KU-CSL
COMPUTER SYSTEMS LAB

VALO: A Versatile Anytime Framework for LiDAR based Object Detection Deep Neural Networks

Ahmet Soyyigit[1], Shuochao Yao[2], Heechul Yun[3]

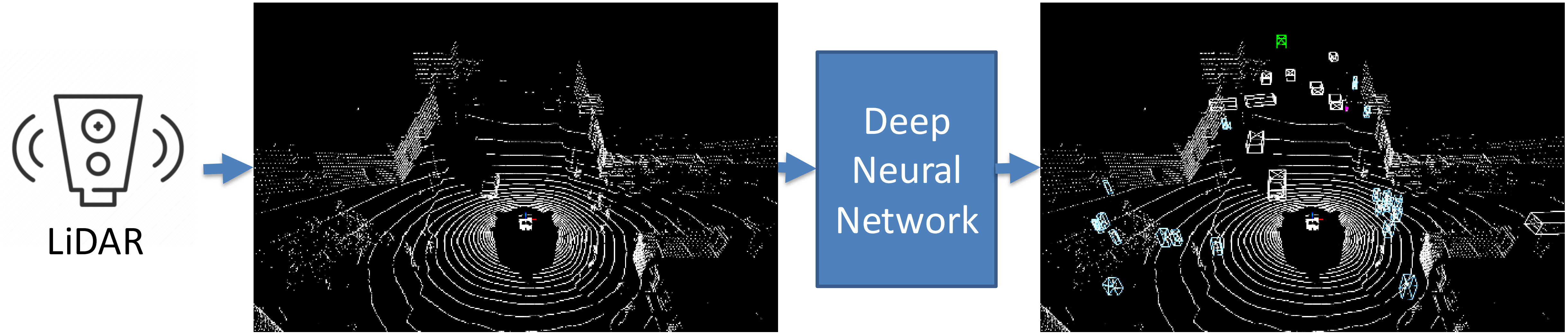
[1,3] University of Kansas, Lawrence, KS

[2] George Mason University, Fairfax, VA

KU THE UNIVERSITY OF KANSAS

3D Object Detection

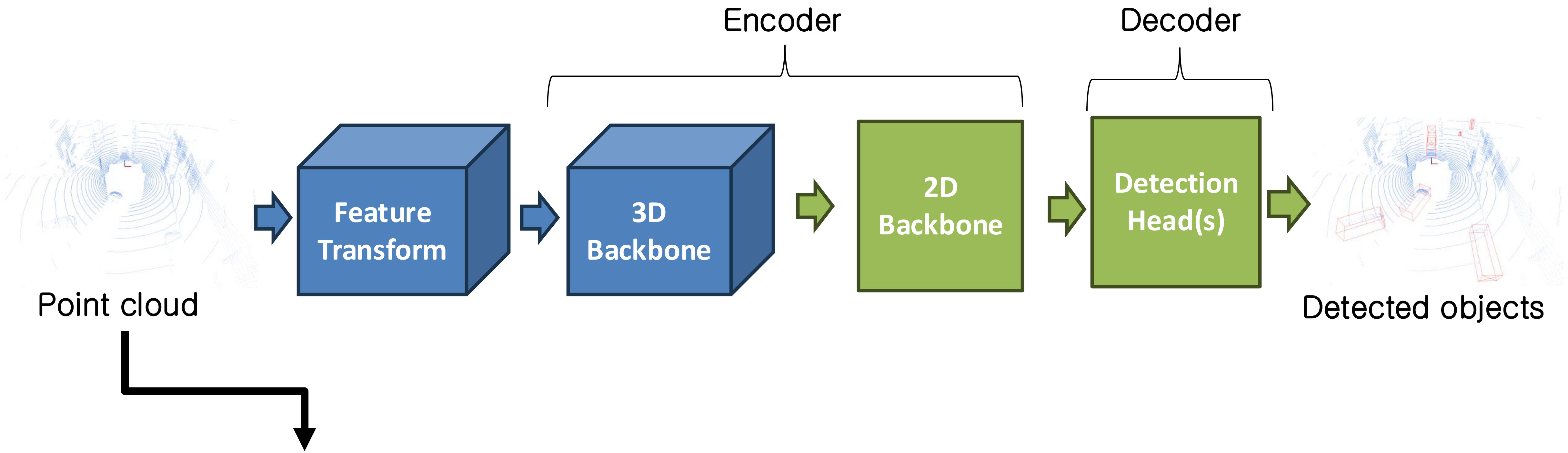
- Cameras, Radars, LiDARs …
- Deep Neural Networks (DNN) are state-of-the-art (SOTA) for LiDAR



Point cloud Detected Objects

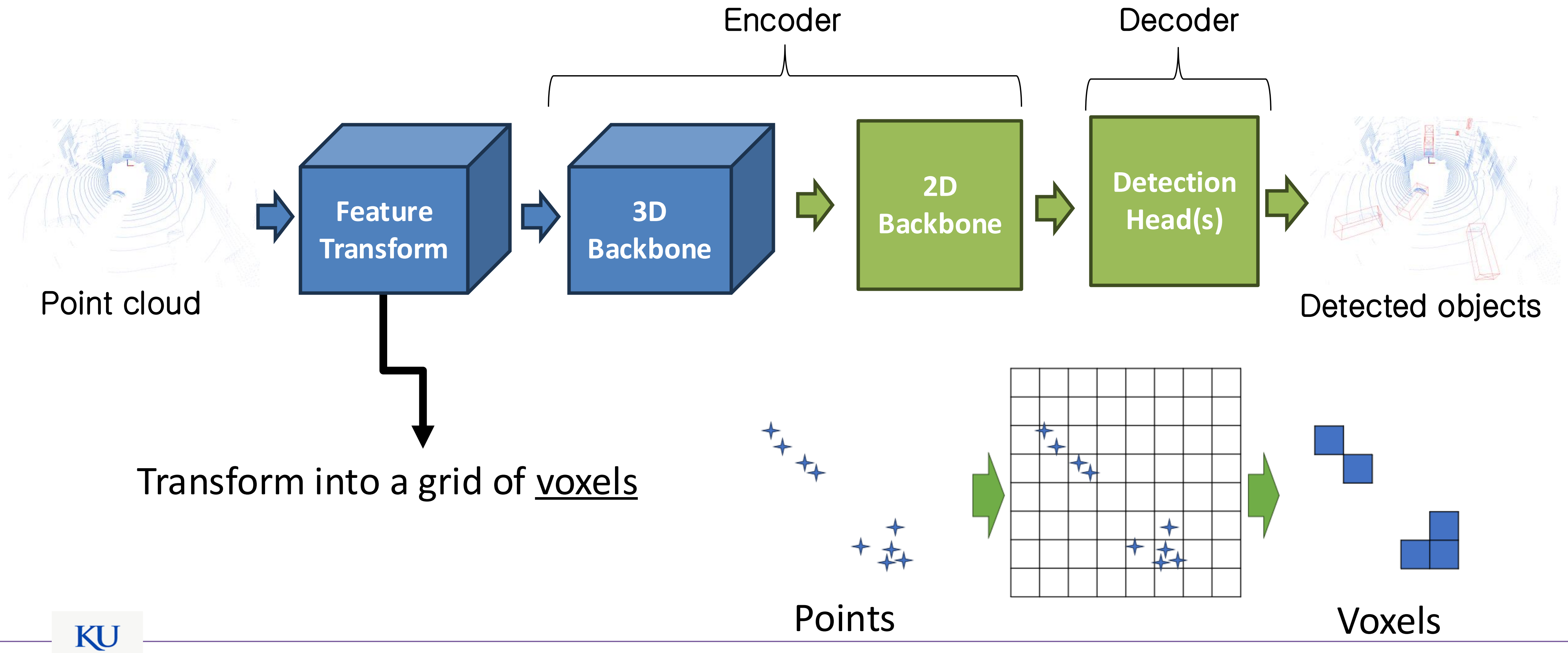
Image credits (leftmost): https://www.rdvsystems.com/lidar-icon-4/

LiDAR Object Detection DNNs

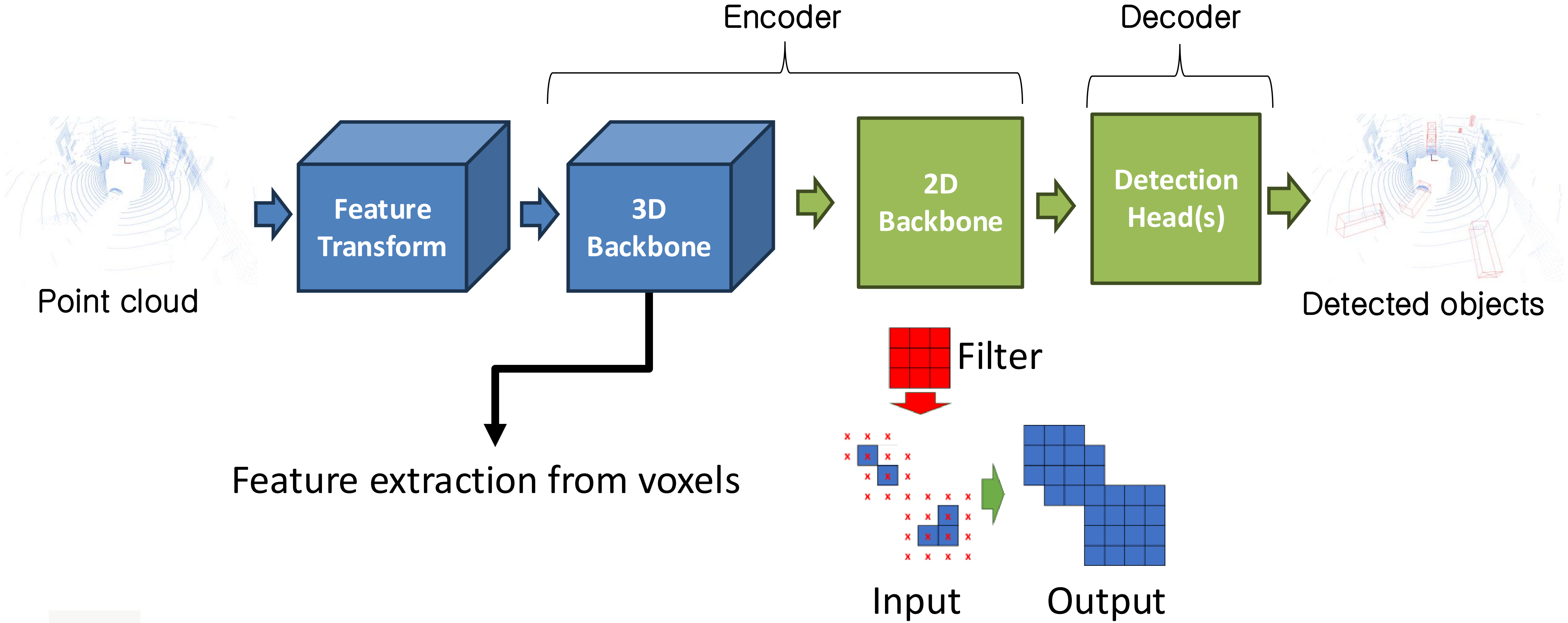


Array of points (x, y, z, intensity)

