THE CHIPS TO SYSTEMS CONFERENCE

SPONSORED BY: CEDA IEEE Council on Electronic Design Automation, SIGDA ACM

FlashFPS: Efficient Farthest Point Sampling for Large-Scale Point Clouds via Pruning and Caching

Yuzhe Fu, Hancheng Ye, Cong Guo, Junyao Zhang, Qinsi Wang, Yueqian Lin, Changchun Zhou, Hai "Helen" Li and Yiran Chen

Duke University | DAC 2026

Duke | CENTER *for* COMPUTATIONAL EVOLUTIONARY INTELLIGENCE

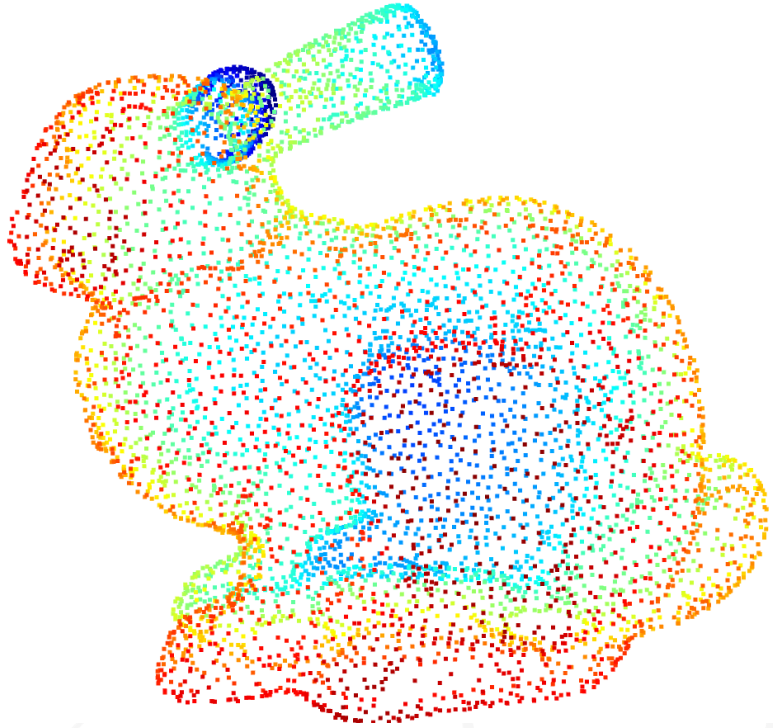
1. Background

2. Method: FlashFPS

3. Experimental Results

Point Cloud Data

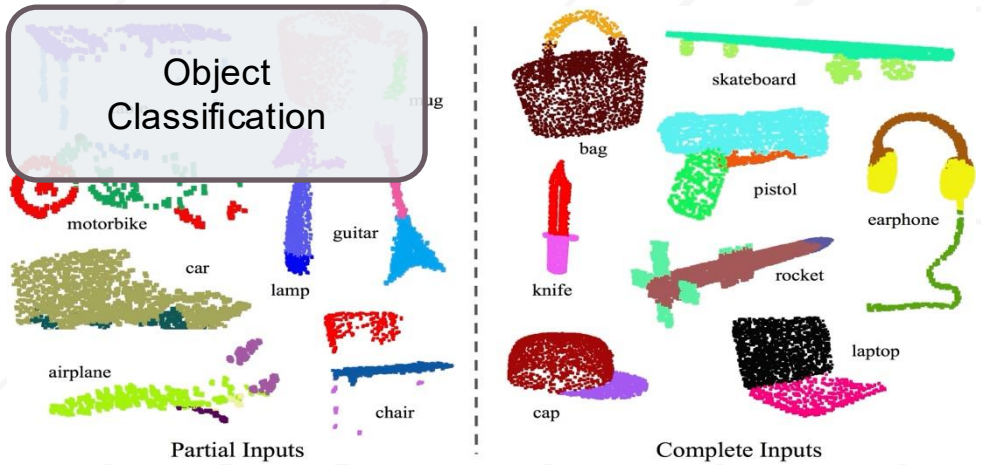
2D Image
Pixels: RGB values

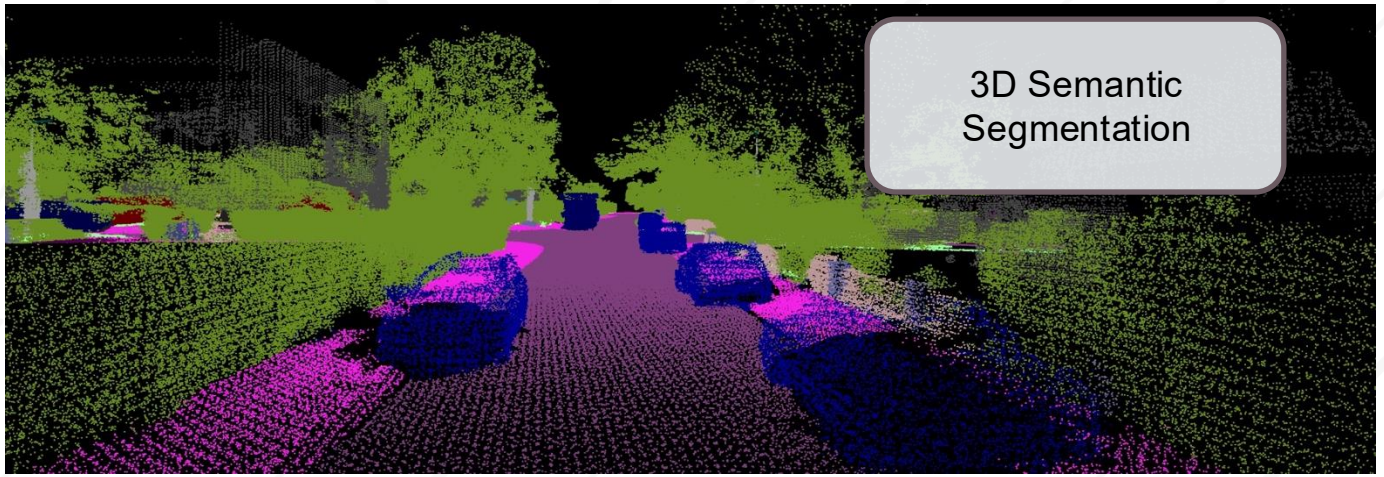


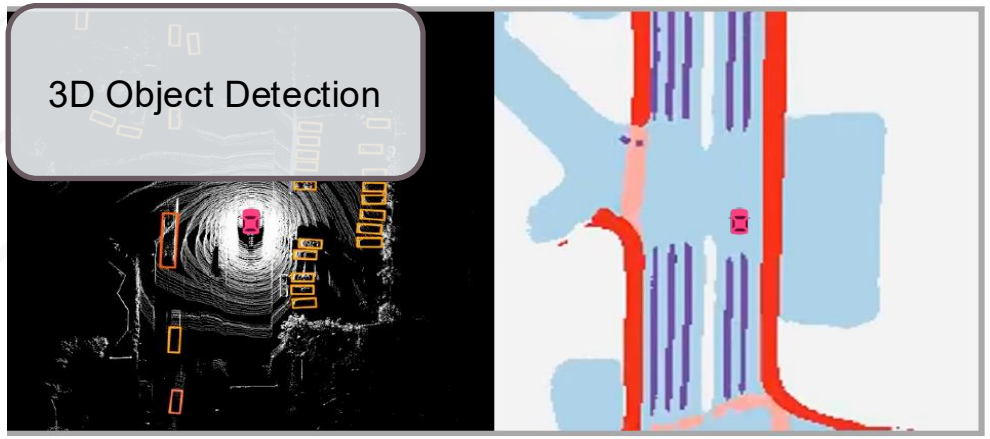
3D Point Cloud
Points: (x, y, z), Feature, …
Richer Environment Information

Zhou et al., Open3D: A modern library for 3D data processing. arXiv, 2018.
https://pixabay.com/zh/photos/rabbit-nature-wildlife-animal-5469252/

Point Cloud in Deep Learning (PNNs)



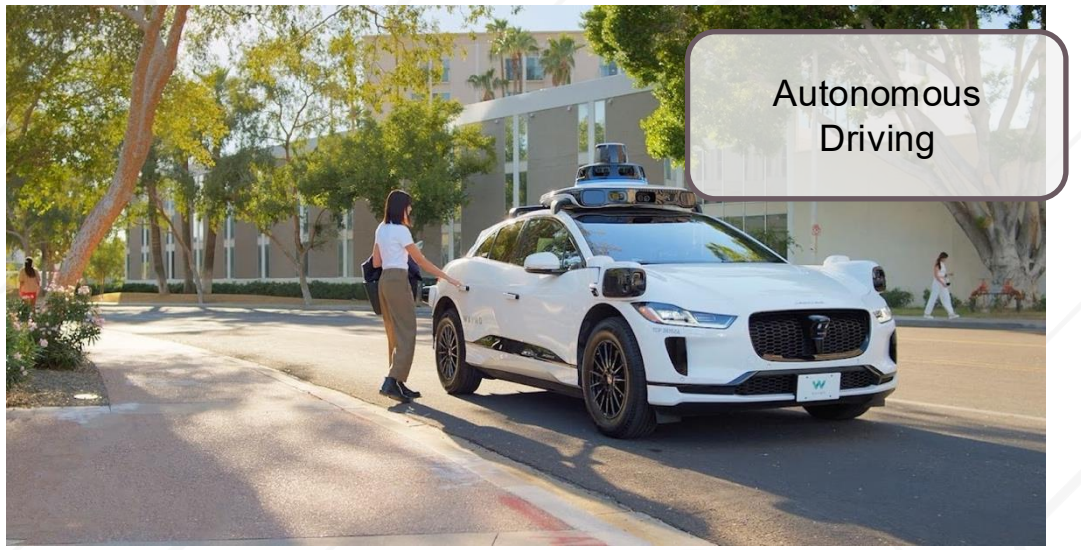




Qi et al., PointNet. CVPR, 2017.
Tang et al., 3D-NAS, ECCV, 2020.

Vallet et al., urban point cloud benchmark, CG 2015.
ANDANTE: 3D Object Detection.

