Cross-Camera Inference on the Constrained Edge

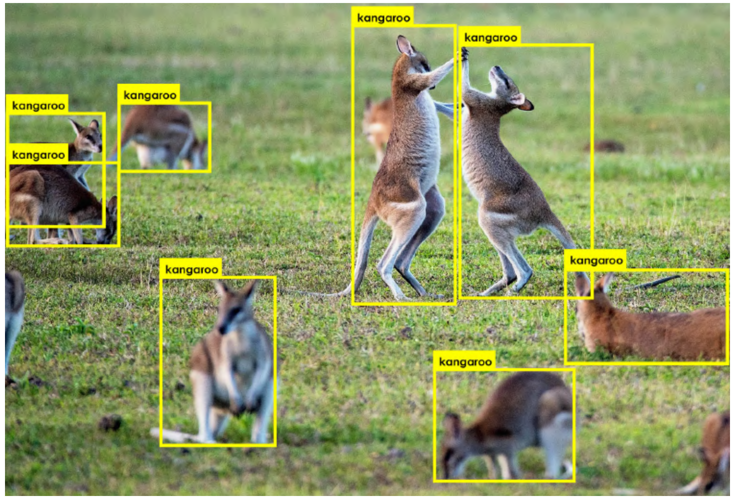
Jingzong Li, Libin Liu, Hong Xu *, Shudeng Wu, Chun Jason Xue

香港城市大學
City University of Hong Kong

中关村实验室
ZGC Lab

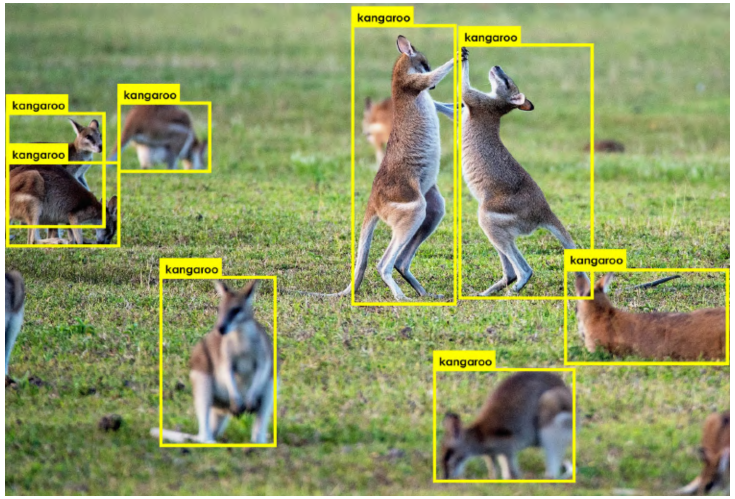
香港中文大學
The Chinese University of Hong Kong

Video analytics become more and more pervasive

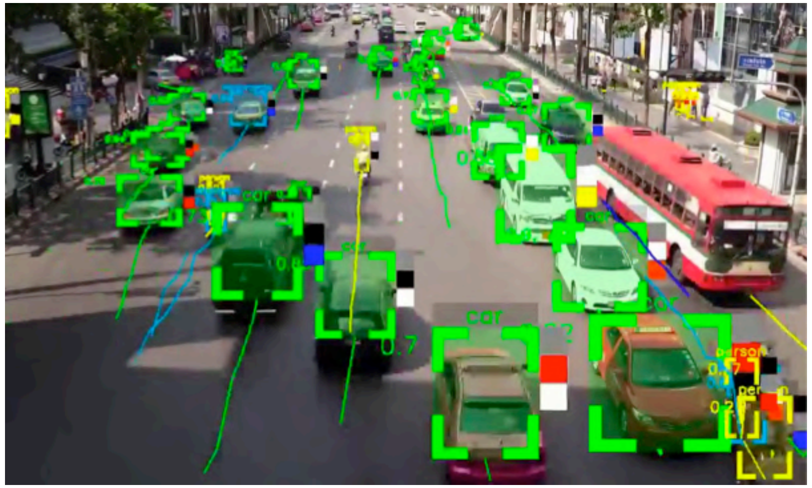


Wild-life camera
Learn about the habit of animals

Video analytics become more and more pervasive

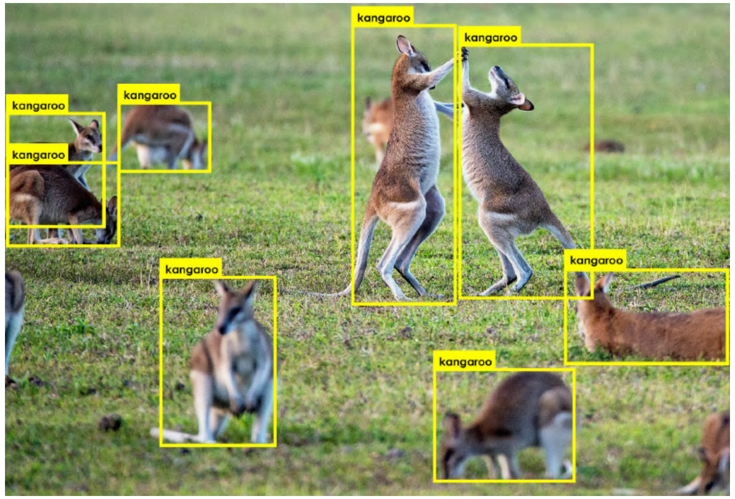


Wild-life camera
Learn about the habit of animals

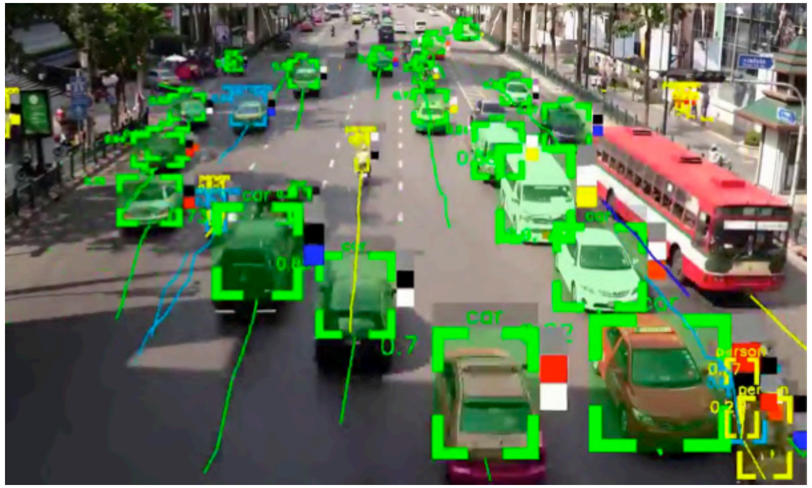


Traffic camera
Monitor the traffic condition

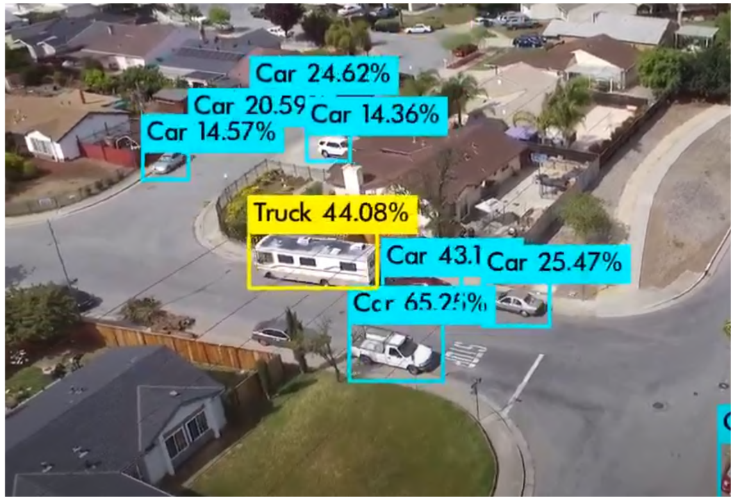
Video analytics become more and more pervasive



Wild-life camera
Learn about the habit of animals



Traffic camera
Monitor the traffic condition



Drone camera
Detect the suspect vehicles

Video analytics on ~~cloud~~ edge era

Computation is shifted to the edge:

- Emergence of smart cameras;
- Unreliable and limited upload bandwidth;
- Privacy reason.

DNNCam™ AI camera

Challenges of video analytics on edge devices

Resource on edge devices is constrained.