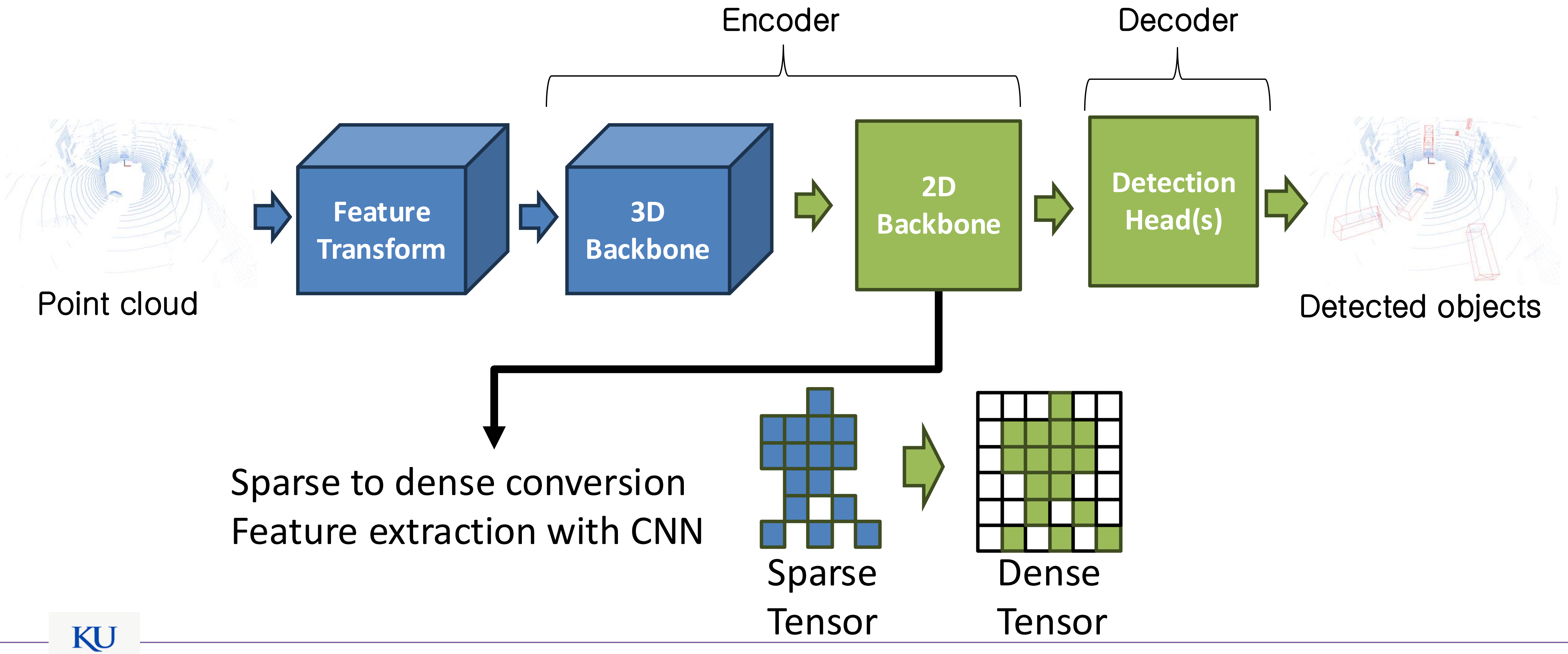
LiDAR Object Detection DNNs



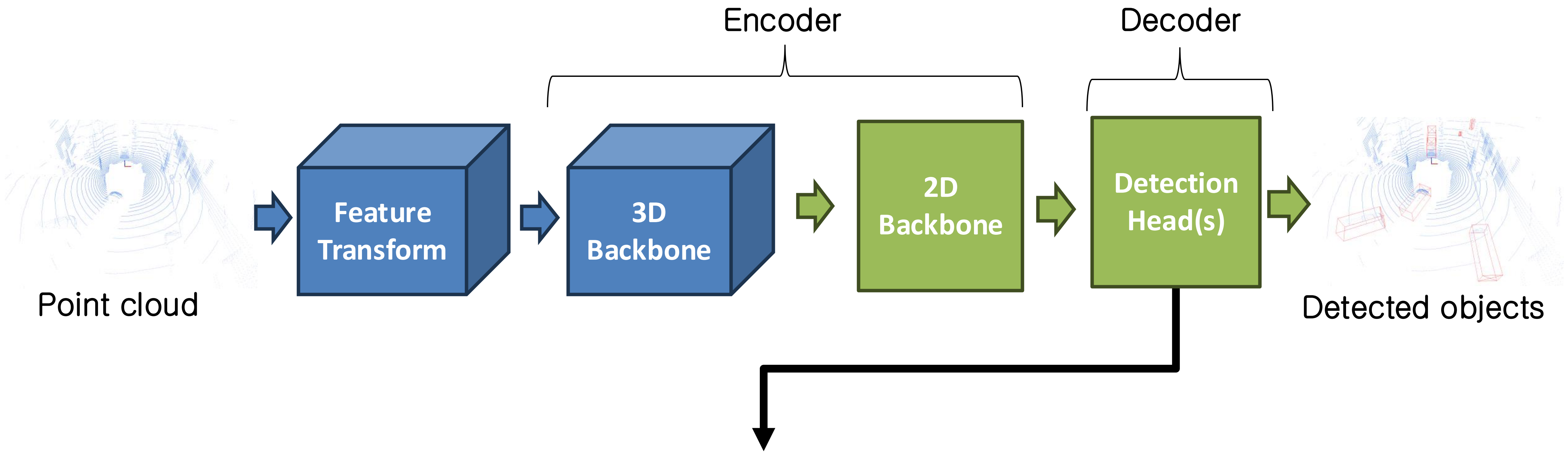
LiDAR Object Detection DNNs



LiDAR Object Detection DNNs

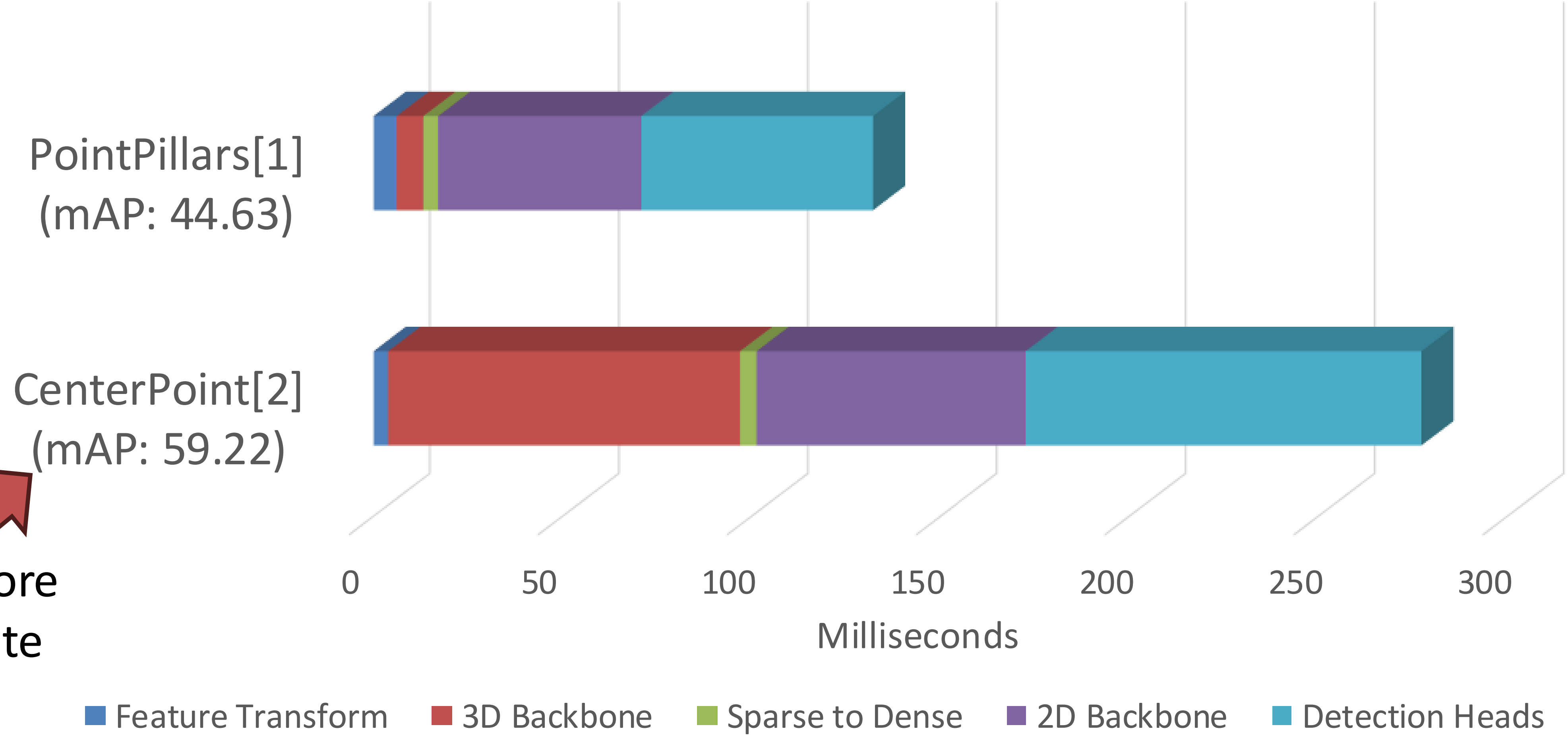


LiDAR Object Detection DNNs



Decode features with CNN
Do post-processing

Mean Latency on Jetson AGX Xavier



34% more accurate

[1] A. H. Lang et. al., "PointPillars: Fast Encoders for Object Detection From Point Clouds," *2019 IEEE/CVF CVPR*
[2] T. Yin et. al., "Center-based 3D Object Detection and Tracking," *2021 IEEE/CVF CVPR*

Favor Latency or Accuracy?

- Objects move while detection happens.
  - Predictions can be misaligned.
- Lower latency is favored in high misalignment scenario.