Point Cloud Application



Apple's Vision Pro with M5
iPad - Apple

Waymo Driver
AI-enabled control system helps autonomous drones | MIT News

Point Cloud Application

VR Glasses

Autonomous Driving

AR in

Drones

LiDAR Scanner

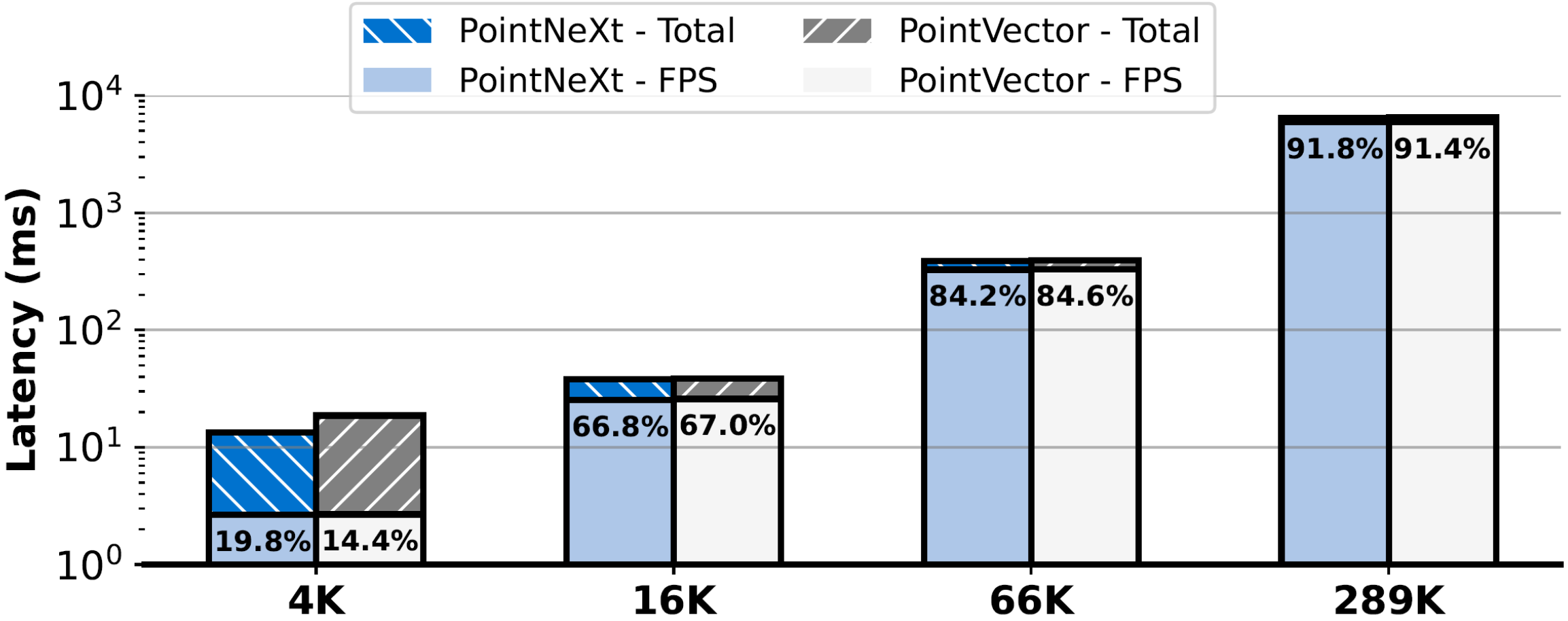
Edge Application

Low latency
High Network Accuracy

Apple's Vision Pro with M5
iPad - Apple

Waymo Driver
AI-enabled control system helps autonomous drones | MIT News

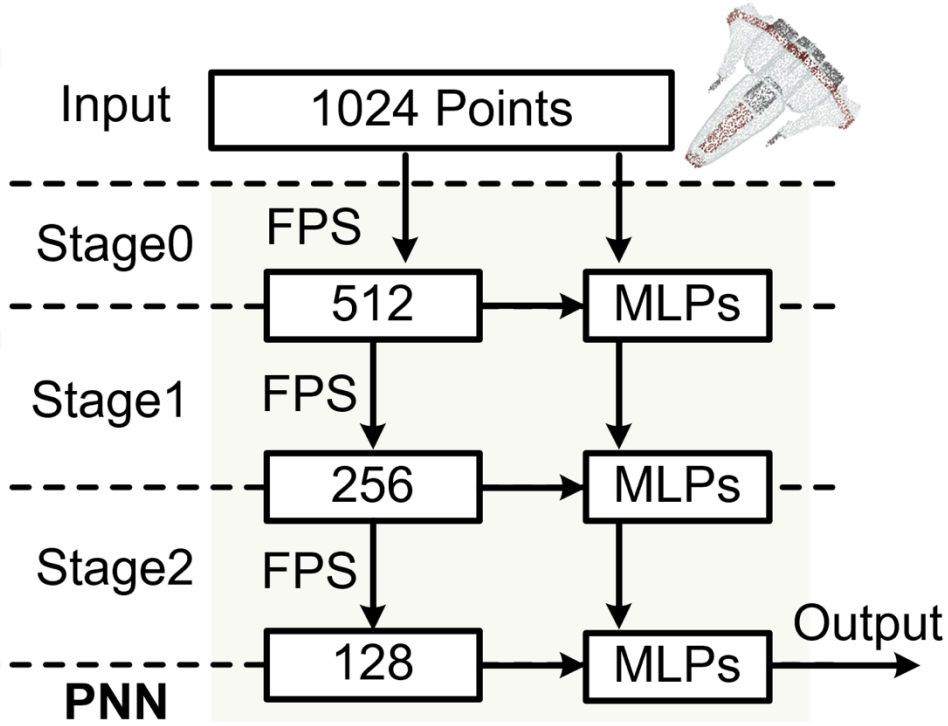
Poor Scaling of PNNs



Point cloud size: 1K (early benchmarks) → ~200K (Semantic Segment) → 300K in real LiDAR