VIEW TECHNICAL SPECIFICATIONS >

GPU	128-core Maxwell
CPU	Quad-core ARM A57 @ 1.43 GHz
Memory	4 GB 64-bit LPDDR4 25.6 GB/s

NVIDIA Jetson Nano

Challenges of video analytics on edge devices

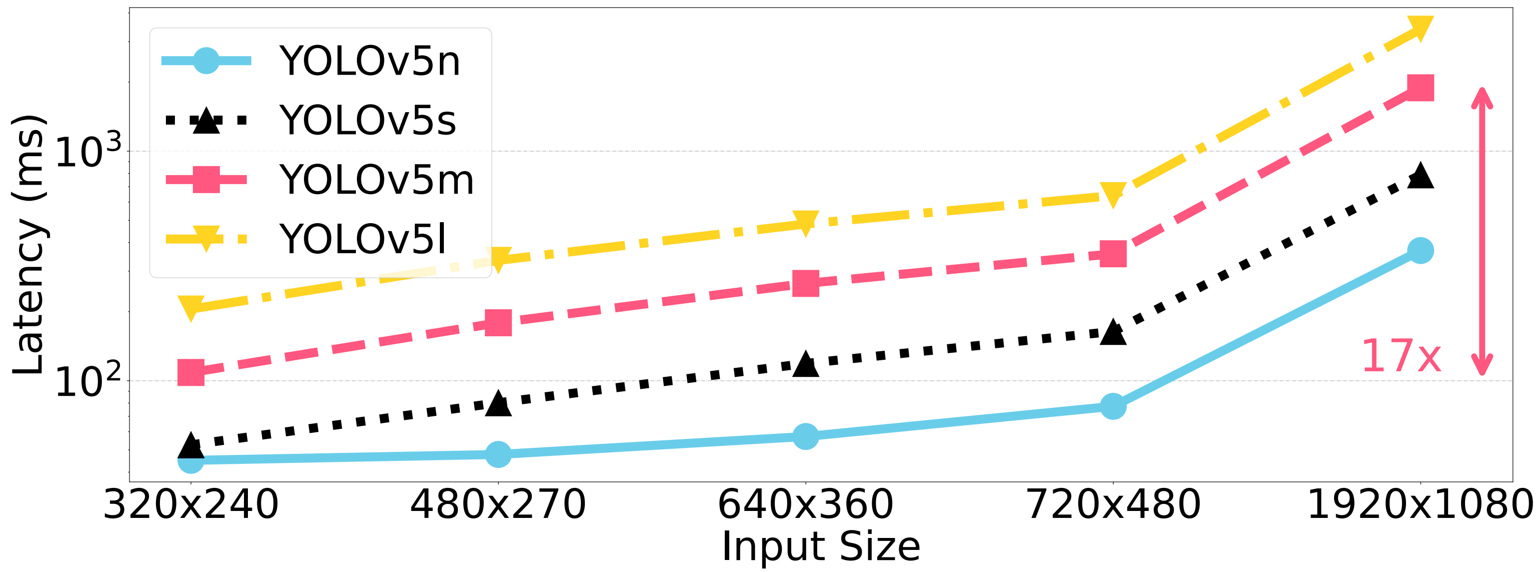
Resource on edge devices is constrained.

Vison-based DNNs are compute-intensive.

Model:
- YOLOv5

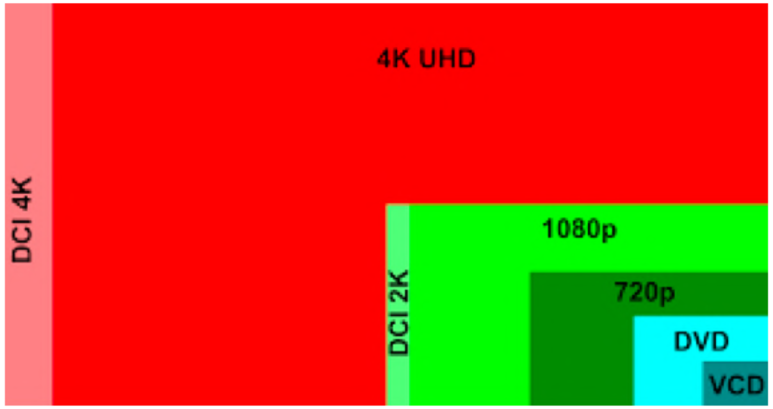
Device:
- Jetson Nano GPU

Dataset:
- NVIDIA AI City Challenge (AICC)



Challenges of video analytics on edge devices

- Resource on edge devices is constrained.
- Vison-based DNNs are compute-intensive.
- Ever-increasing video resolution of cameras.



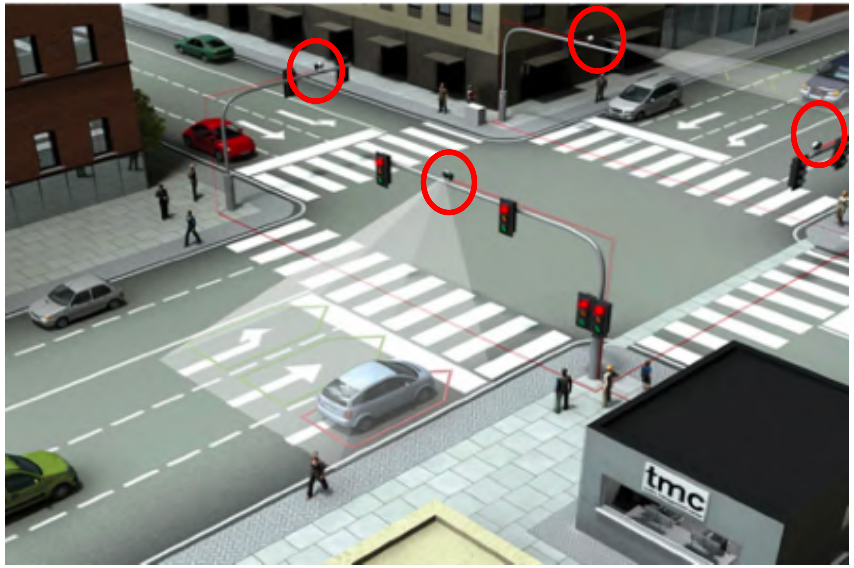
Challenges of video analytics on edge devices

Resource on edge devices is constrained.

Vison-based DNNs are compute-intensive.

Ever-increasing video resolution of cameras.

Compute cost grows with number of cameras.



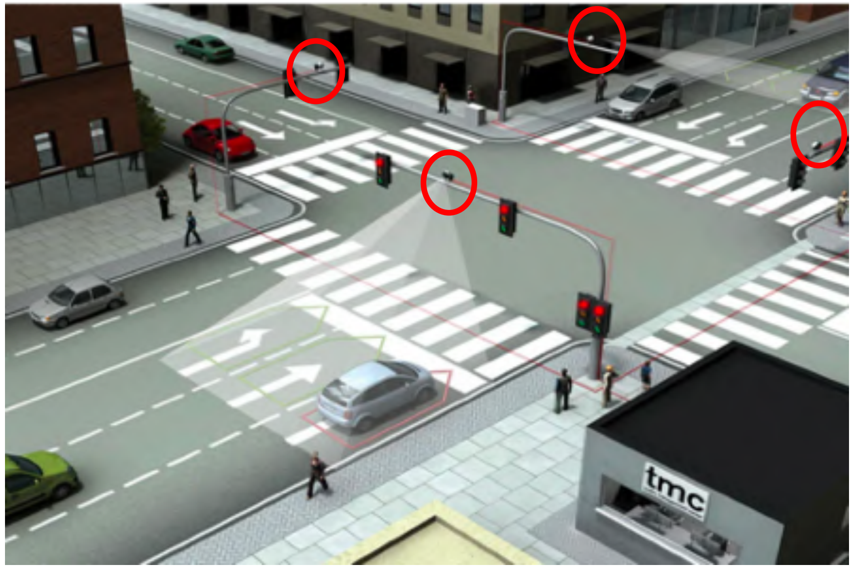
Challenges of video analytics on edge devices

Resource on edge devices is constrained.

Vison-based DNNs are compute-intensive.

Ever-increasing video resolution of cameras.

Compute cost grows with number of cameras.



Resource-efficient inference is needed on constrained edge devices

Overlapping field-of-views (FoVs) across cameras

Cam. 1

Cam. 2

Cam. 3

Cam. 4

Observation: large overlapping FoVs

Source: NVIDIA AI City Challenge (AICC)

Overlapping field-of-views (FoVs) across cameras

Cam. 1

Cam. 2

Cam. 3

Cam. 4

Observation: large overlapping FoVs

Redundant inference work!

Source: NVIDIA AI City Challenge (AICC)

Overlapping field-of-views (FoVs) across cameras

Cam. 1

Cam. 2

Cam. 3

Cam. 4

Observation: large overlapping FoVs

Redundant inference work!

Can they share inference results between each other?

Source: NVIDIA AI City Challenge (AICC)

Overlapping field-of-views (FoVs) across cameras

Cam. 1

Cam. 2

Cam. 3

Cam. 4

Observation: large overlapping FoVs

↓

Redundant inference work!

↓

Can they share inference results between each other?

↓

Only need to run DNNs on pixels that are beyond Overlapping FoVs

Source: NVIDIA AI City Challenge (AICC)