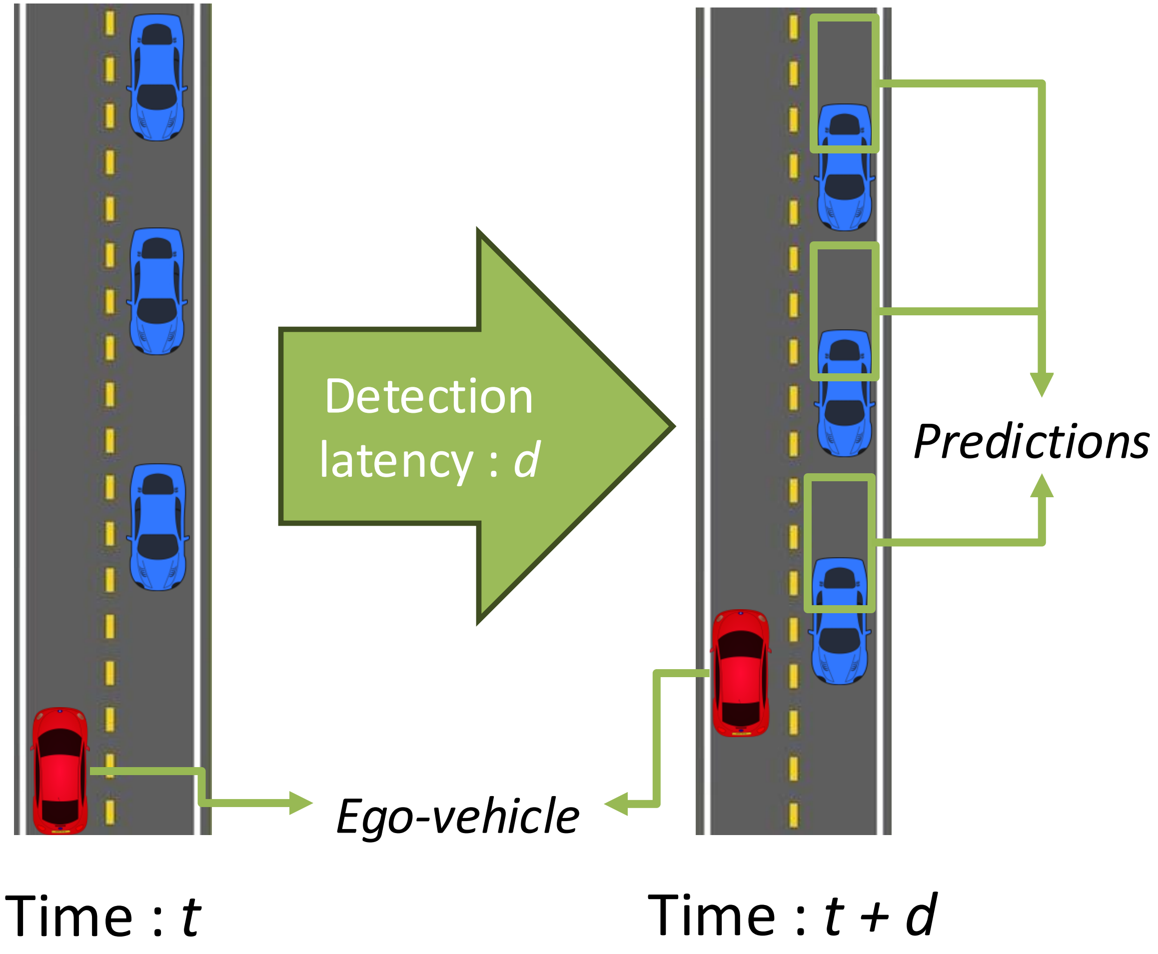
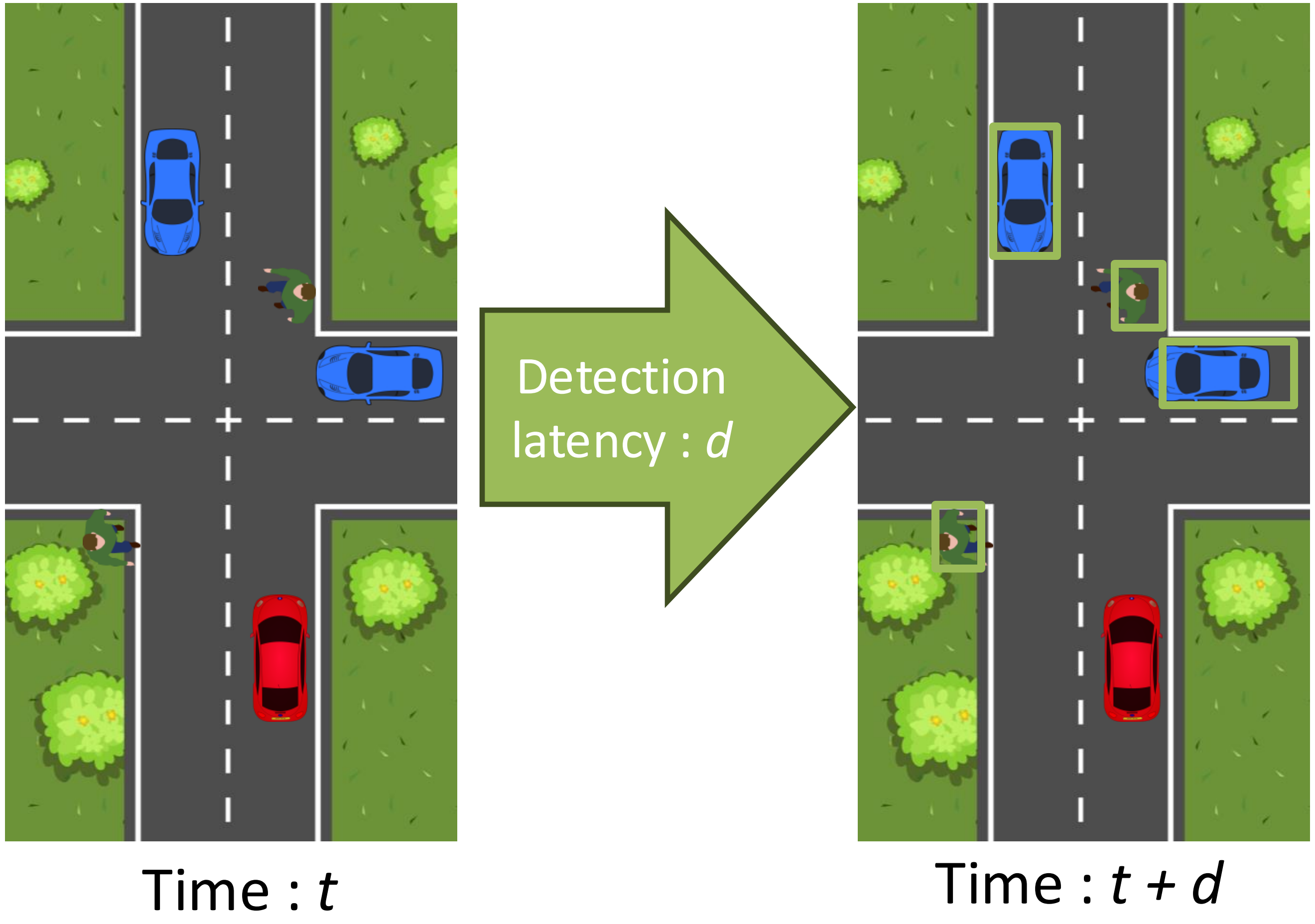


Image credits: https://tr.pinterest.com/pin/633811347586237182/

Favor Latency or Accuracy?

- Low misalignment scenario.
  - Higher latency is tolerable.
  - Higher accuracy is favored.



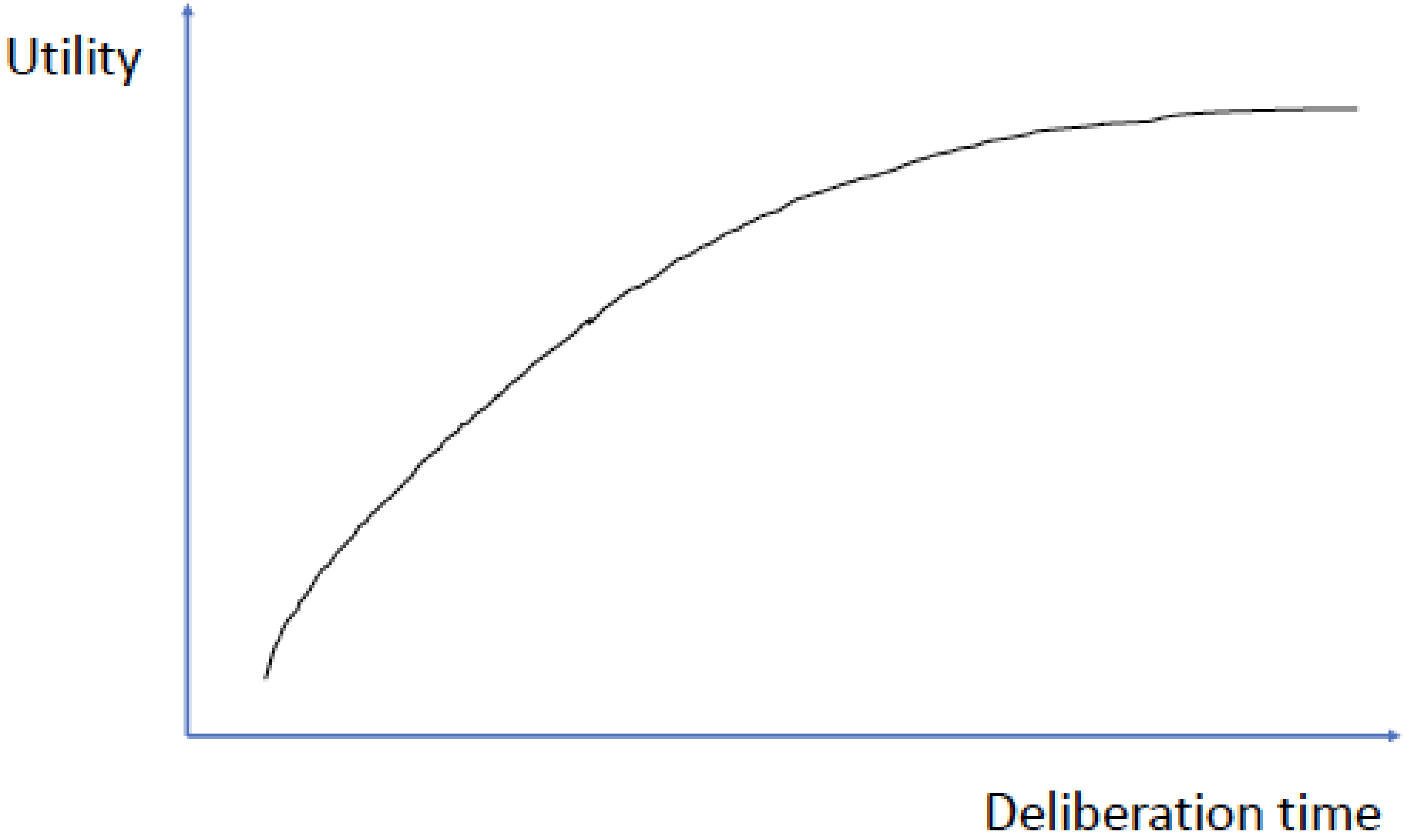
Crossing image credits: brgfx on Freepik

Favor Latency or Accuracy?

- Best DNN model is environment dependent.
  - One model does not fit all.

Can we dynamically reconfigure our object detector to make environment-dependent latency and accuracy trade-offs?

Anytime Algorithms



[3] Boddy, M., and Dean T. L. "Solving Time-Dependent Planning Problems." *In Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, 1989.

LiDAR Object Detection DNN as an Anytime Algorithm

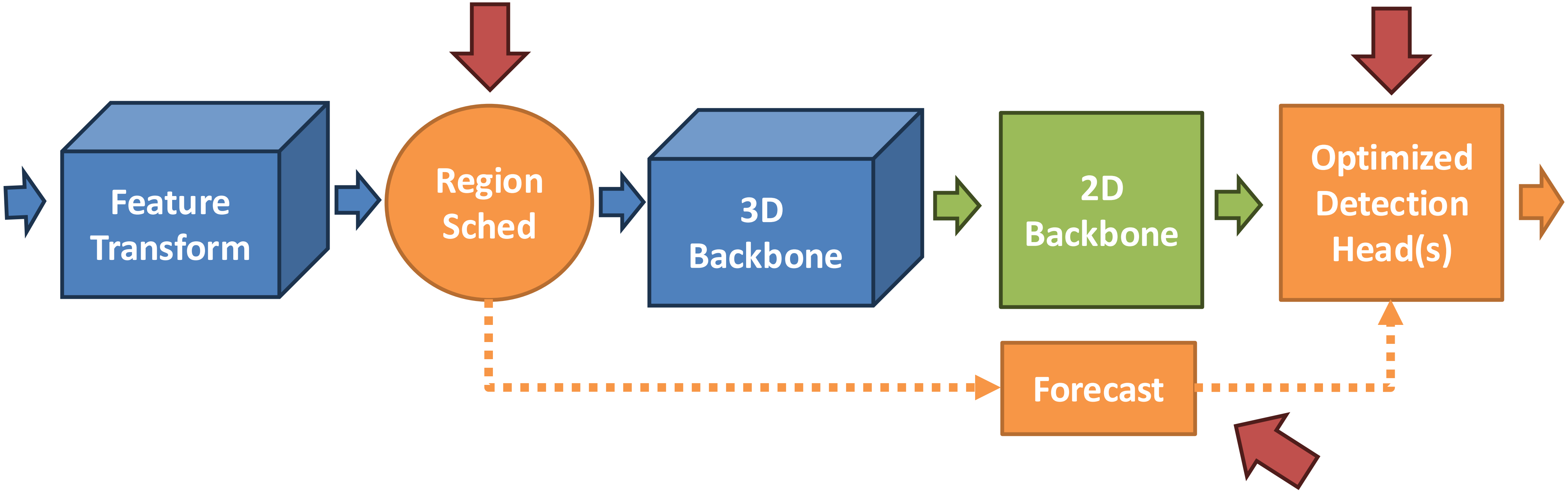
- DNN’s are not anytime by default.
- Possible Solution: Dynamically switching multiple DNNs
  - Memory overhead
    - Not suitable for edge
  - Train/fine-tune multiple DNNs
    - Might not have access to training pipeline