FPS accounts for >90% of total inference time at large scale

Why is FPS slow?

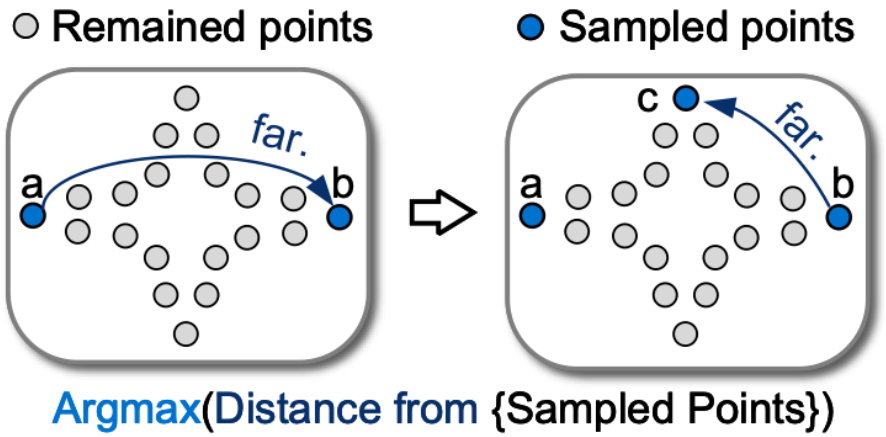


Multi-Layer FPS in PNNs

Select representative points.

Farthest Point Sample (FPS):

- Select the farthest unsampled point as next sample
- Input N points, sample out M points.
- *O(M)* iteration
- *O(N)* distance computation per iteration
- **Iterative Greedy Selection**



SOTA Optimization for FPS

CUDA-Level:

- [NeurIPS'22] FPS-CUDA: Intra-Iteration parallelism
- [TCAD'23] QuickFPS: KD-Tree Partition

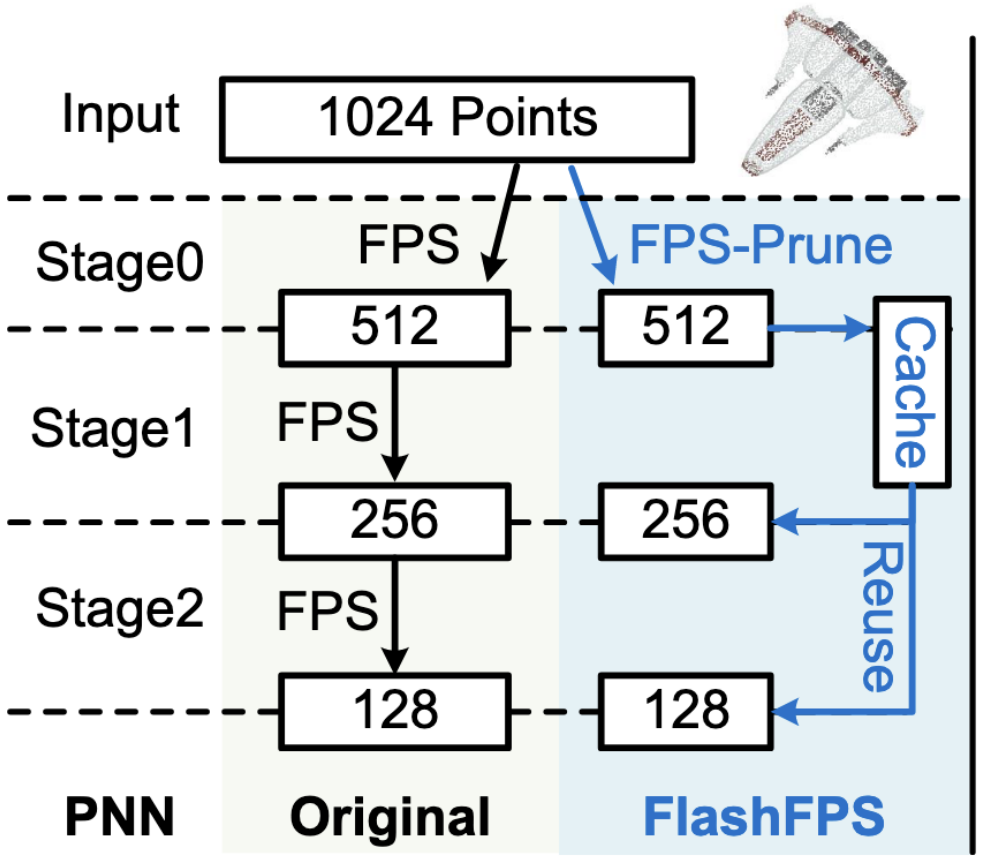
Accelerator-Level:

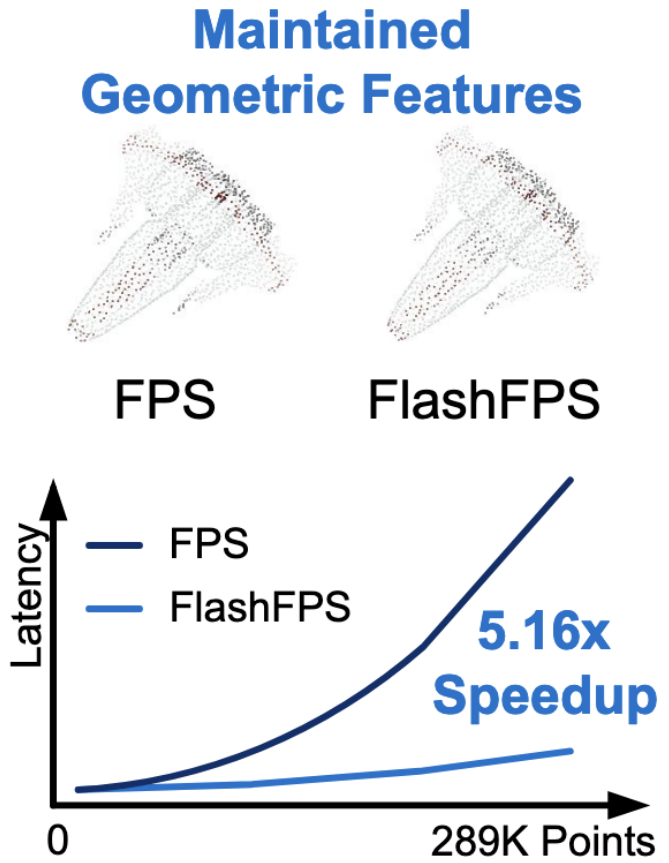
- [MICRO'21] PointAcc: Intra-Iteration parallelism
- [HPCA'26] FractalCloud: Fractal Partition, Inter-Block parallelism

Overlook the intrinsic redundancy in FPS

Qian et al., PointNeXt. NeurIPS, 2022.
Han et al., QuickFPS, TCAD, 2023.
Fu et al., FractalCloud, HPCA, 2026.
Lin et al., PointAcc, MICRO, 2021.

FlashFPS: FPS-Prune and FPS-Cache





FPS-Prune: Optimize FPS

FPS-Cache: Optimize multi-FPS layers

Plug-and-Play

Hardware-agnostic

PointNeXt-L@S3DIS — FPS-CUDA vs FlashFPS

Baseline: FPS-CUDA

Start running

Progress: 0/68

test_oa : --

test_macc : --

test_miou : --

S3DIS Test Progress

End-to-end speedup

5.28x

Real Latency on TITAN GPU

Baseline : -- s

FlashFPS : -- s

Status

Baseline : not started

FlashFPS : not started

Ours: FlashFPS

Start running

Progress: 0/68

test_oa : --

test_macc : --

test_miou : --

Video time scaled from per-test latency. Codes: https://github.com/Yuzhe-Fu/FlashFPS

63 THE CHIPS TO SYSTEMS CONFERENCE

SPONSORED BY: CEDA IEEE Council on Electronic Design Automation, SIGDA ACM

FlashFPS: Prune and Cache Inspired from three redundancies

Three Redundancies

Within a single FPS:

Redundancy1: Full-Cloud Computation

Redundancy2: Diminishing Returns in Late Iterations

Across hierarchical layers:

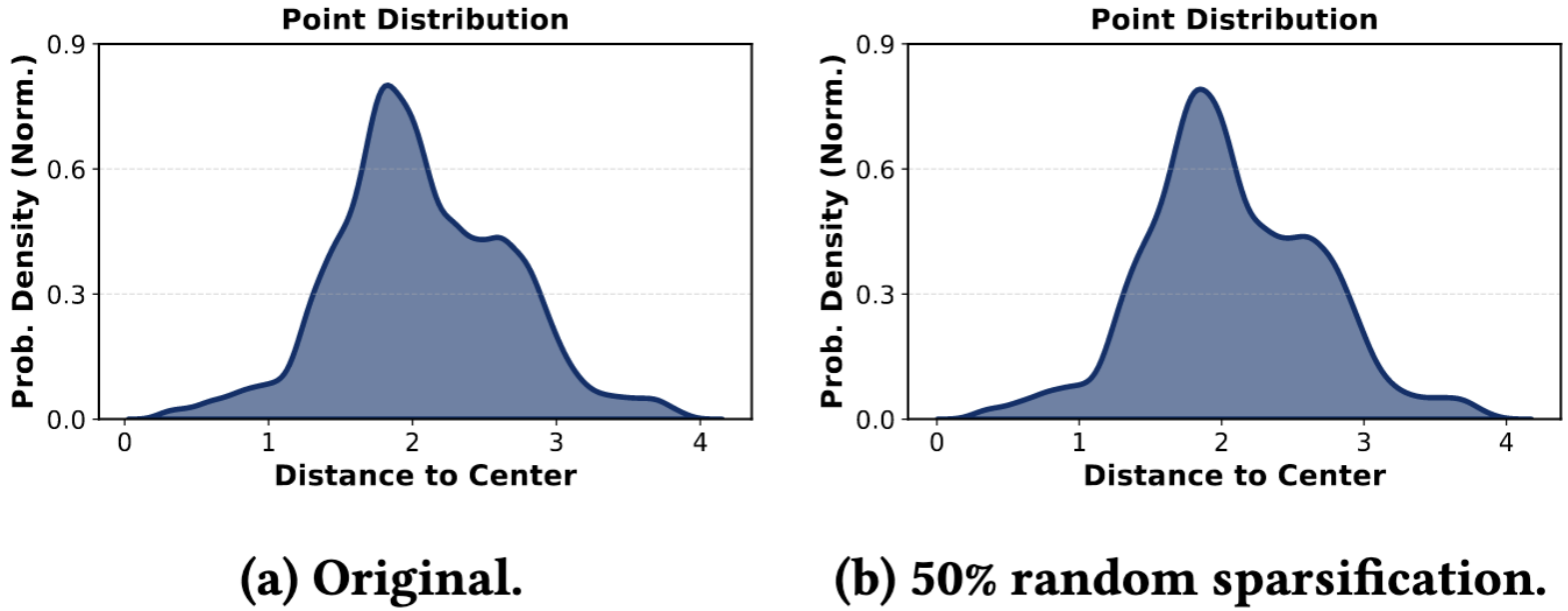
Redundancy3: Inter-layer Recomputation

FPS-Prune: Candidate Pruning

Redundancy 1: Full-Cloud Computation

FPS selects the point farthest from the sampled set.