We propose Polly to achieve inference sharing

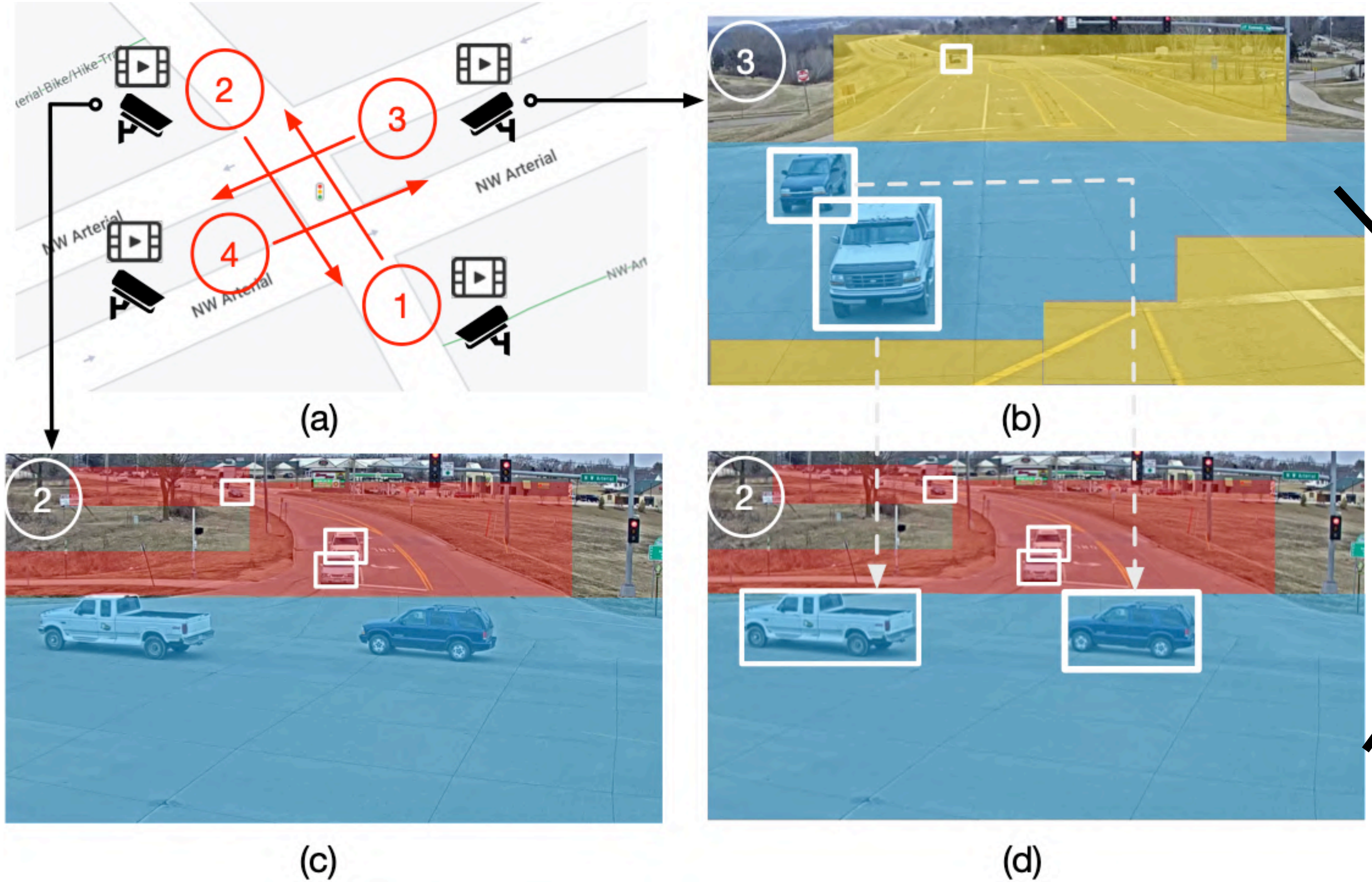


Illustration of inference sharing

Take camera 2 and camera 3 as example

Blue zones are overlapping FoVs

DNN model only needs to run on the unique patches (red).

Challenges to achieve inference sharing

- How to identify the overlapping FoVs **automatically**?
- How to share or transfer the inference results across cameras **effectively**, so that the overall inference accuracy loss is **minimal**?

Challenges to achieve inference sharing

- How to identify the overlapping FoVs **automatically**?
- How to share or transfer the inference results across cameras **effectively**, so that the overall inference accuracy loss is **minimal**?

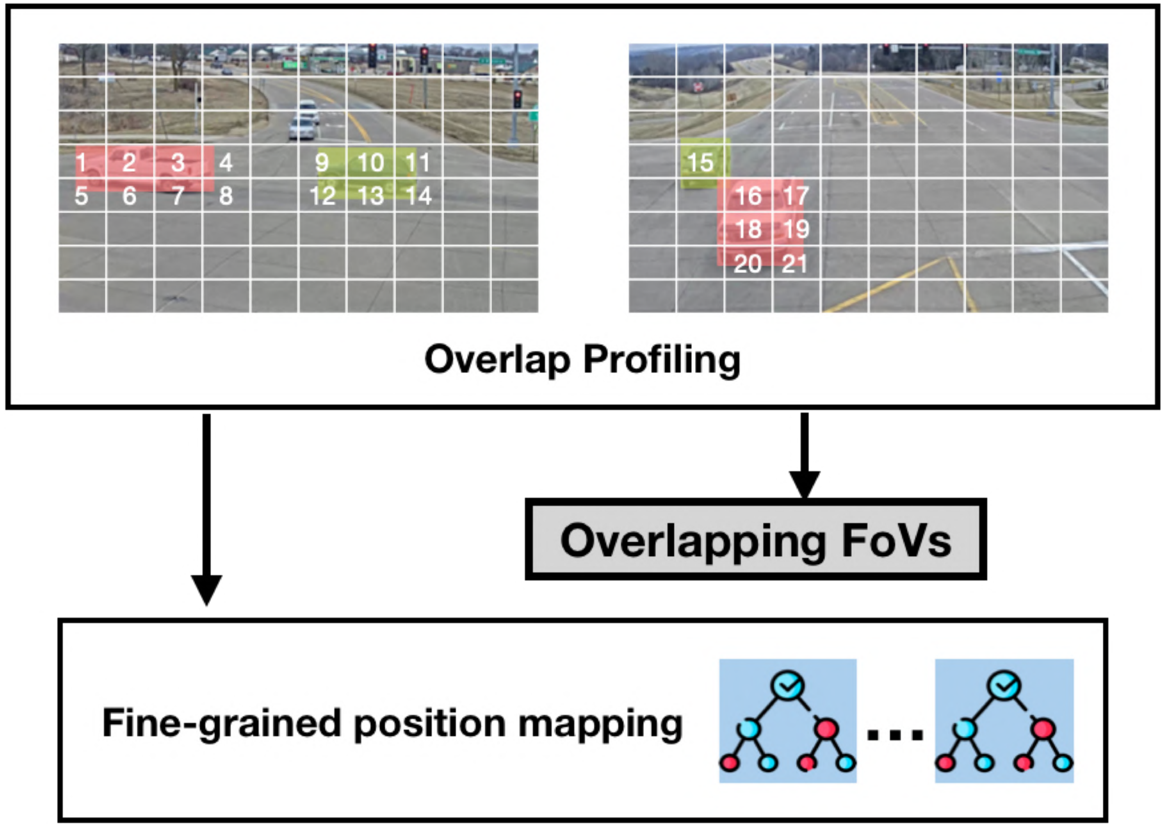
- Offline phase
- Online phase

Polly System Overview

Server (Offline Phase)

Edge Device (Online Phase)