Goal

- Develop a *versatile* framework that can transform *any* single LiDAR object detection DNN into an *anytime-capable* one
  - So, it can trade-off latency and accuracy dynamically at runtime.
- Our previous work[4] has limitations
  - Modifies DNN architecture, enforces training
  - No trade-offs on 3D backbone
- We need a versatile solution
  - Minimal dependency on DNN architecture
    - Does not enforce training
    - Broadly applicable
  - Considers all important stages of the DNN

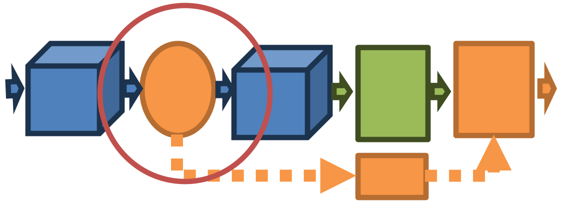
[4] A. Soyyigit et. al., "Anytime-Lidar: Deadline-aware 3D Object Detection," *2022 IEEE 28th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*

VALO: Versatile Anytime framework for LiDAR Object Detection

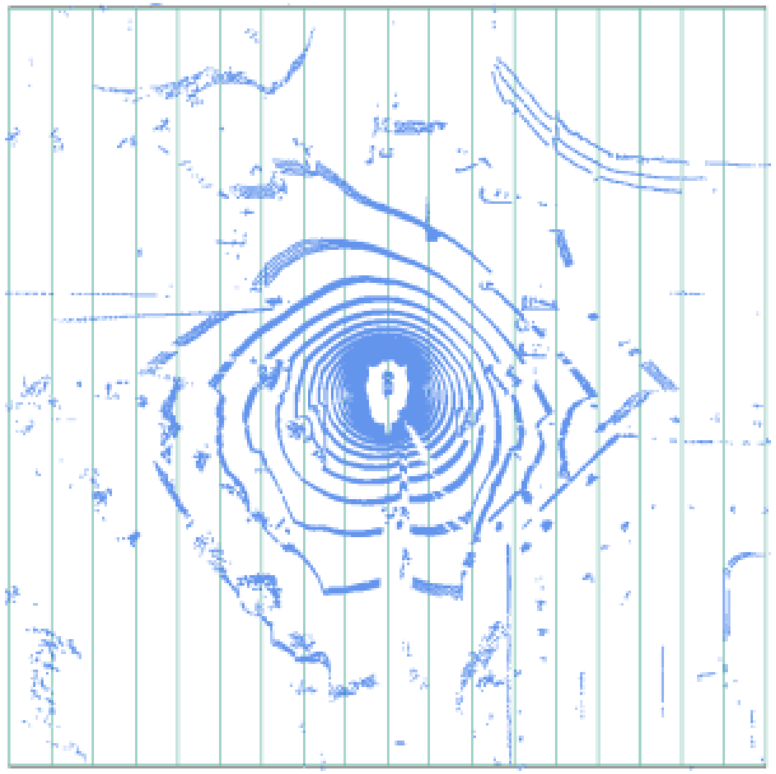


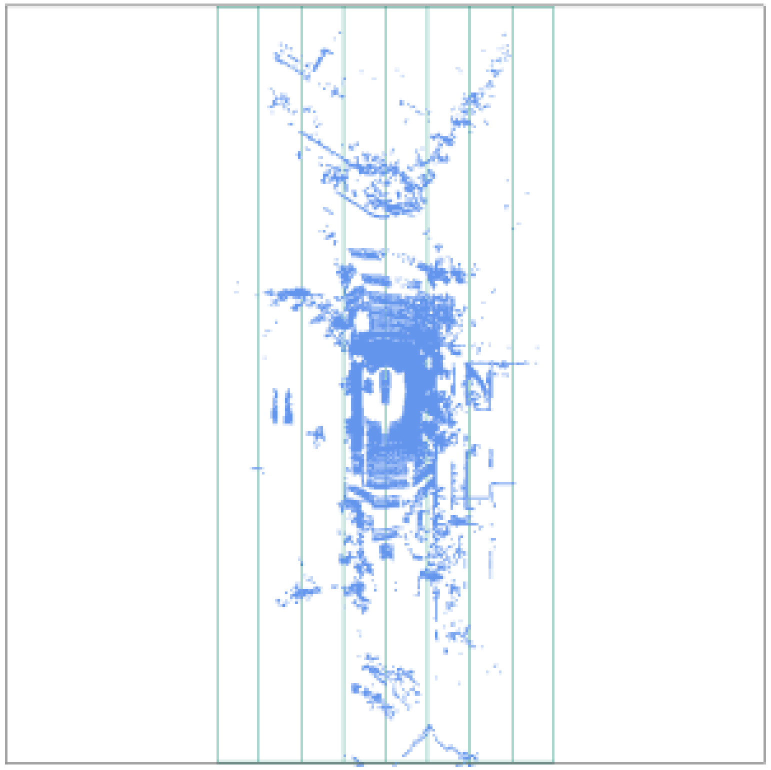
- Anytime computing with *scheduling of input data*.

Region Scheduling

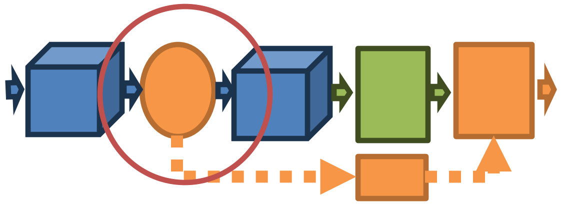


- Consider the input as a fixed number of vertical regions.
  - Skips empty regions by default.

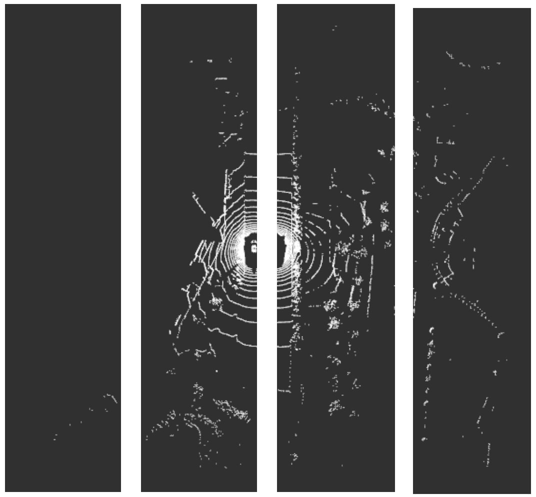




Region Scheduling



- Select max num of regions that can meet the deadline.
- Follow round-robin order.



1 2 3 4

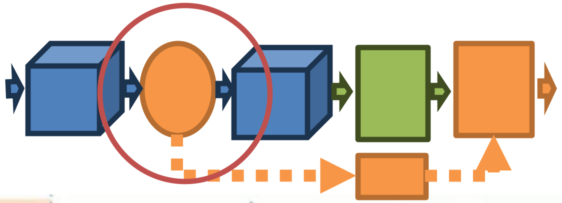
Last processed region

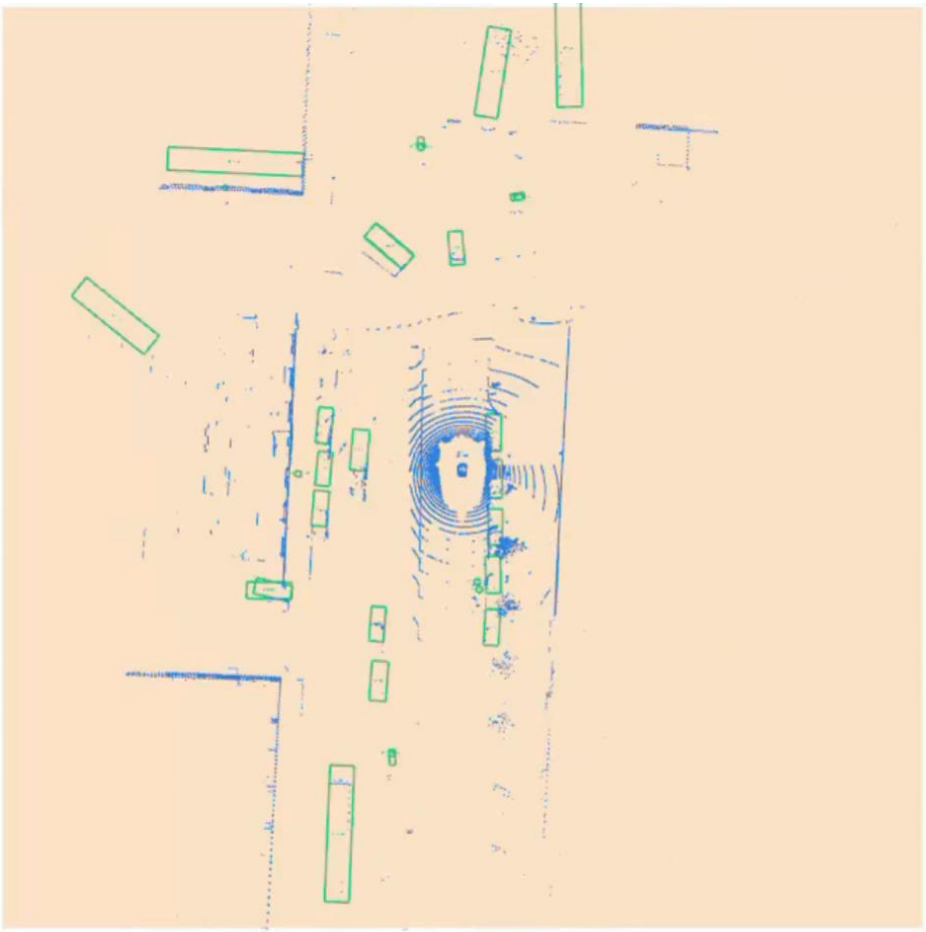
Regions To Select	Predicted WCET
3	80 ms
3, 4	115 ms
3, 4, 1	130 ms
3, 4, 1, 2	170 ms