Many clustered points are almost never selected.



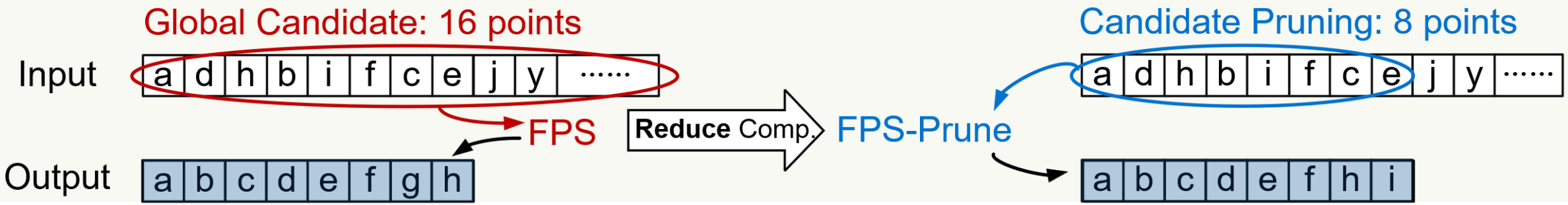
(a) Original. (b) 50% random sparsification.

Similar FPS Point Distribution after 50% input point Pruning

FPS-Prune: Candidate Pruning

Candidate Prune: *p* = pruning ratio

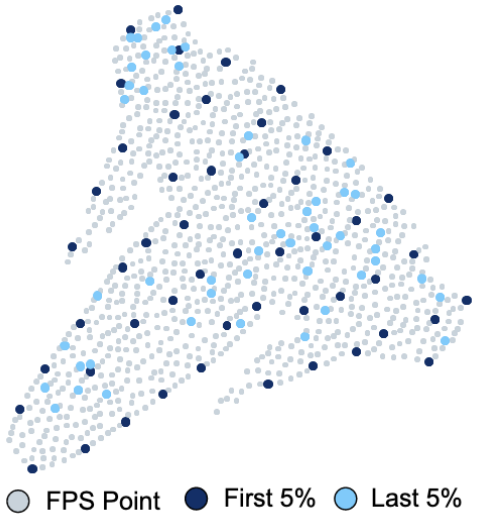
Pruning *p*% input points, then apply FPS



FPS-Prune: Iteration Pruning

Redundancy 2: Diminishing Returns in Late Iterations

- Early FPS iterations capture global structure
- Late iterations refine local details, which are diminished at deeper PNN layers

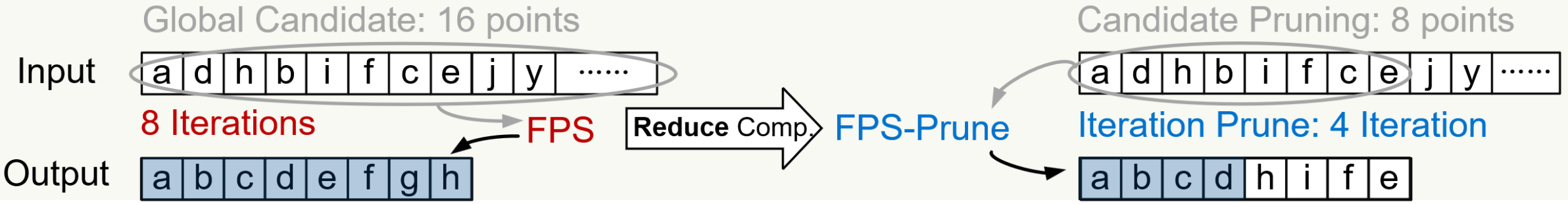


Late FPS iterations contribute less.

FPS-Prune: Iteration Pruning

Iteration Pruning:

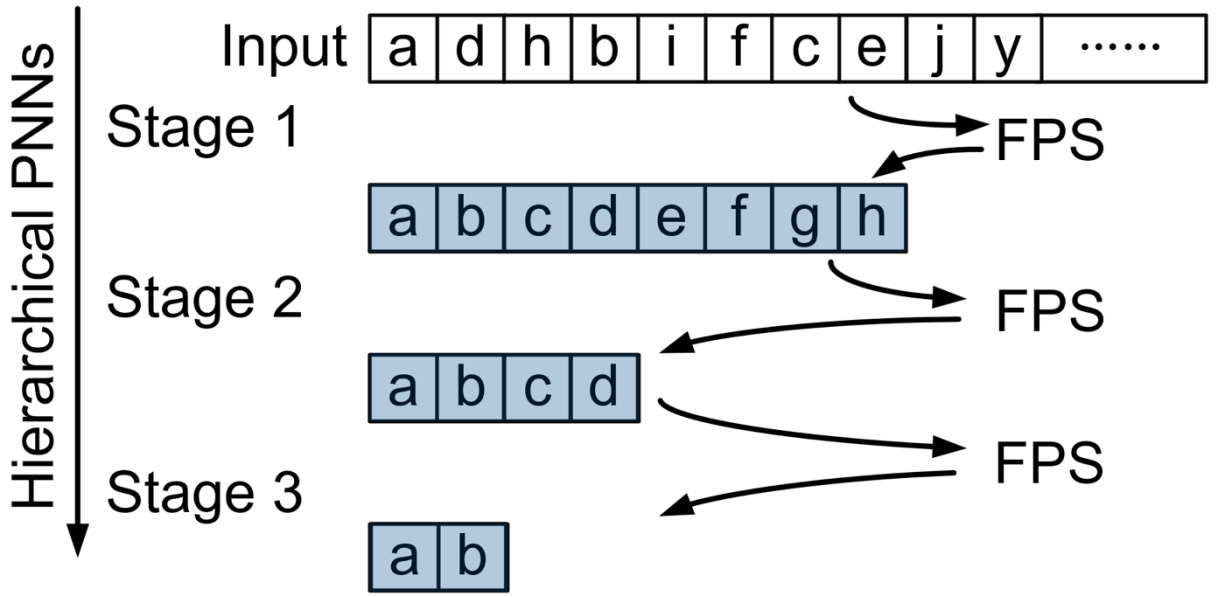
Pruning *p*% FPS iteration, with remaining filled with random sample.