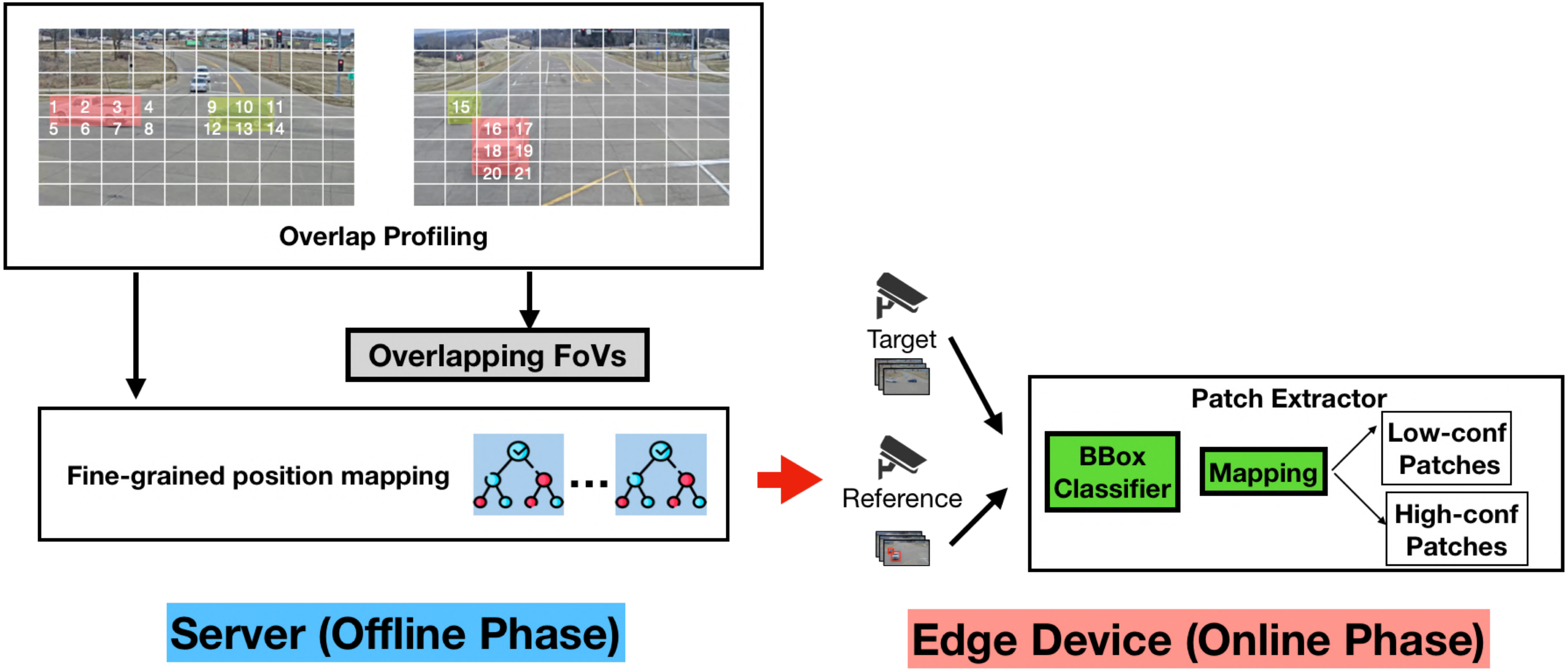
Polly System Overview



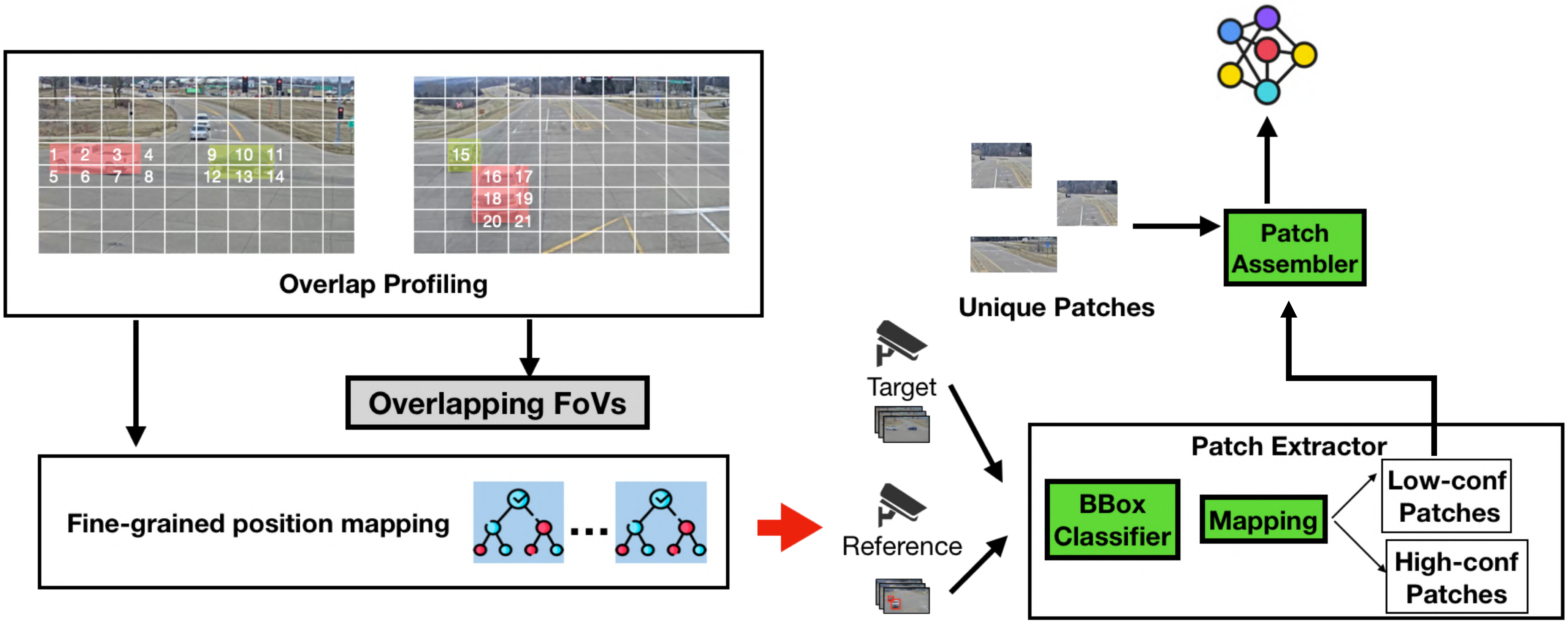
Server (Offline Phase)

Edge Device (Online Phase)

Polly System Overview



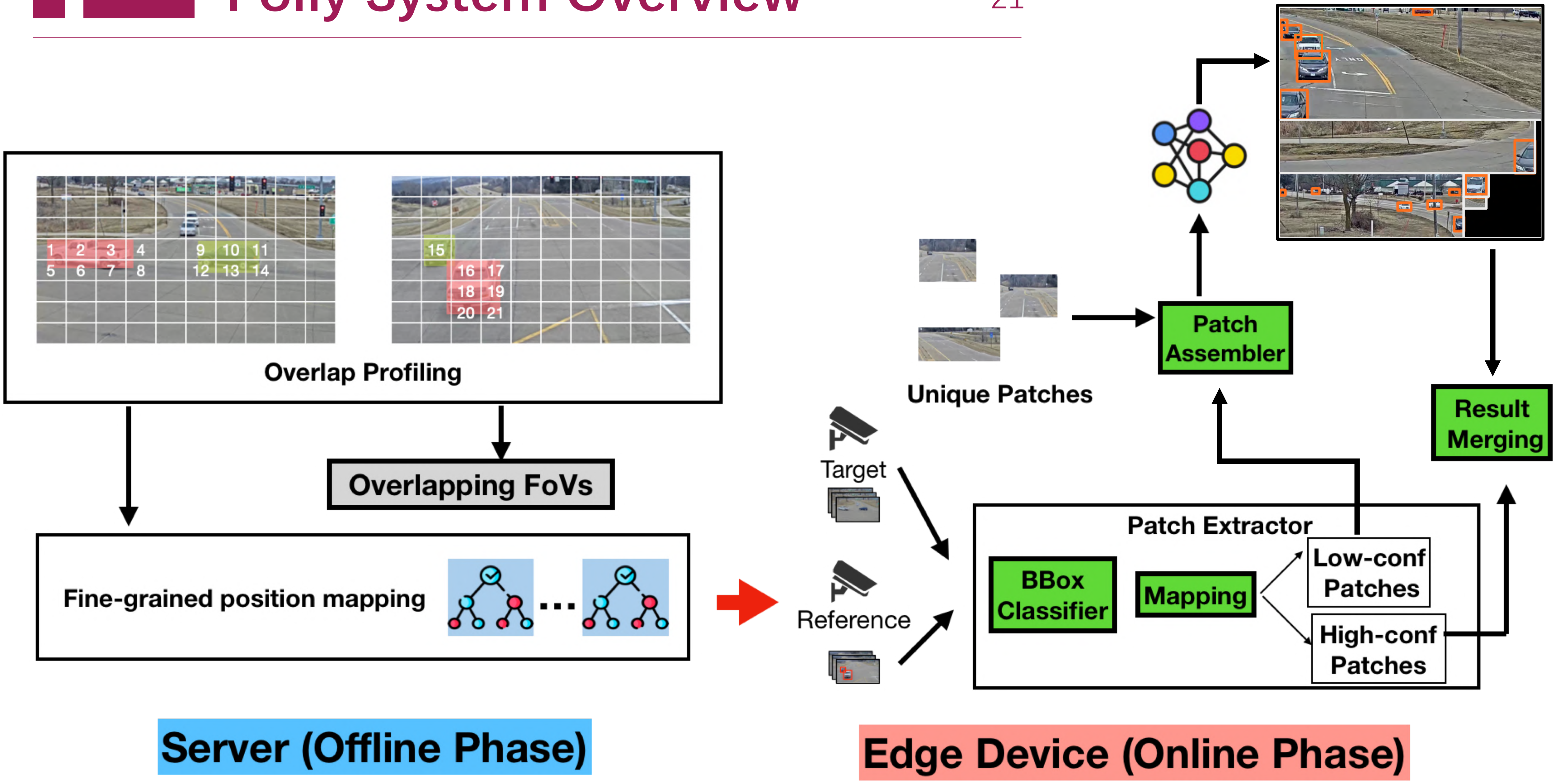
Polly System Overview



Server (Offline Phase)

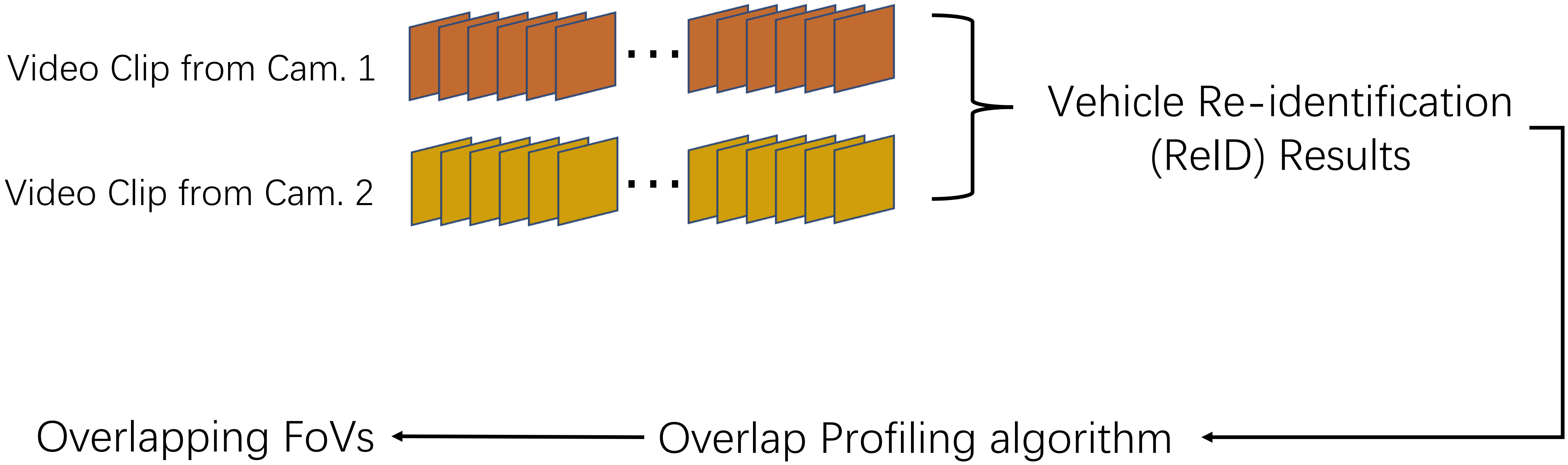
Edge Device (Online Phase)

Polly System Overview



Offline Phase - Overlap Profiling Algorithm

Rationale: The algorithm utilizes the positions of the same vehicle in different cameras to profile the overlapping FoVs automatically between them.