Deadline: 140 ms

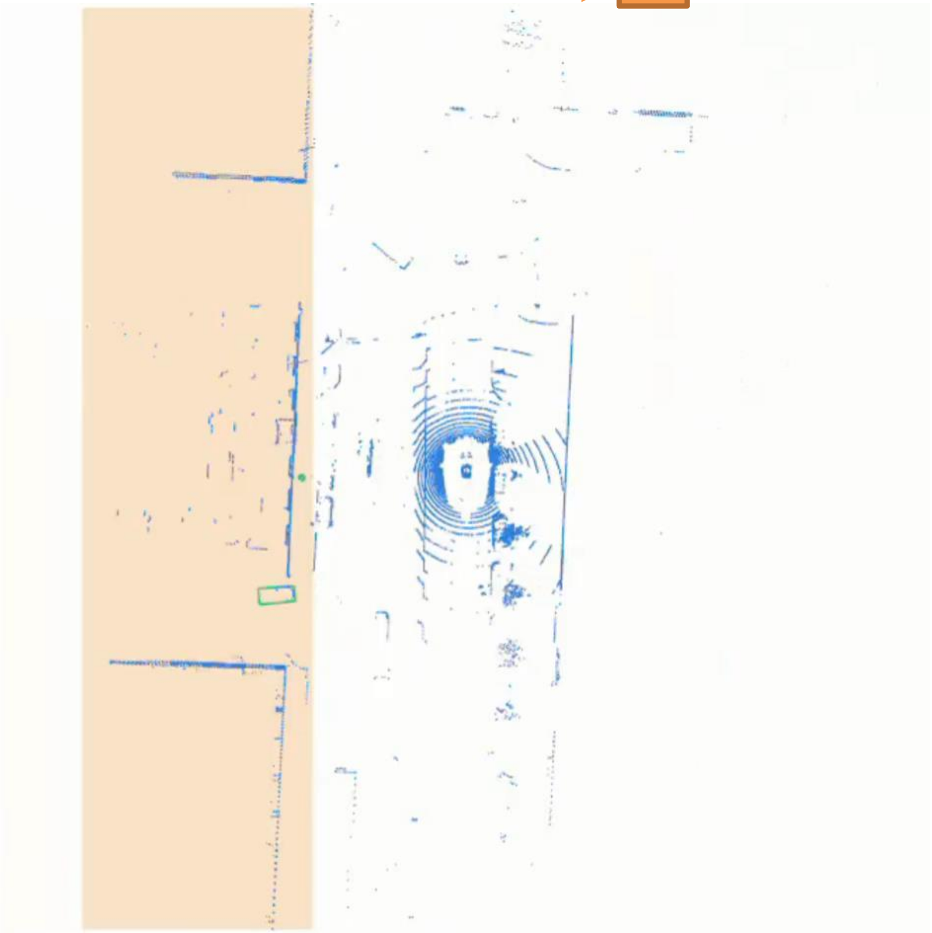
Selected regions: 3, 4, 1

Region Scheduling



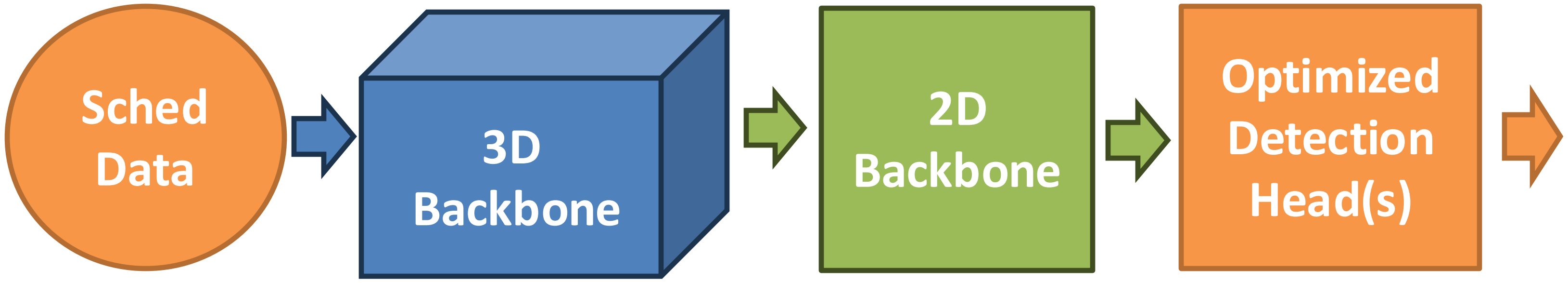


Baseline



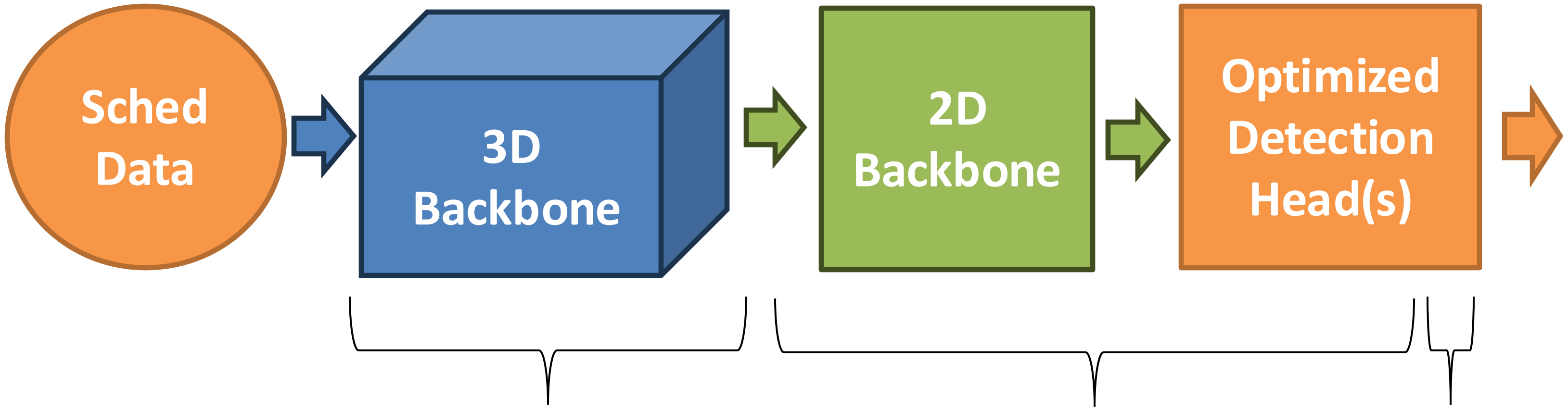
VALO

WCET Prediction For a Given Subset of Regions



$$WCET = E_s + E_D + E_R$$

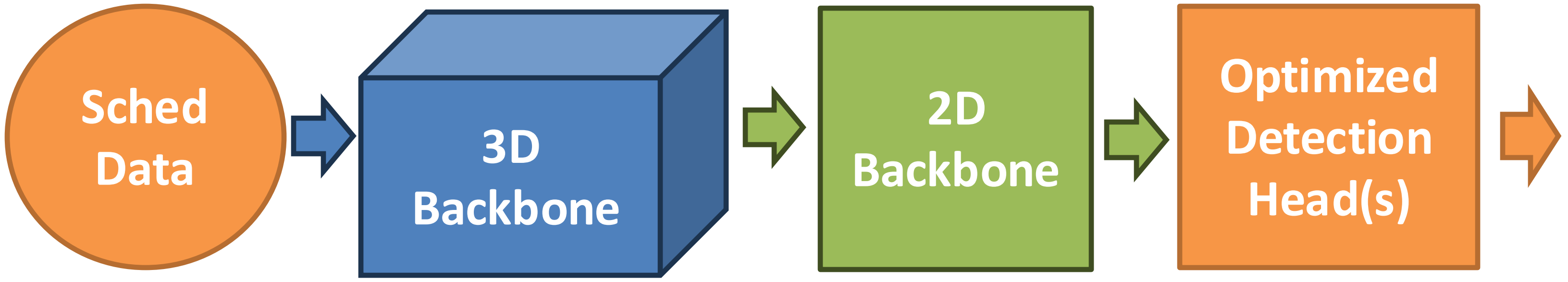
WCET Prediction For a Given Subset of Regions



$$WCET = E_s + E_D + E_R$$

- Post-processing time (e.g. Non-maximum suppression)
  - Can be considered constant

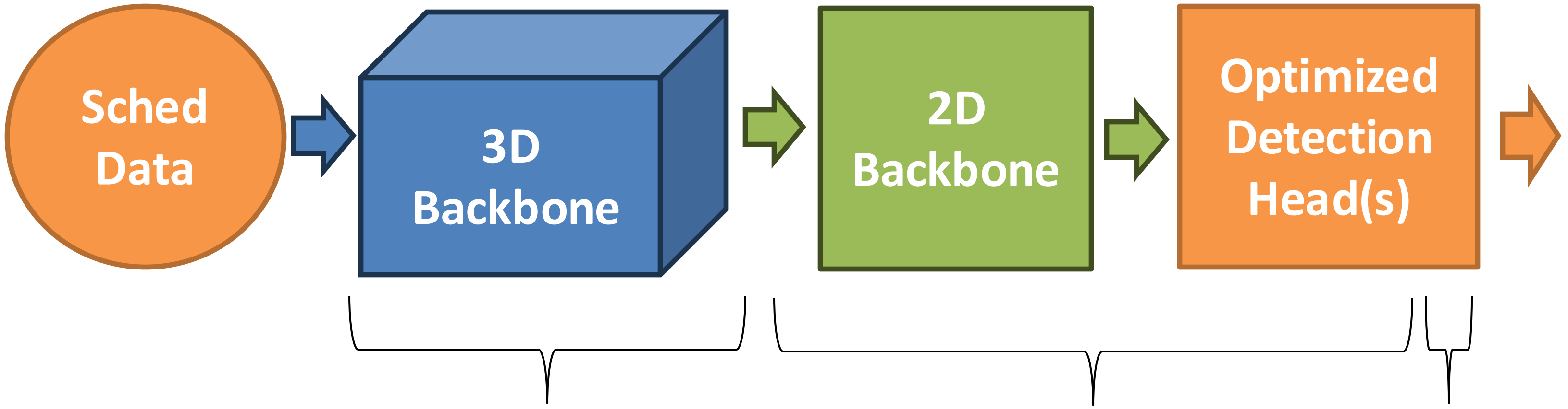
WCET Prediction For a Given Subset of Regions



$$WCET = E_s + E_D + E_R$$

- Dense input processing time
  - Fixed for a given number of regions

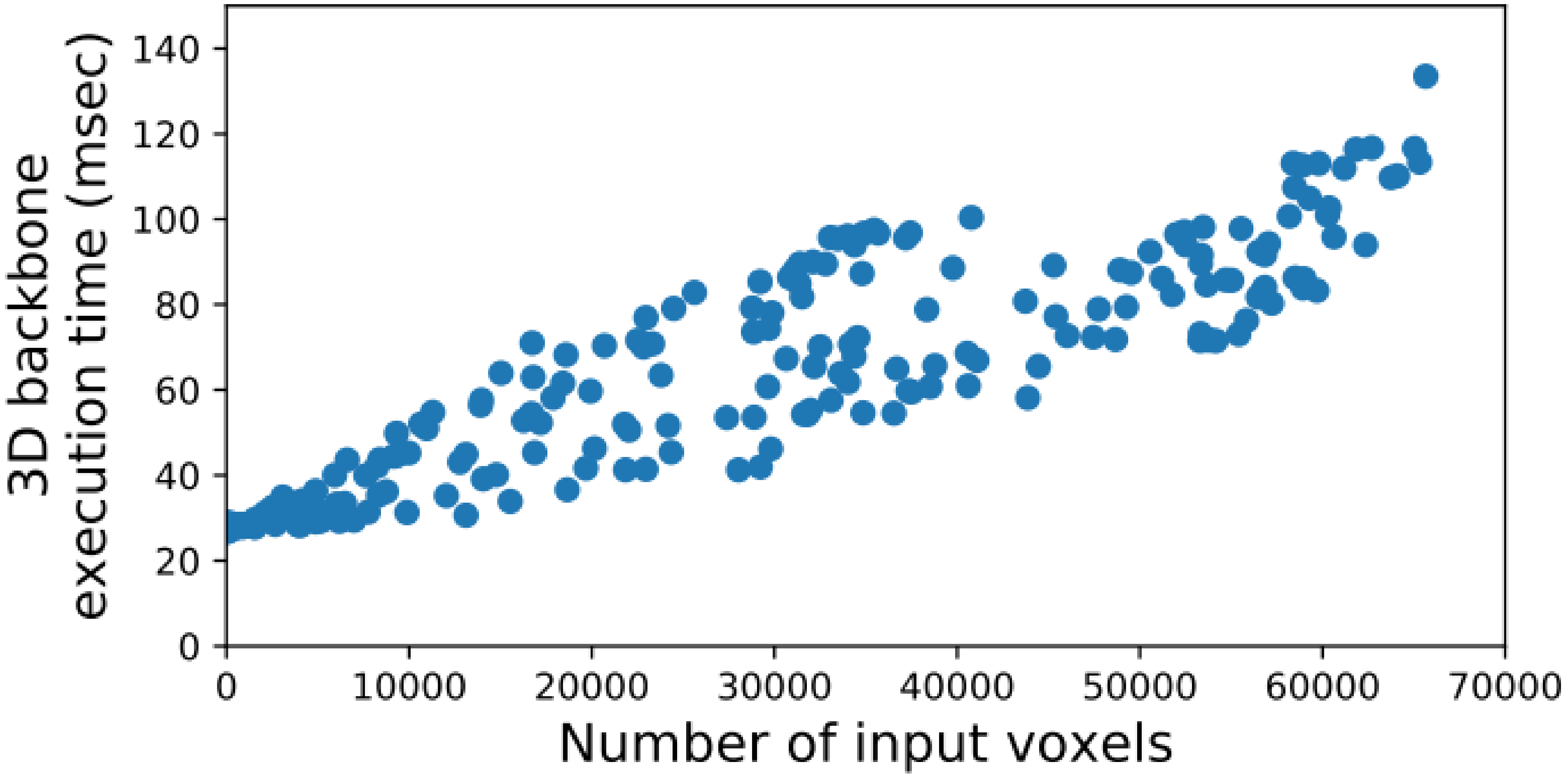
WCET Prediction For a Given Subset of Regions