FPS-Cache: Cache and reuse

Redundancy 3: Hierarchical

- Modern PNNs fix the FPS inital point.
- FPS fully deterministic for a given input

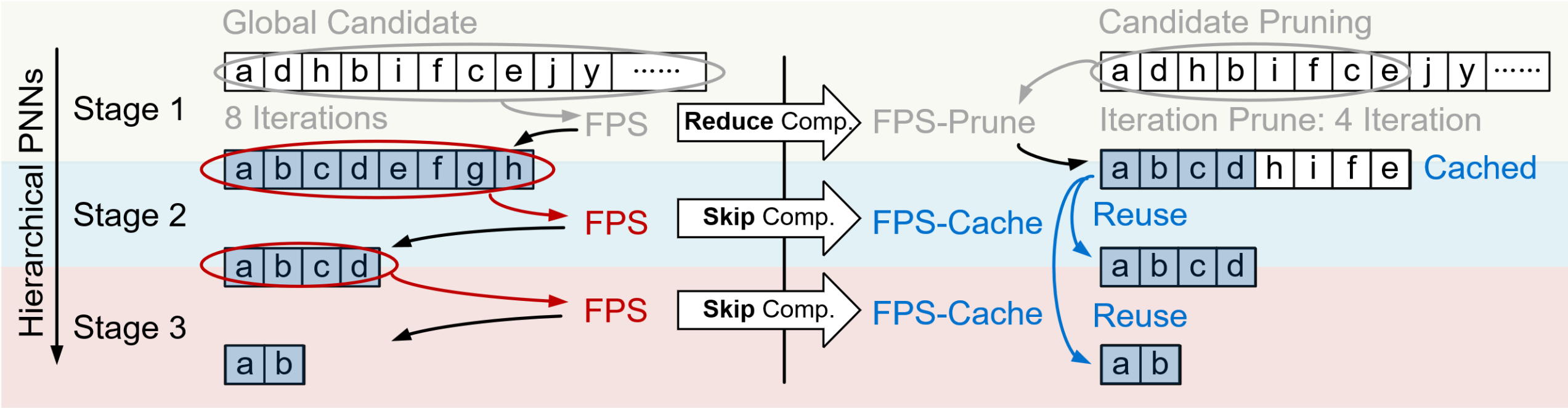


Prefix Theory: with a fixed seed, the output of deeper FPS layer is always a prefix of the first-layer output.

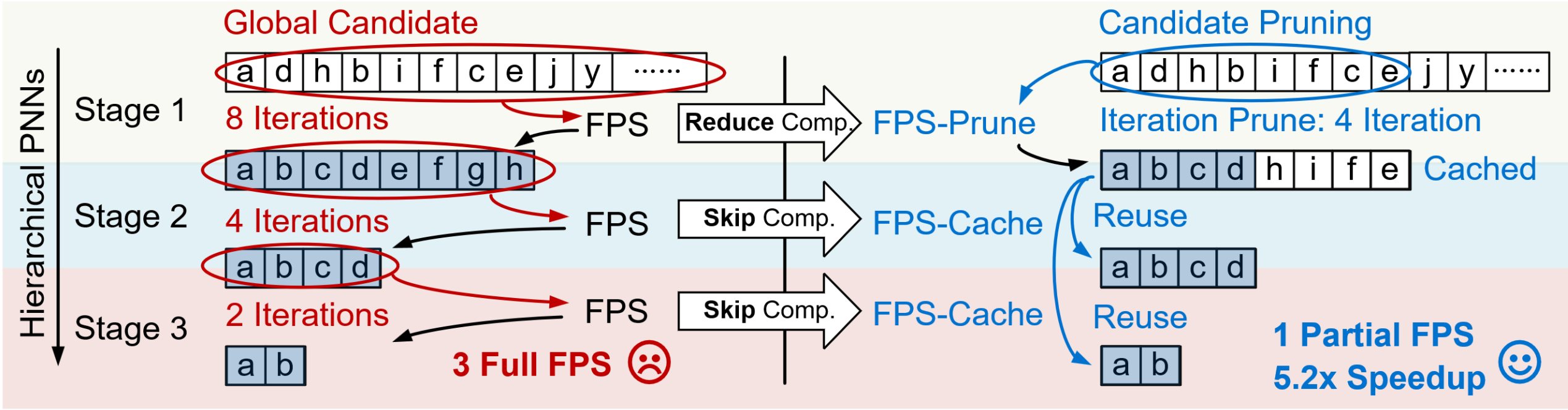
FPS-Cache: Cache and Reuse

FPS-Cache:

cache the First FPS output, and reuse for all deeper FPS layers.



Pipeline of FlashFPS



Original: 3 Full FPS.

FlashFPS: 1 Partial FPS

Experimental Results

Speedup vs. Network Accuracy

Two SOTA PNNs:

- PointNeXt-L
- PointVector-L

Two Benchmarks:

- S3DIS
- ScanNet

Under 75% pruning rate,

Average **5.16x** Speedup

Negligible accuracy loss

Workload	Method	OA (%)	mAcc (%)	mIoU (%)	Speedup
S3DIS PointNeXt-L	FPS-CUDA	90.09	75.76	69.33	1
	FlashFPS-25%	90.05	75.7	69.21	**1.76**
	FlashFPS-50%	90.00	75.6	69.13	**2.72**
	FlashFPS-75%	89.92	75.48	69.01	**5.28**
S3DIS PointVector-L	FPS-CUDA	90.97	76.75	70.83	1
	FlashFPS-25%	90.93	76.74	70.79	**1.5**
	FlashFPS-50%	90.84	76.56	70.55	**2.76**
	FlashFPS-75%	90.7	76.32	70.27	**4.98**
ScanNet PointNeXt-L	FPS-CUDA	89.07	78.66	70.54	1
	FlashFPS-25%	89.07	78.62	70.45	**1.7**
	FlashFPS-50%	89.01	78.41	70.17	**2.67**
	FlashFPS-75%	88.87	78.1	69.79	**5.27**
ScanNet PointVector-L	FPS-CUDA	89.1	77.95	70.03	1
	FlashFPS-25%	89.12	77.97	70.07	**1.5**
	FlashFPS-50%	89.05	77.92	69.97	**2.8**
	FlashFPS-75%	88.78	77.24	69.27	**5.13**