Offline Phase - Overlap Profiling Algorithm

Example of overlap Profiling:

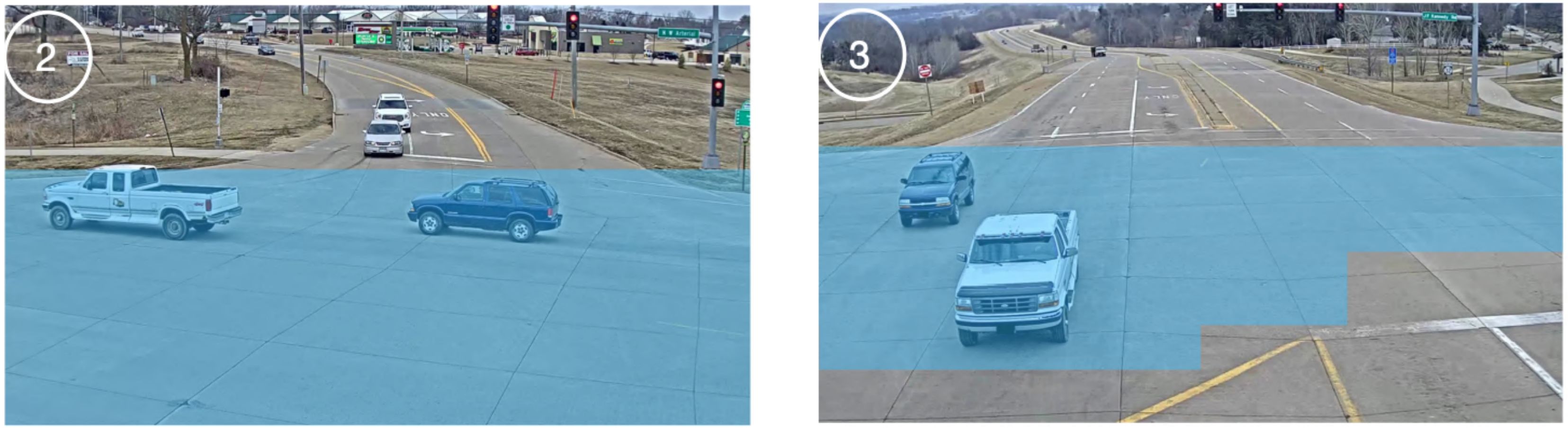
(a) Cam. 2 (b) Cam. 3

Offline Phase - Overlap Profiling Algorithm

Example of overlap Profiling:

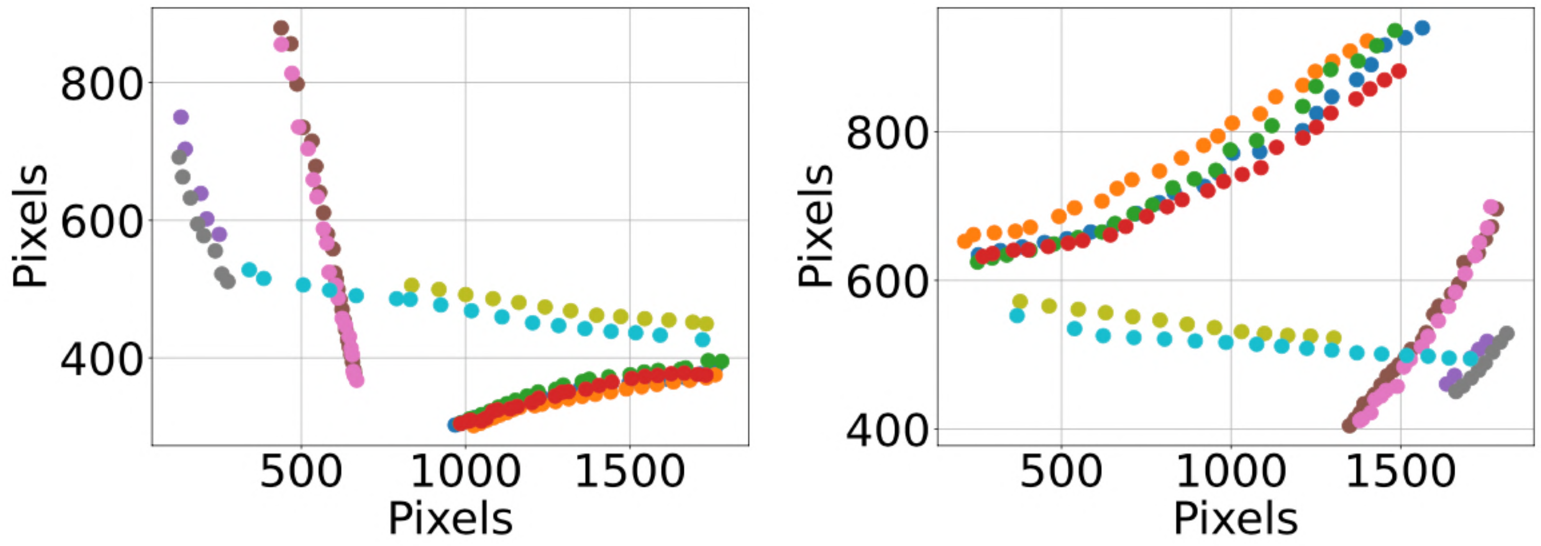
(a) Cam. 2 (b) Cam. 3

Overlapping FoVs:



Offline Phase - Fine-grained Position Mapping

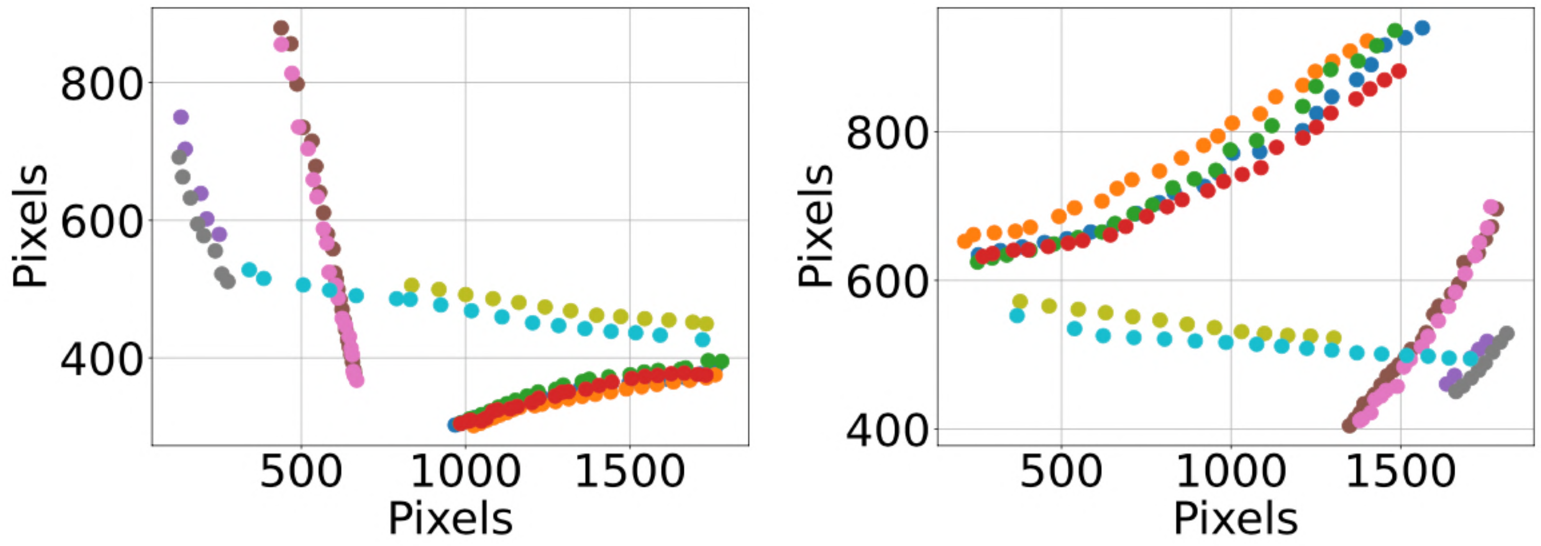
Based on the overlapping FoVs from the profiling algorithm, Polly can further build a fine-grained position mapping for effective inference sharing.



Trajectories of randomly selected vehicles from two cameras. Dots with the same color represent the centroids of the same vehicle.

Offline Phase - Fine-grained Position Mapping

Based on the overlapping FoVs from the profiling algorithm, Polly can further build a <u>fine-grained position mapping</u> for effective inference sharing.



Trajectories of randomly selected vehicles from two cameras. Dots with the same color represent the centroids of the same vehicle.

We can see that vehicles with **similar trajectories** in one camera appear in the other one with also very **similar trajectories**.