$$WCET = E_s + E_D + E_R$$

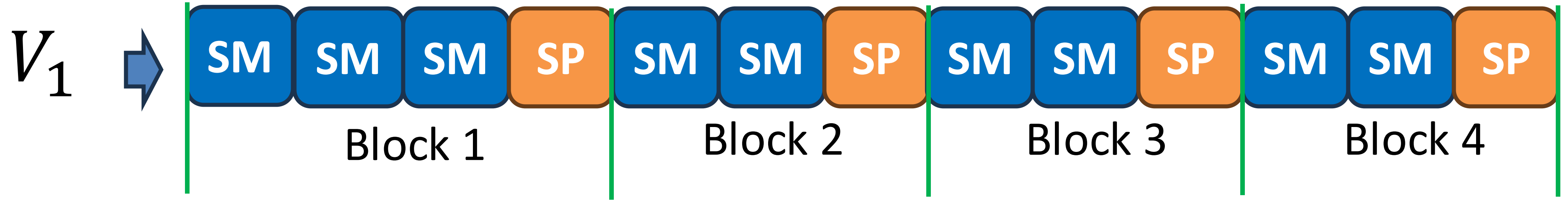
- Sparse input processing time
  - Can we predict it from the number of input voxels?

Calculating E_S



KU THE UNIVERSITY OF KANSAS

3D Backbone Architecture

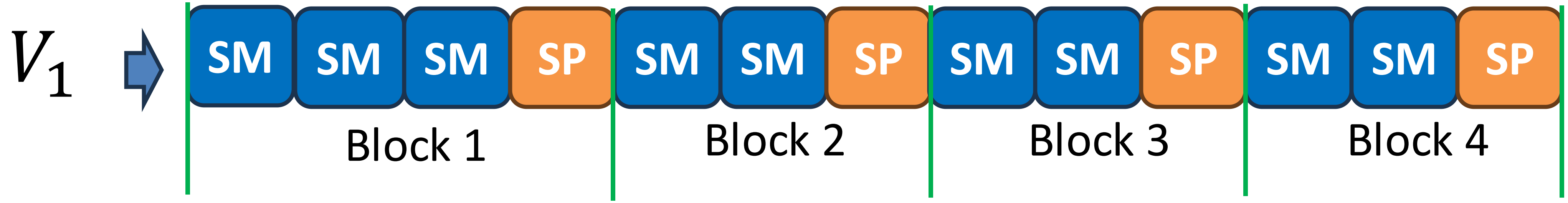


V_1 Input Voxels of Block 1

SM Submanifold Convolution

SP Sparse Convolution

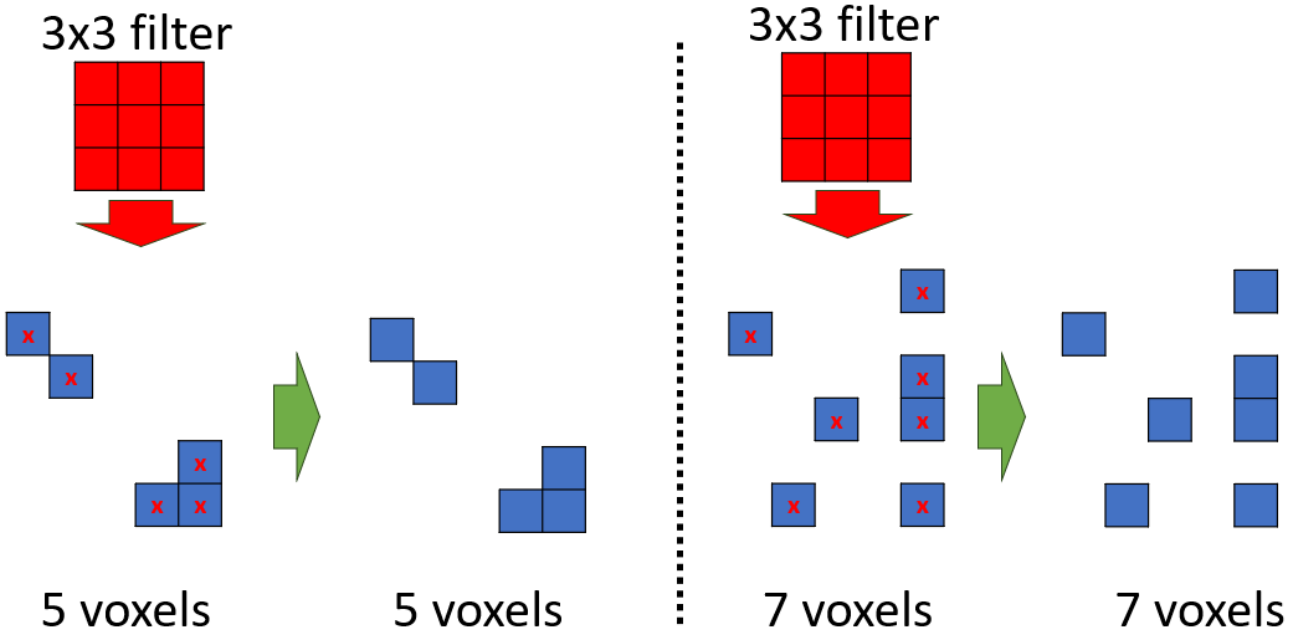
3D Backbone Architecture



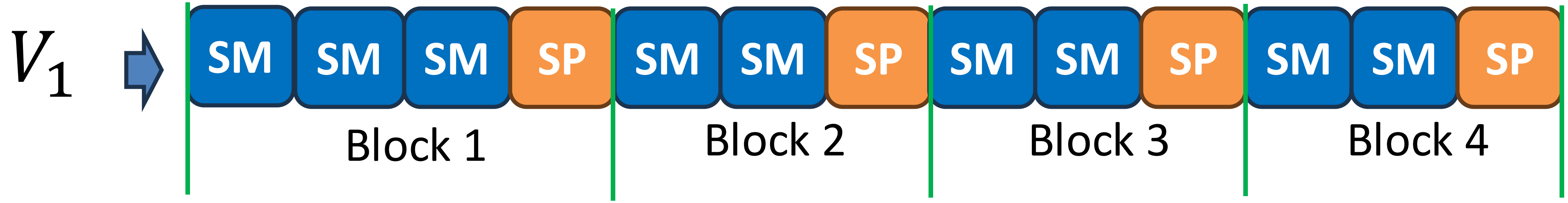
V_1 Input Voxels of Block 1

SM Submanifold Convolution

SP Sparse Convolution



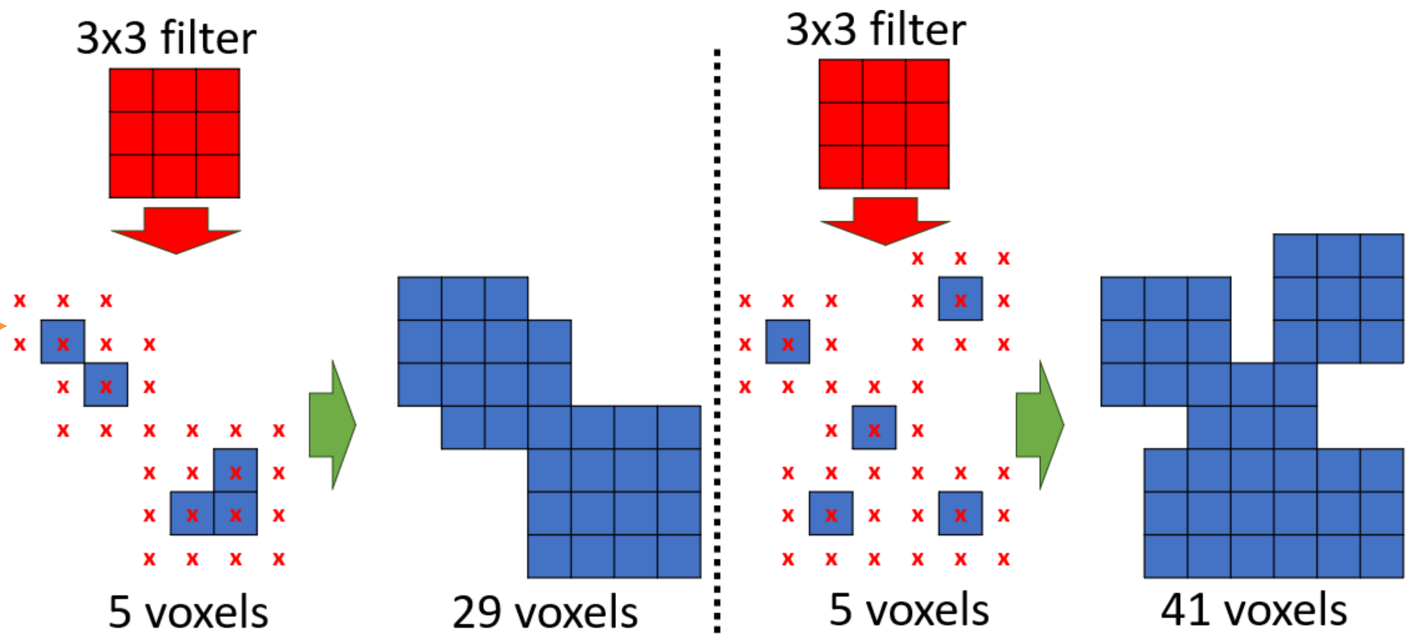
3D Backbone Architecture



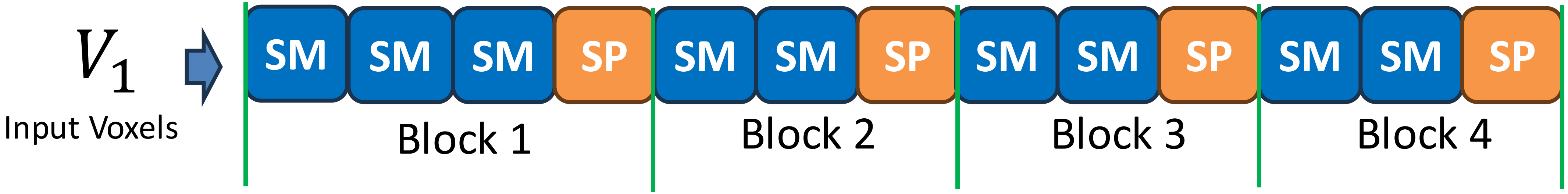
V_1 Input Voxels of Block 1

SM Submanifold Convolution

SP Sparse Convolution

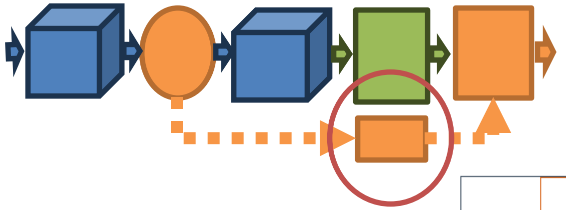


3D Backbone Architecture

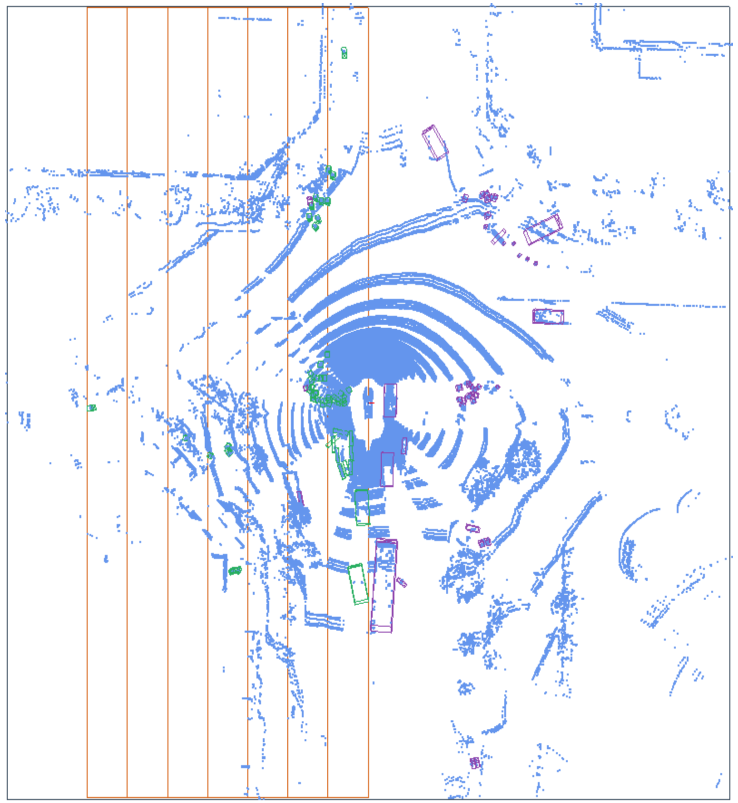


- Layers in each block maintain same input size.
- Latency of each block is predictable for given $|V_1|, |V_2|, |V_3|, |V_4|$.