Speedup with CUDA-based Works

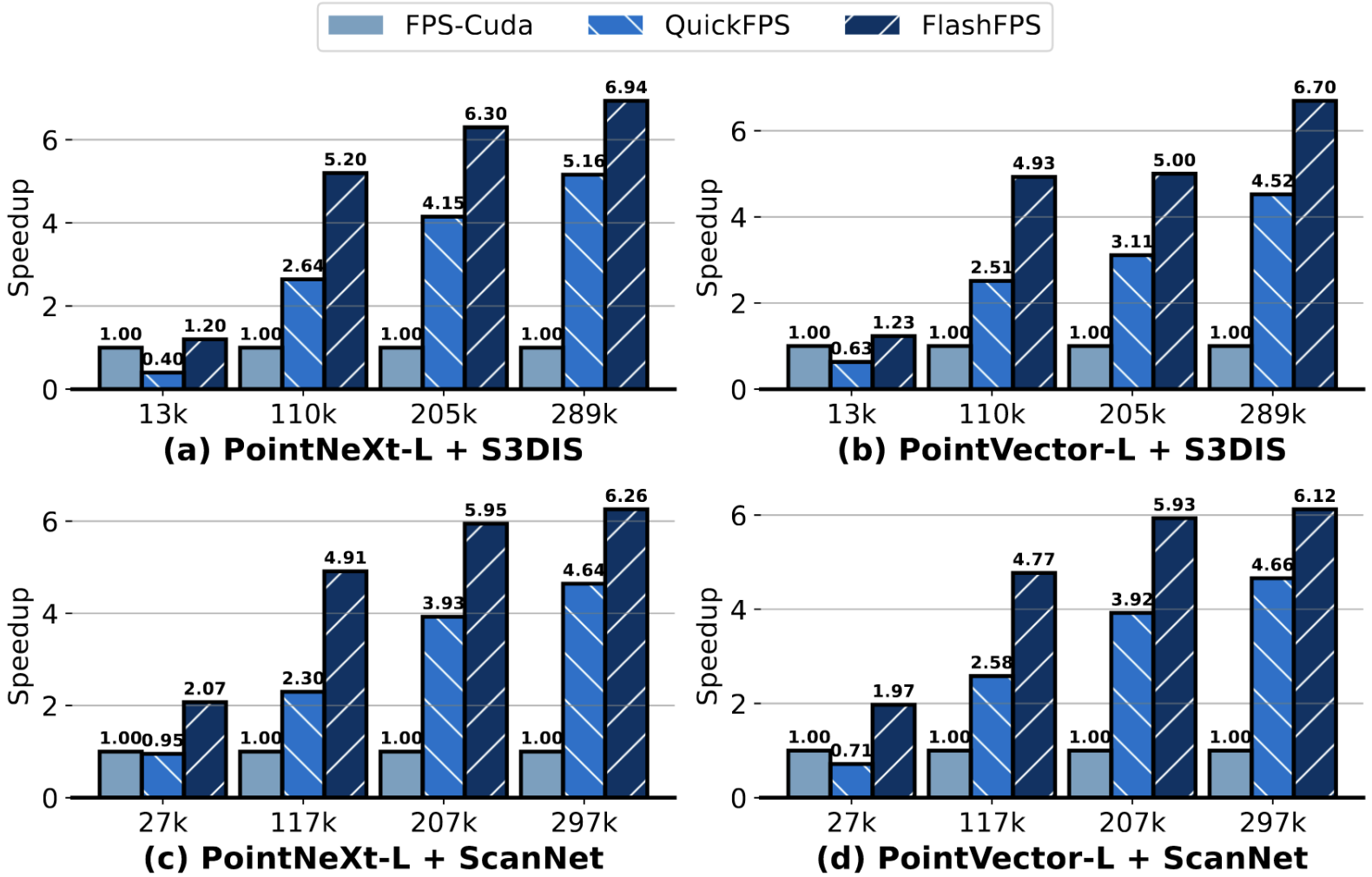
Two CUDA baselines:

- FPS-CUDA [NeurIPS'22]
- QuickFPS [TCAD'23]

5.16x than FPS-CUDA

1.96x than QuickFPS

Average **3.56x** Speedup



Speedup on Different Hardware Platforms

Three different platforms:

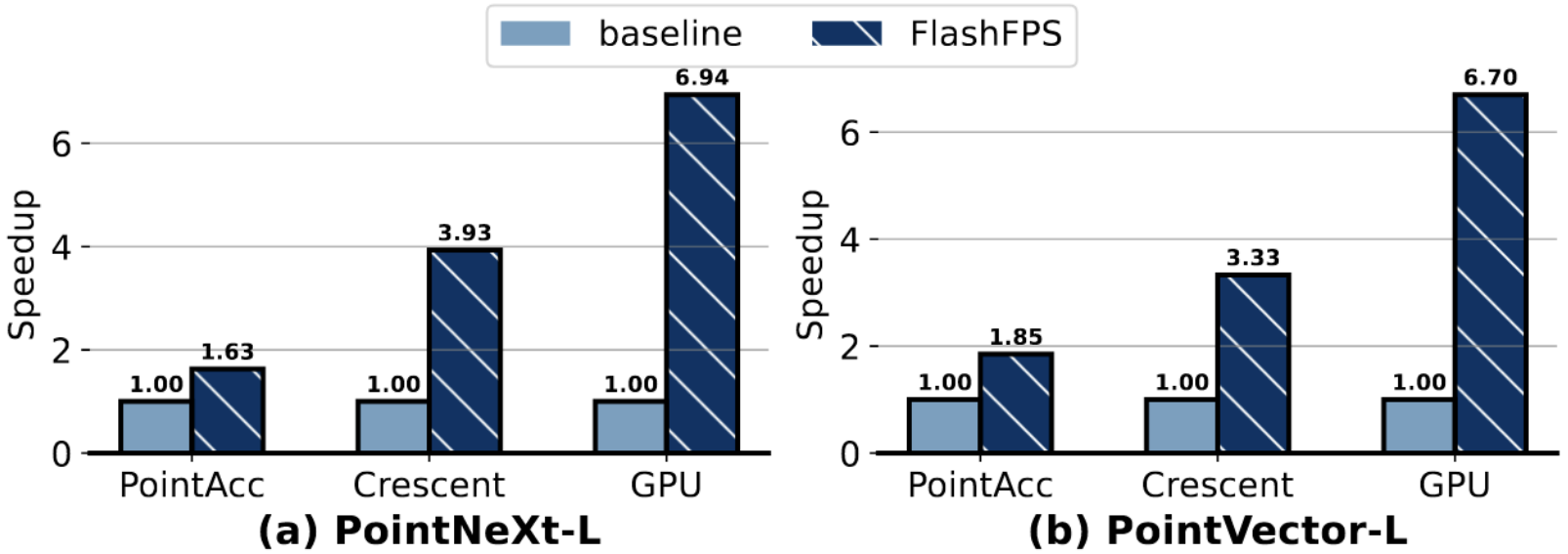
- PointAcc [MICRO'21]
- Crescent [ISCA'22]
- GPU (NV TITAN)