Offline Phase - Fine-grained Position Mapping

Position Mapping:
- Input: [x, y, l]
- Ouput: [x', y', l']

Compare two approaches:
- Black-box: MLP (64-64-3)
- White-box: Regression Tree

Data augmentation:
- Enrich training samples
- Enhance mapping accuracy

Method	R^2	Training Time (s)	Inference Time (ms)
MLP	0.950	631	0.38
Multi-output Regression Tree	0.996	5	0.27

TABLE III: Performance of different mapping methods

Offline Phase – Other design

grass fields, bushes and houses

- **Background removal**: running vehicle detection on the background areas where vehicles do not exist at all is a waste of resource,

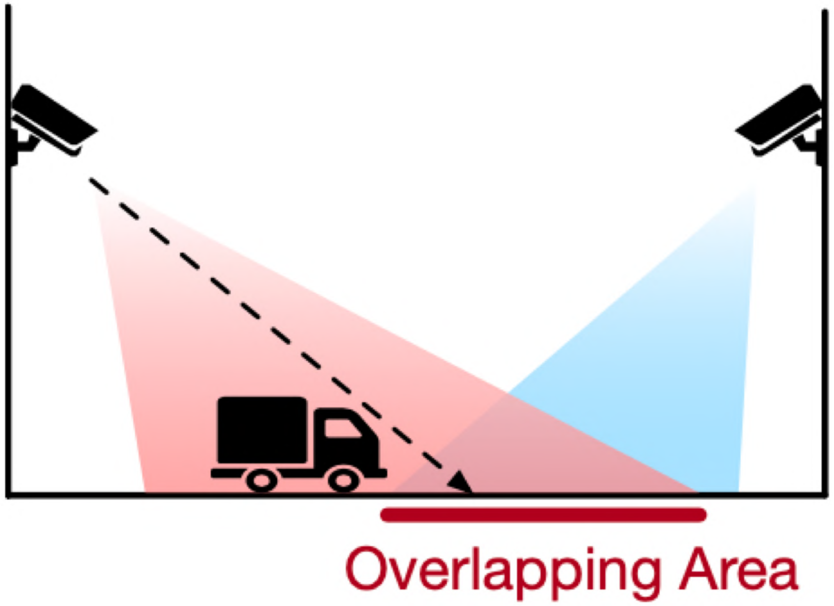
- **Reference camera selection**: we select the camera that has the largest average overlapping FoVs with the remaining ones as the reference camera.

Please refer to the paper for more details.

Online Phase – Perspective Effect

Directly mapping the detected vehicles from the reference camera to the target camera can lead to **erroneous results** due to the low quality of shared inference result.

One main cause is the **perspective effect**.

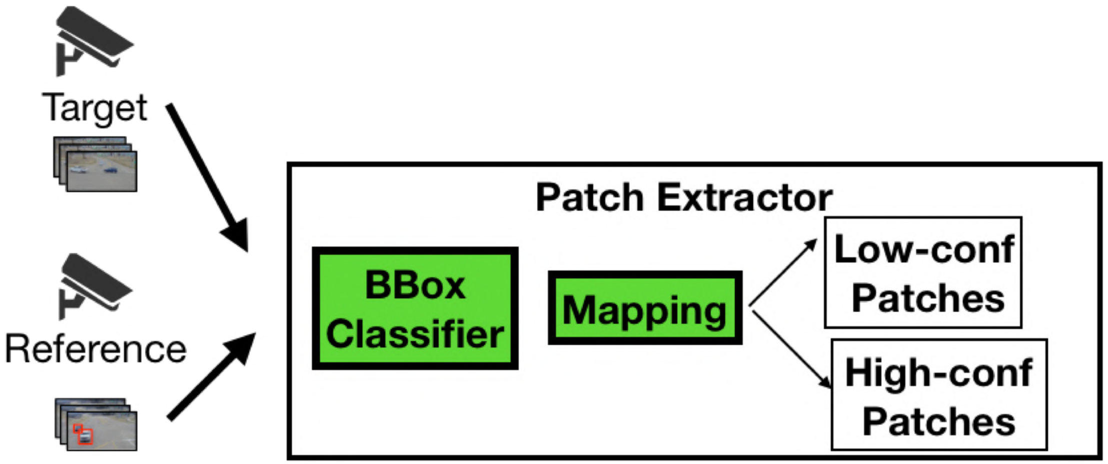


An illustration of the perspective effect.

Online Phase – Patch Extractor

Problem: some parts of the reference camera's frame has low sharing quality.

Solution: just run the DNN model on the corresponding parts of the frame in the target camera. We design the **patch extractor** to identify the low-confidence parts and extract them.

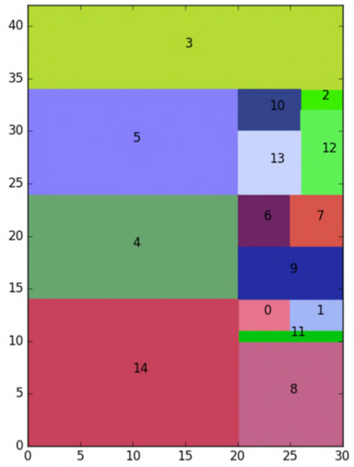


Besides low-confidence patches, the target camera's **unique FoVs** are also extracted.

Online Phase – Patch Assembler

Problem: Both the low-confidence and unique patches need to be processed by the DNN for object detection, and running the model on each patch individually is inefficient.

Solution: we design the **patch assembler** to tile these patches into a minimized rectangle so as to reduce inference latency (e.g. 1888ms for 1920×1080 full frame, 265ms for a 640×360 tiled rectangle).

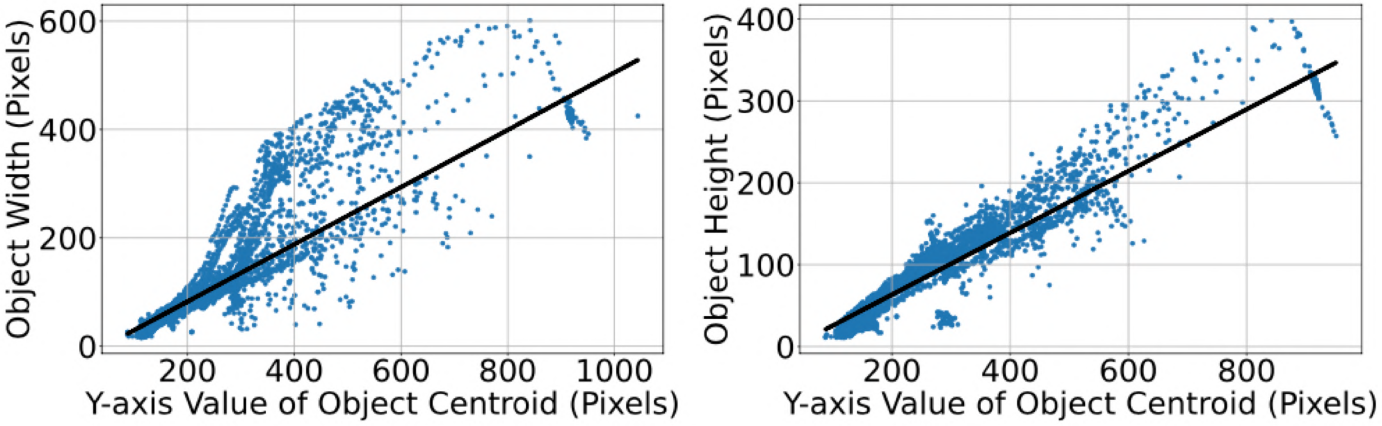


2D Bin packing problem:

- NP-hard
- Fix the width of the bin
- Guillotine bin packing algorithm

Online Phase – Patch Assembler

Observation: the object size is roughly linearly relative to the Y coordinate of the object's centroid.



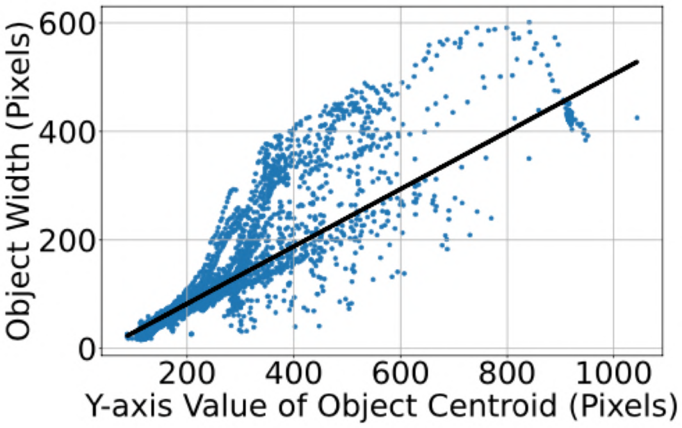
(a) Y-axis value vs. width. (b) Y-axis value vs. height.

Fig. 8: Y-axis value of object centroid versus object size.

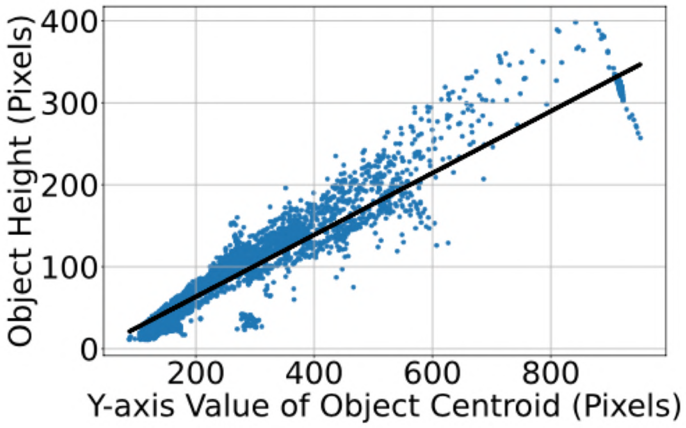
Optimization: apply down-sample to certain patches.

Online Phase – Patch Assembler

Observation: the object size is roughly linearly relative to the Y coordinate of the object's centroid.



(a) Y-axis value vs. width.



(b) Y-axis value vs. height.

Fig. 8: Y-axis value of object centroid versus object size.