- We only know $|V_1|$ before execution.
- We utilize the prior values of $|V_2|, |V_3|, |V_4|$ for prediction.

Forecasting

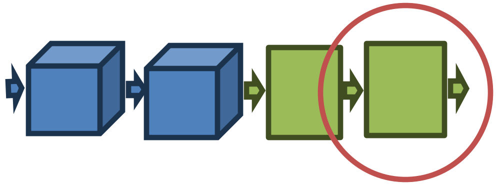


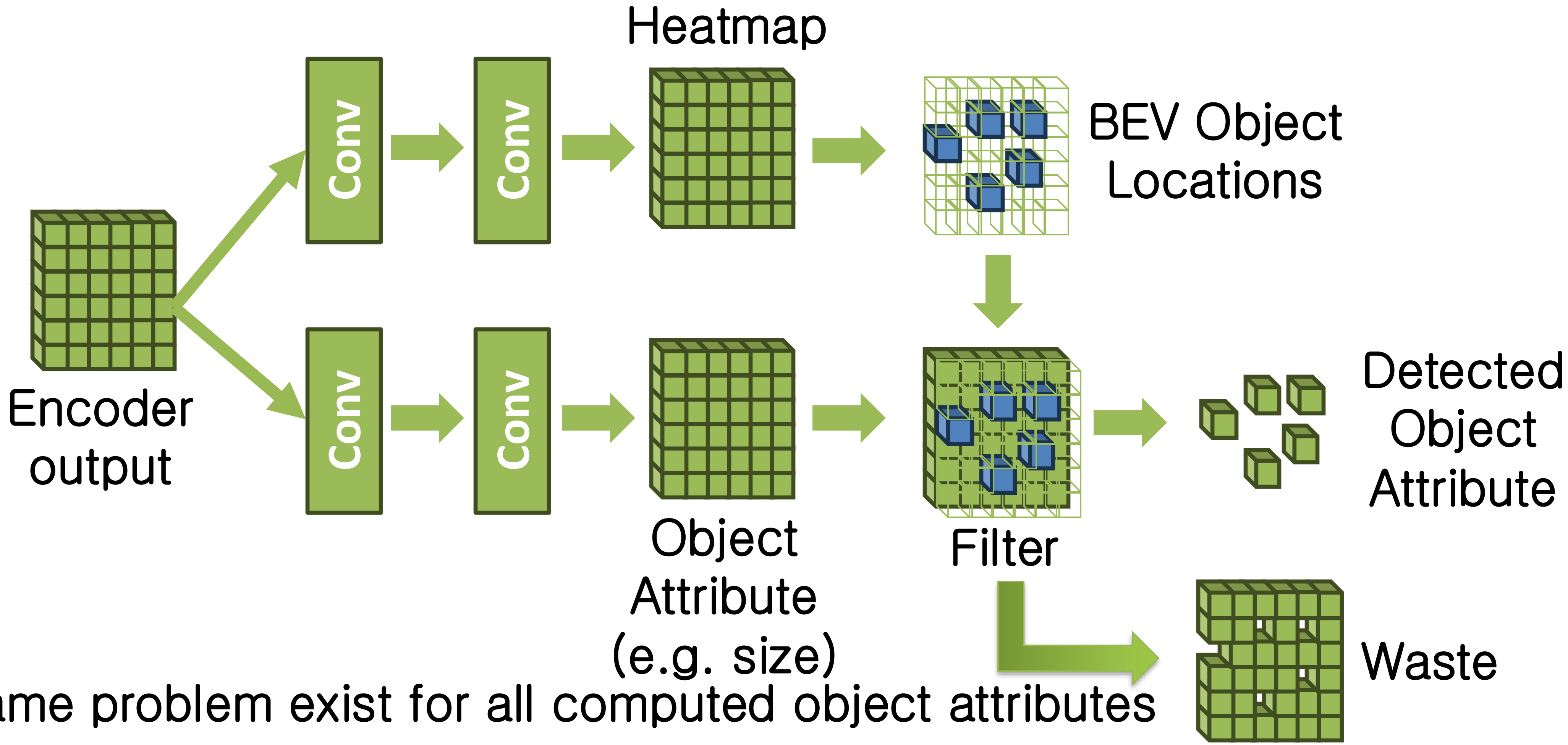
- Predict current object positions of previously detected objects
  - Buffer latest detections for each input region.
  - Forecast all buffered objects.
    - Runs in parallel.
  - Prioritize detections over forecasted objects.



Purple objects are forecasted

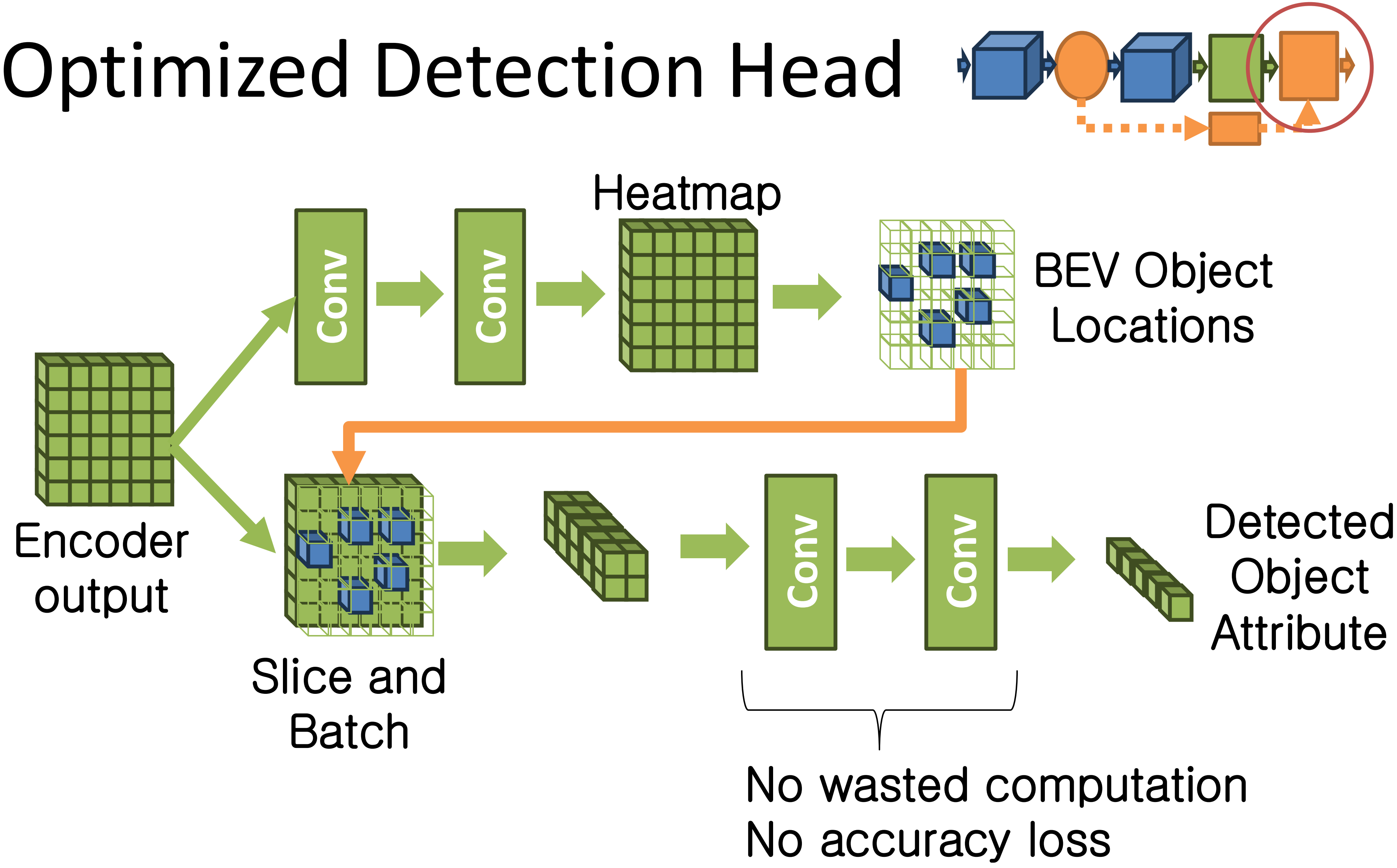
Baseline Detection Head





Same problem exist for all computed object attributes

Optimized Detection Head



Evaluation

- Applied VALO on SOTA detectors:
  - CenterPoint
    - Feat. Ex. → 2D BB → 3D BB → DetHead
  - VoxelNeXt[5]
    - Feat. Ex. → 3D BB → Sparse DetHead
- Evaluated on NVIDIA Jetson AGX Xavier
  - 512-core Volta iGPU
  - 8-core ARM CPU
  - 16 GBs of RAM
- nuScenes dataset
  - Used 30 driving scenes each being 20 seconds