On 289K point inference, additional speedup:

1.74x on PointAcc

3.63x on Crescent

6.81x on TITAN GPU



Ablation Study of FlashFPS Components

Candidate Prune (CP)

Iteration Prune (IP)

FPS Cache (FC)

PointNeXt-L@S3DIS

CP alone: 2.44x

+ IP = 3.63x

+ FC = 5.28x

w/ CP	w/ IP	w/ FC	mIoU (%)	Speedup
FPS-CUDA (baseline)			69.33	1.00
✓			69.01	2.44
	✓		69.13	2.31
		✓	69.33	1.10
✓	✓		69.01	3.63
✓		✓	69.01	2.87
	✓	✓	69.13	2.77
✓	✓	✓	**69.01**	**5.28**

All components are additive and complementary

Conclusion

Conclusion

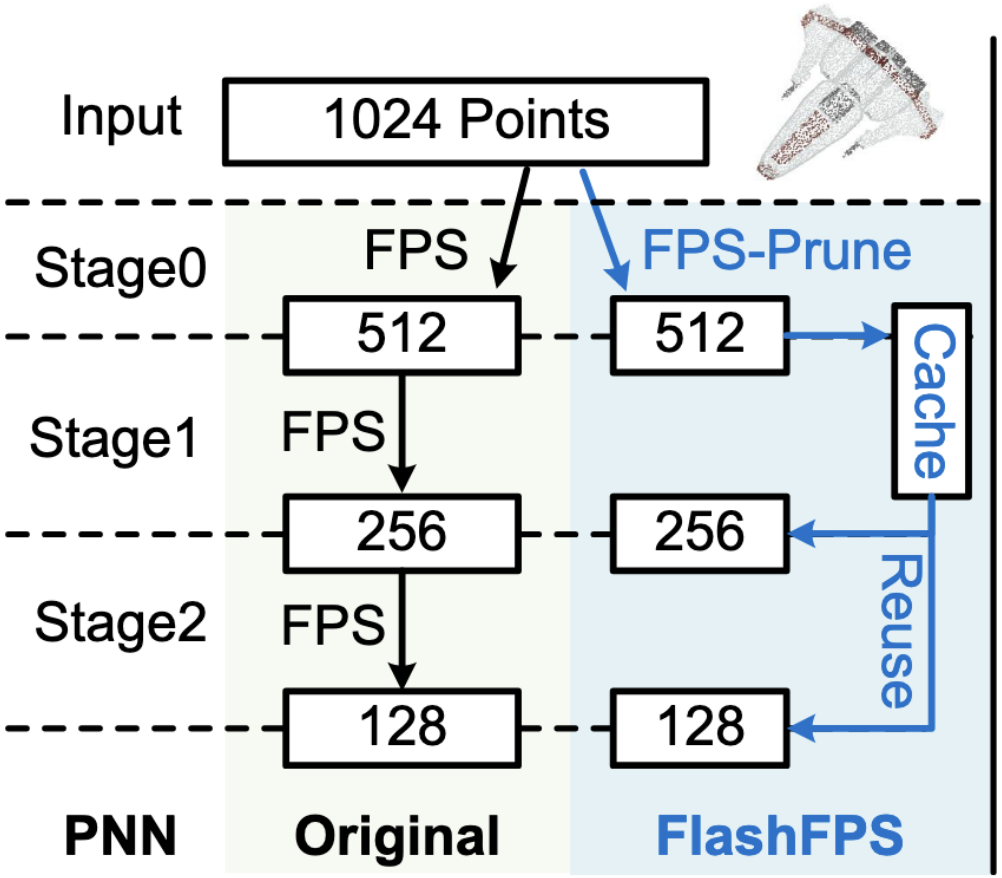
FlashFPS

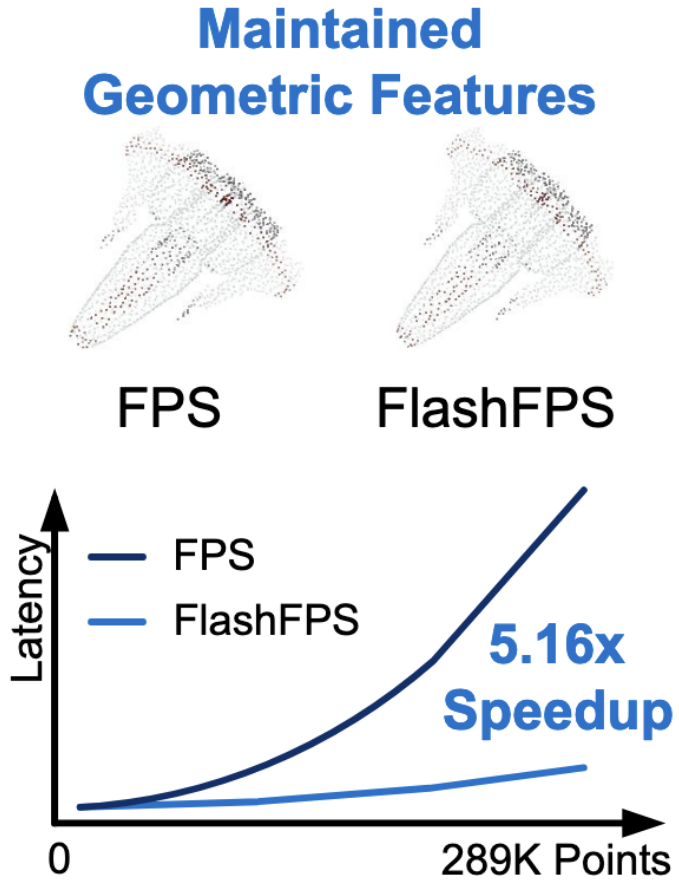
- FPS-Prune
- FPS-Cache

5.16x Speedup

Plug-and-Play

Hardware-agnostic





Acknowledgements

Thanks for Listening.

FlashFPS
DAC 2026

Codes are open-sourced at:

https://github.com/Yuzhe-Fu/FlashFPS

Author: Yuzhe Fu,
Hancheng Ye, Cong Guo,
Junyao Zhang, Qinsi Wang,
Yueqian Lin, Changchun Zhou,
Hai “Helen” Li and Yiran Chen

Duke UNIVERSITY

Center of Computational Evolutionary Intelligence (CEI)