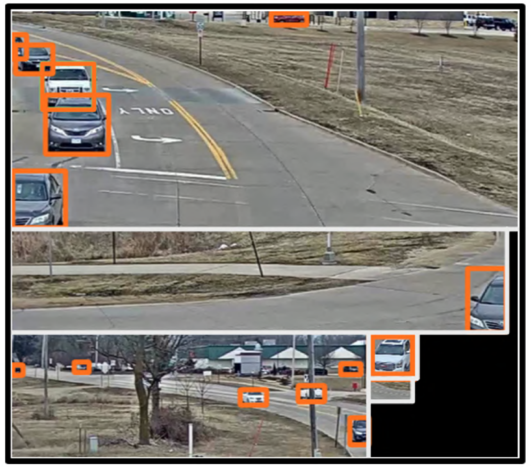
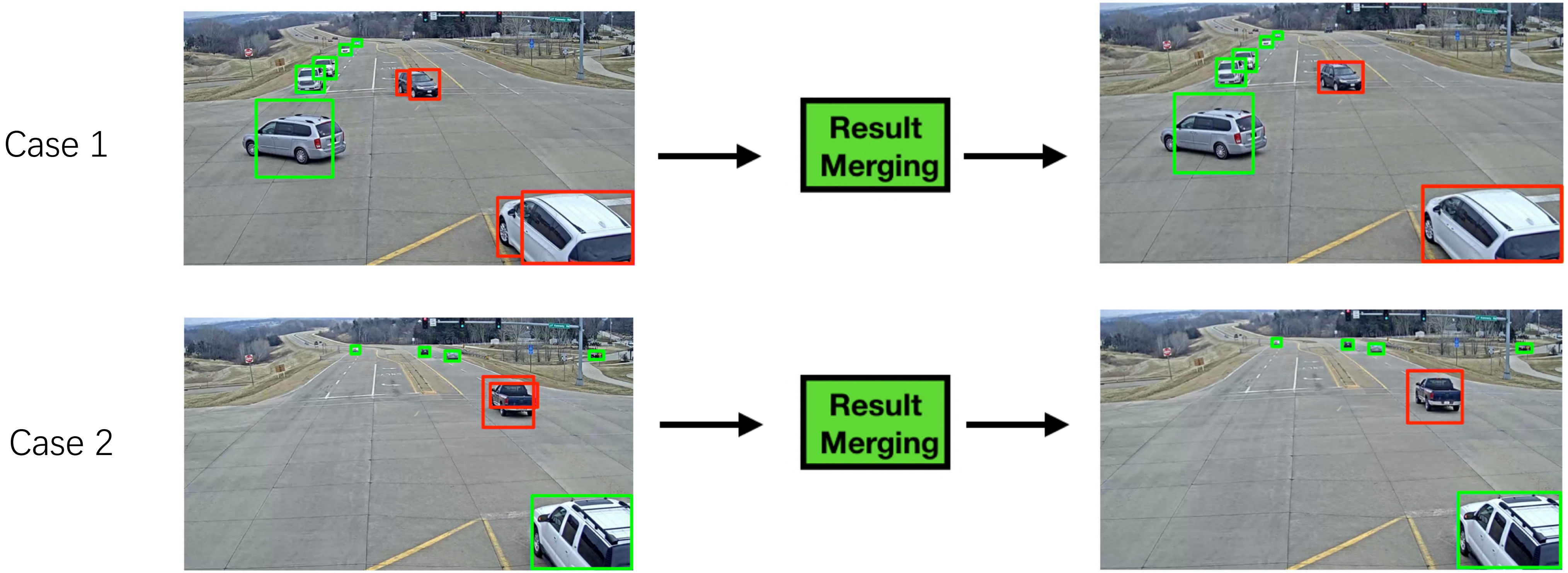


Fig. 9: An example of the tiled rectangle with detection results.

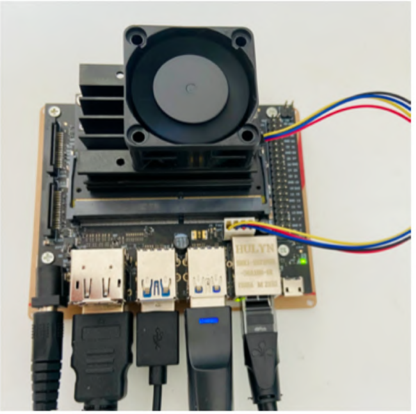
Optimization: apply down-sample to certain patches.

Online Phase – Result Merging

At the last step, Polly needs to combine the inference results from the two pipelines, i.e., **inference sharing** from the reference camera, and **direct inference** of the titled image.



Evaluation – Experiment Settings



Testbed: We run our experiments on Jetson Nano, a commonly used edge device equipped with one 128-core NVIDIA Maxwell GPU.

Dataset: NVIDIA AI City Challenge (AICC) [1]

Detection Models: YOLOv5 variants with different sizes

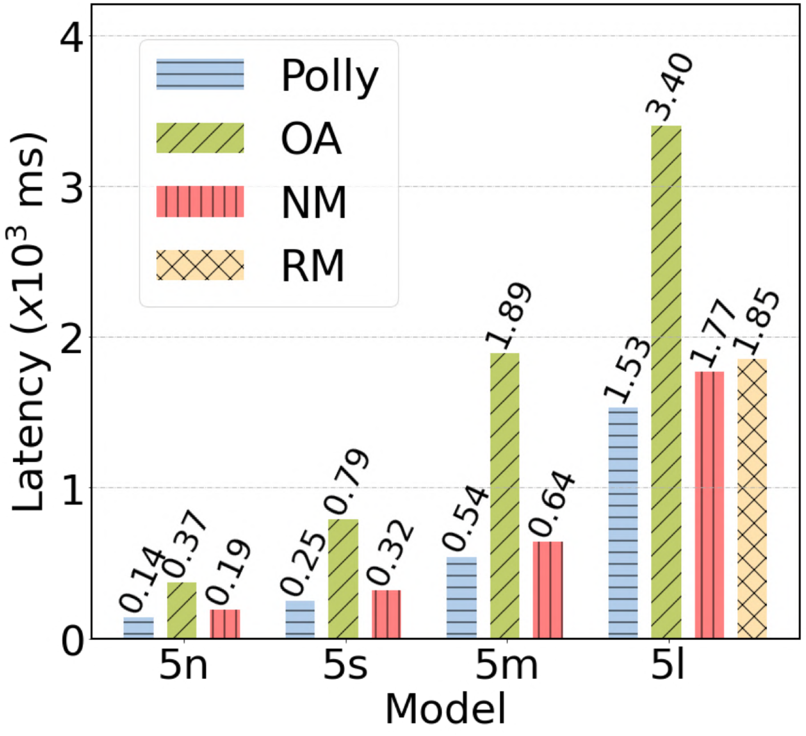
Baselines:
- Overlapping-Agnostic (OA)
- Naive-Merging (NM)
- Remix-Mimic (RM)

Metrics:
- End-to-end latency
- Detection Accuracy (F1)

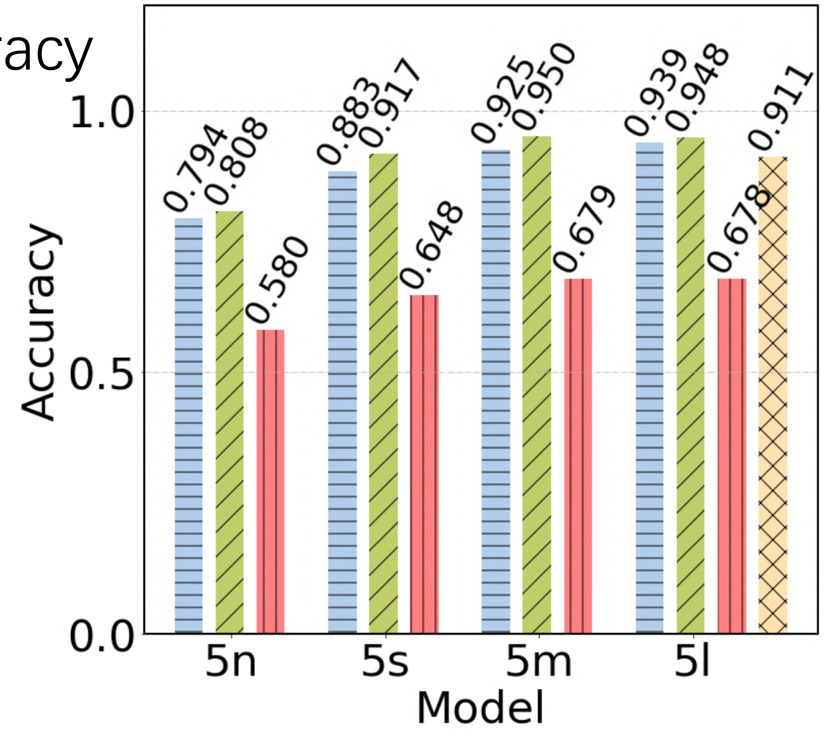
[1] 2022 NVIDIA AI City Challenge. https://www.aicitychallenge.org/.