NUSCENES by Motional

THE UNIVERSITY OF KANSAS

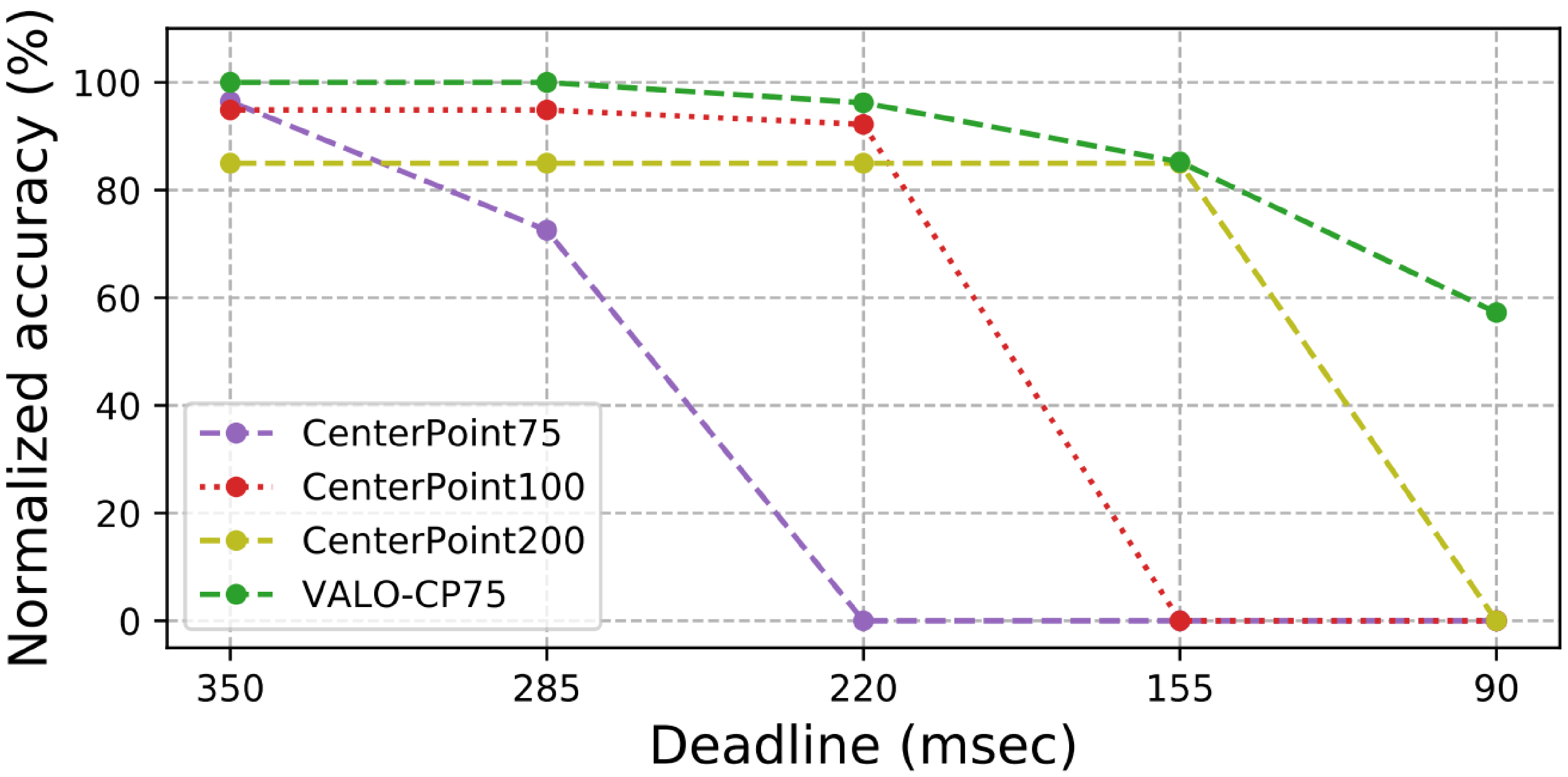
[5] Y. Chen, et. al., "VoxelNeXt: Fully Sparse VoxelNet for 3D Object Detection and Tracking," *2023 CVPR*

Comparison With Baselines

- VALO on CenterPoint (voxel size = 75mm)
- Baseline CenterPoint with different voxel sizes:
  – 75mm
  – 100mm
  – 200mm
- Tested for a range of deadlines
- Results are valid when deadline is met
- Results are nullified when deadline is missed

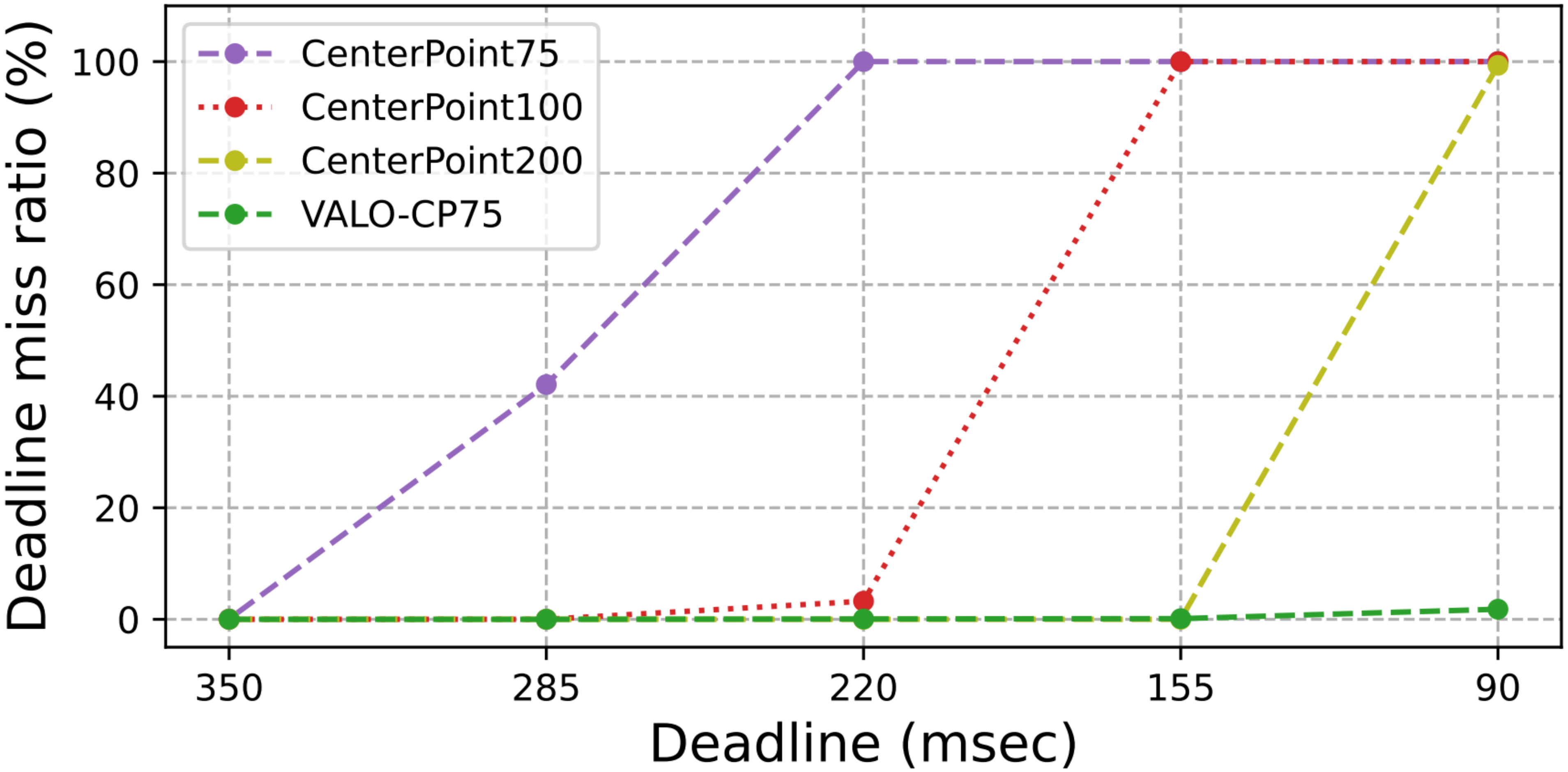
THE UNIVERSITY OF KANSAS

Comparison With Baselines

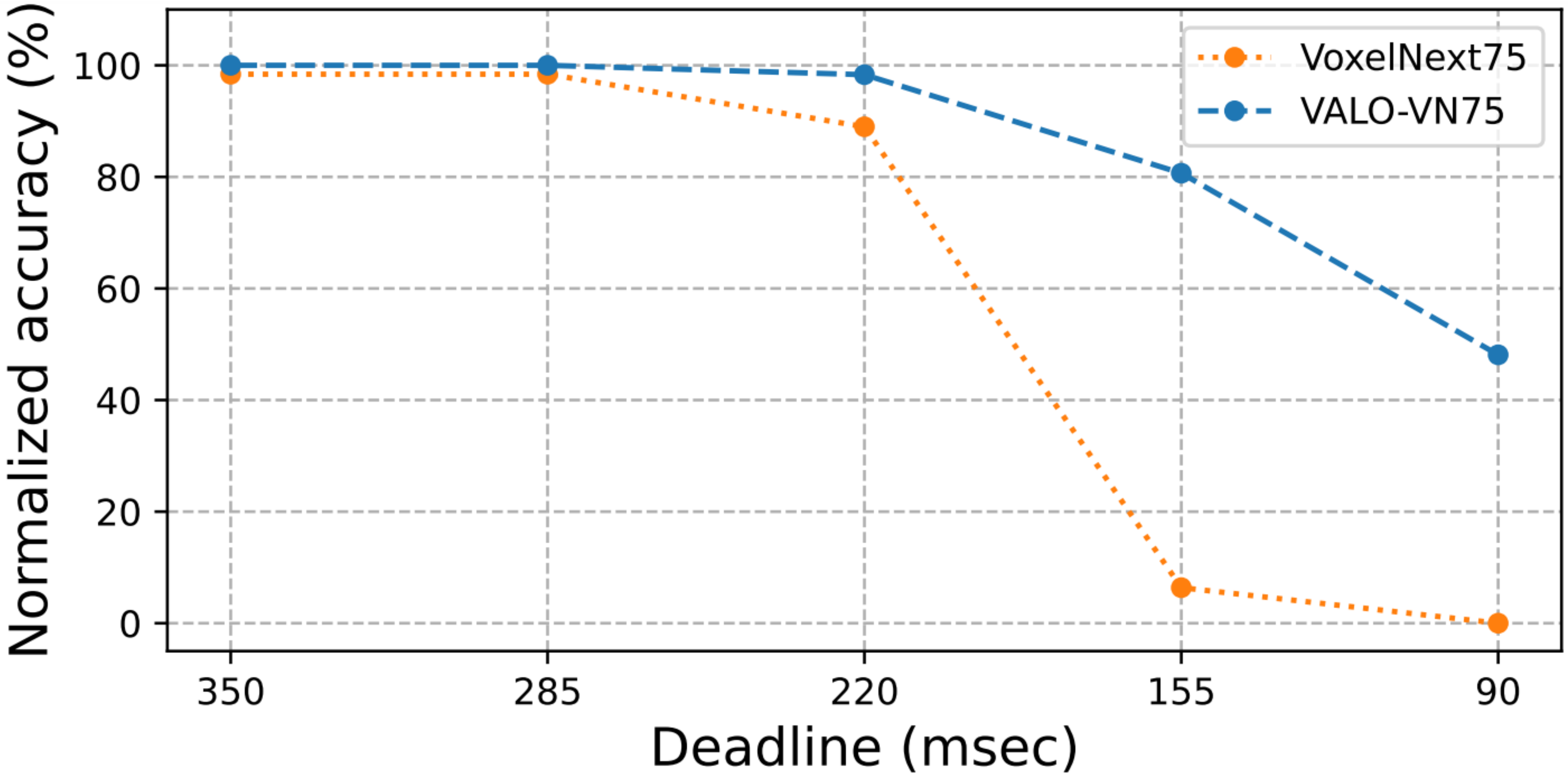


KU THE UNIVERSITY OF KANSAS

Comparison With Baselines



Results on VoxelNeXt



Conclusion

- In this work, we presented:
  - A versatile scheduling framework for LiDAR object detection DNNs
  - We implemented our method on CenterPoint and VoxelNeXt and evaluated its performance on Jetson AGX Xavier
  - Results show that our method significantly surpass baseline methods and provides a versatile solution for anytime perception for LiDAR
- GitHub Link: https://github.com/CSL-KU/VALO

THE UNIVERSITY OF KANSAS

Thank You

THE UNIVERSITY OF KANSAS