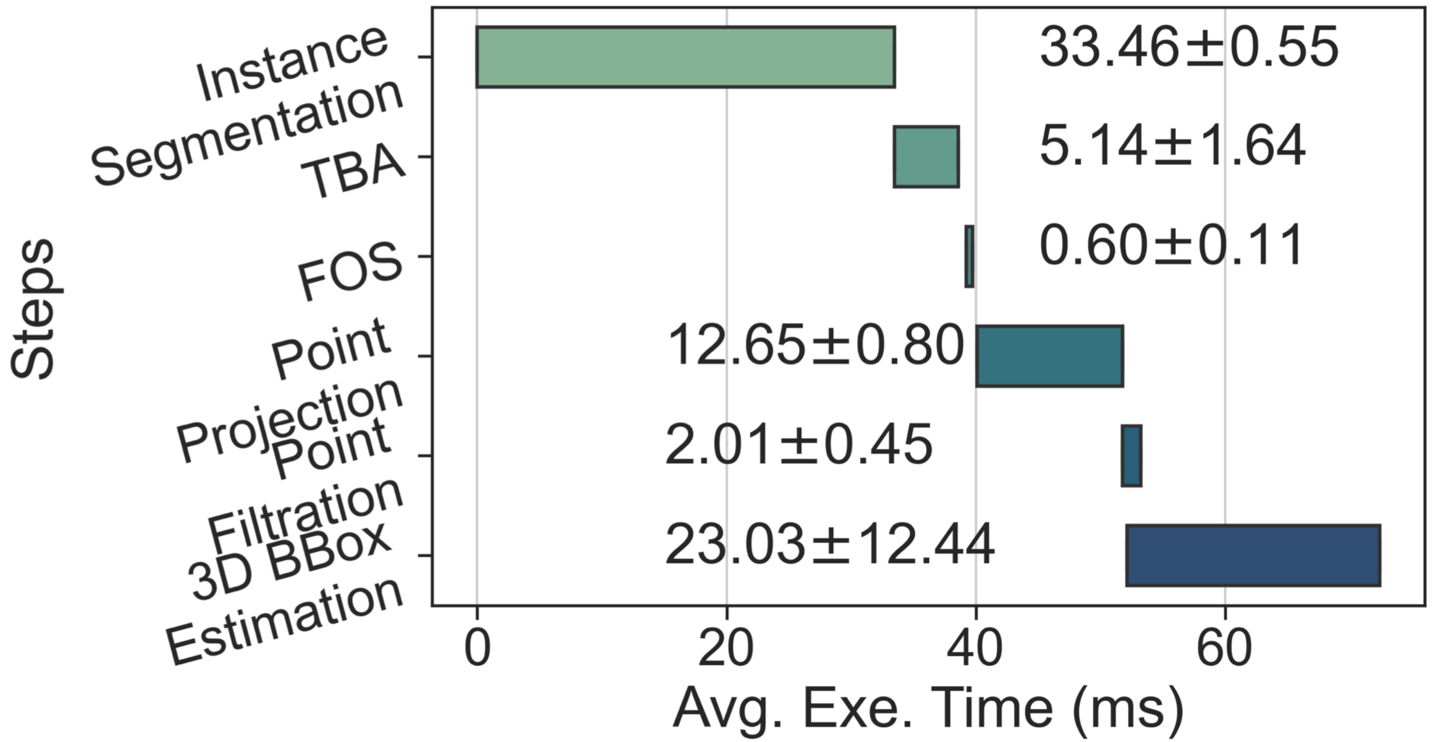
Evaluation – Impact of each component

Impact of each design component.

	Components	Accuracy	Latency (ms)	On-board Latency (ms)
2D-to-3D transformation	TRS	0.762	88.44	88.44
+ Frame offloading scheduler	TRS+FOS	0.787	112.06	89.45
+ Tracking-based association	TRS+FOS+TBA	0.814	99.23	76.29