Evaluation – Overall Performance

Latency



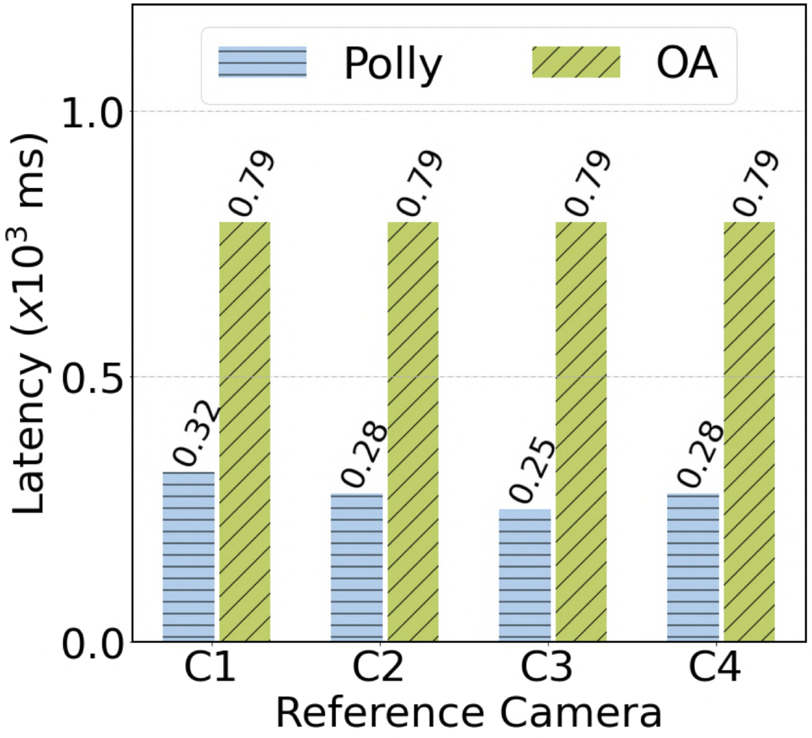
Accuracy

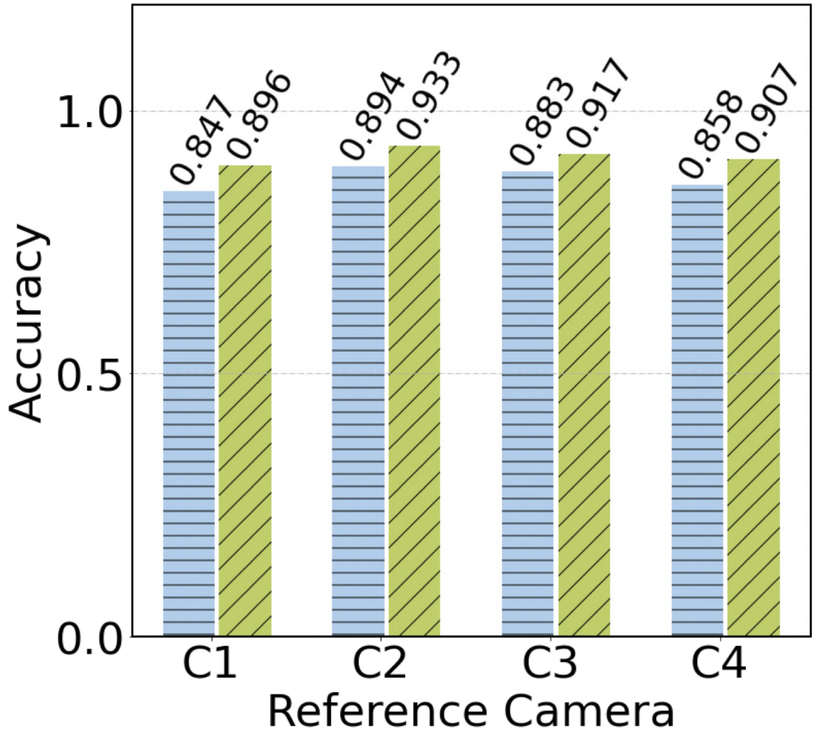


- Polly **outperforms** all baselines.
- Compared with OA, Polly's overall latency reductions range from 55% to 71%.

- Polly's accuracy is almost **on par with** OA, with at most 3.7% loss.
- Polly can achieve better accuracy while reducing latency by **70%** compared to RM.

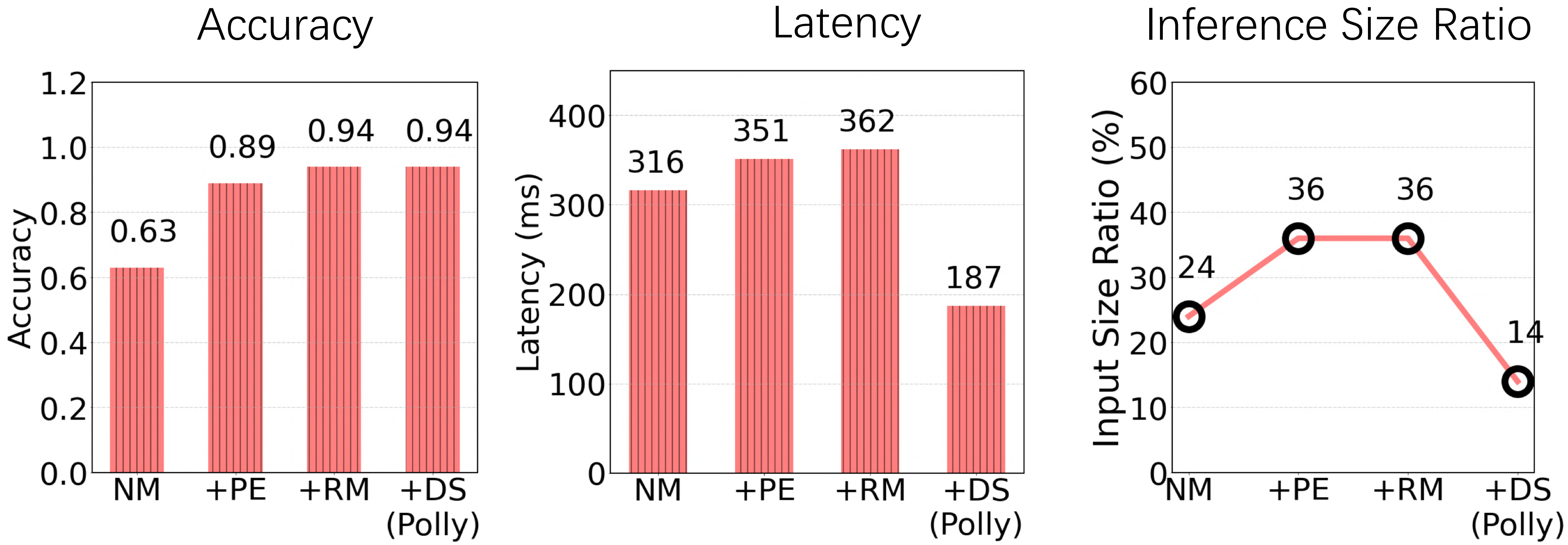
Evaluation – Impact of Reference Camera Selection





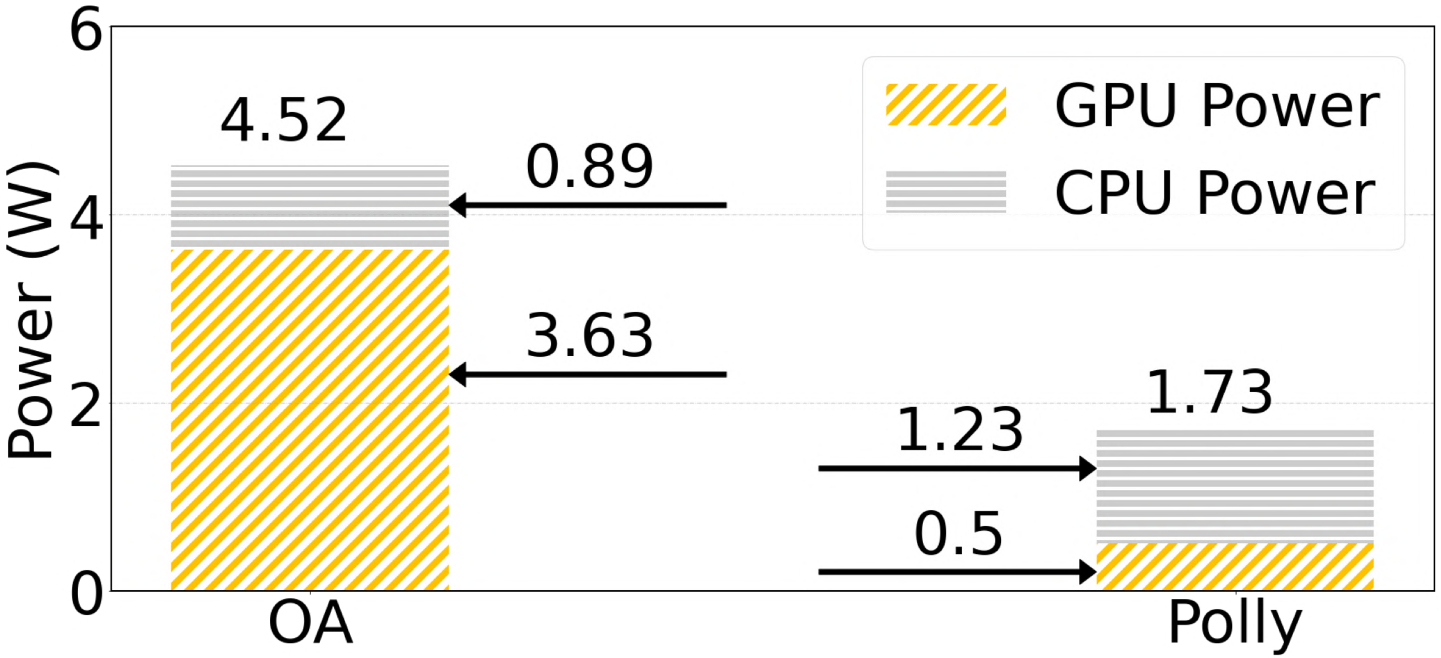
- Inference sharing is **consistently** effective with different cameras selected as the reference.

Evaluation – Performance Breakdown



- PE brings **considerable** accuracy gain.
- DS improves the latency by 48 % **without** accuracy loss.

Evaluation – Energy Consumption



Polly saves 61.73% power in total against OA:

- 86.23% GPU power reduction
- Slightly more CPU consumption

Conclusion

- **Problem**: In the cross-camera scenario, video analytics tasks pose severe burden for resource-constrained edge devices.
- **Our contribution**: Polly, the **first** full-fledged video analytics system to achieve inference sharing.
- **Results**: Polly achieves significant latency reduction almost without impairing accuracy.

Cross-Camera Inference on the Constrained Edge

Thank you

{ jingzong.li@my.cityu.edu.hk }