Evaluation – Overheads

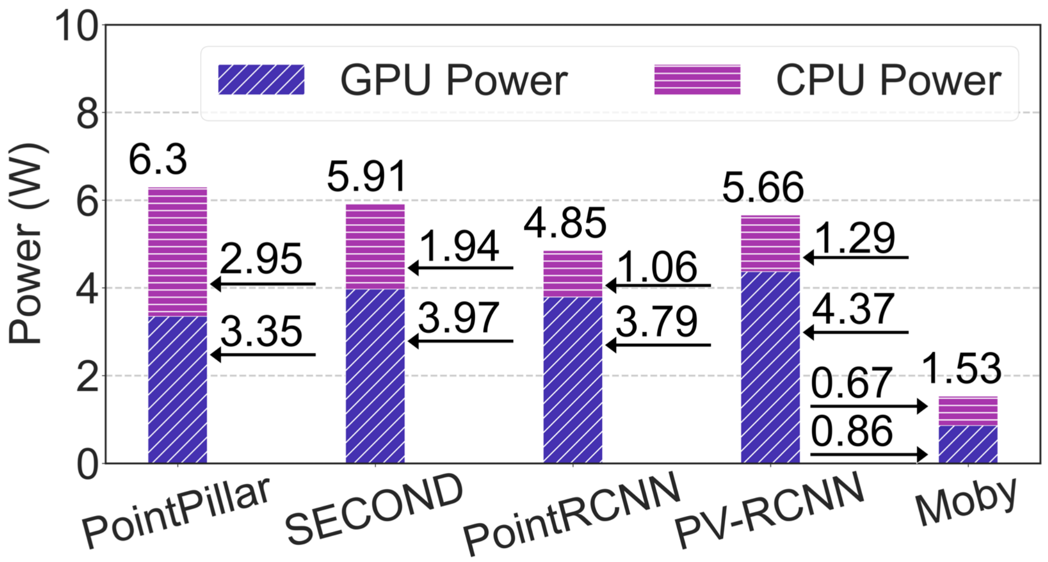
The avg. execution time of key steps over 300 runs



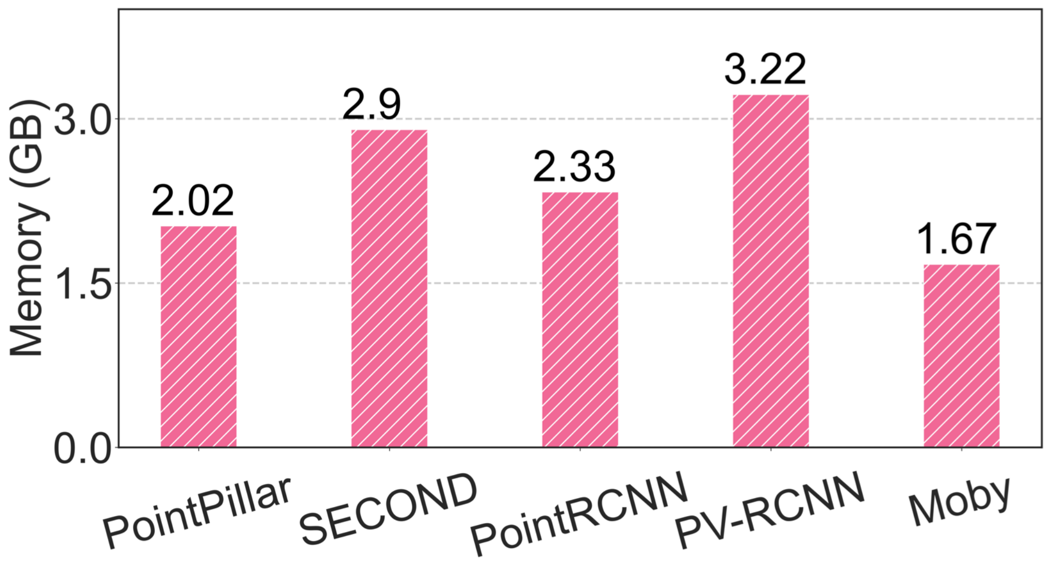
Instance segmentation takes the longest, accounting for 43.9%

Evaluation – System Efficiency

Energy consumption



Memory footprint



Memory reduction ranges from 17.3% to 48.1%.

Conclusion

- **Problem**: Point cloud analytics tasks pose severe burden for resource-constrained edge devices, edge-only and cloud-only are both ill-suited.
- **Our contribution**: Moby, the first work to propose such 2D-to-3D transformation, which is capable of transferring vision semantics to 3D space and leveraging a light-weight geometric method to construct 3D bounding boxes swiftly and accurately.
- **Results**: Moby achieves significant latency reduction with only modest accuracy loss.