GreenAutoML4FAS: Demonstration und Evaluation (*AP6*)

VISCODA GmbH, 2026

Lin
Chen

Tobias
Geib

Kai
Cordes

Hellward
Broszio

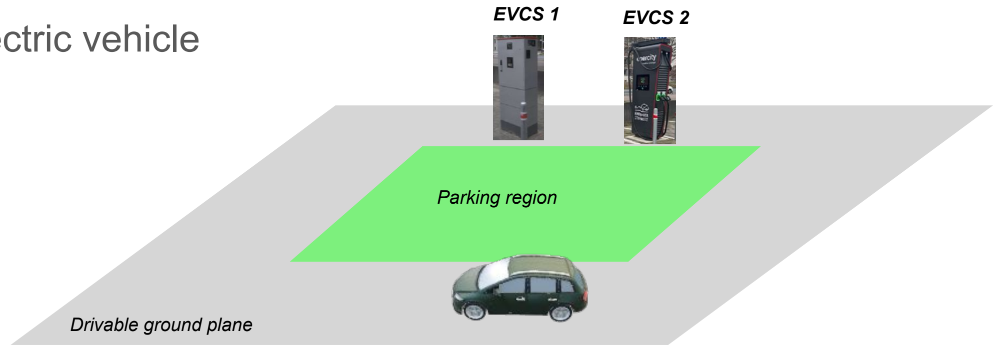
Overview

Objective: Parking assistance system for electric vehicle

- Low energy consumption
- 3D perception is required

Dual track of development

- AI model development
  - Network pruning: EVCS detector
  - AutoML: single-view depth estimation network
- Application software development
  - EVCS localization and tracking
    - Where are they?
  - Single-view depth estimation
    - How far is the object? (EVCS, any dynamic objects)



EVCS: Electric Vehicle Charging Station

Overview (Working packages)

- EVCS Detection Dataset (UAP 6.3)
  - Dataset Overview
  - Detection Baseline

- GreenAutoML4FAS Demonstration
  - Application software development (UAP 6.1 + 6.2)
  - EVCS detection and tracking
    - Evaluation (UAP 6.4)
    - Demonstration (Live) (UAP 6.5)
  - Single-View depth estimation
    - Model development (UAP 6.5)
    - Evaluation (UAP 6.4)
    - Demonstration (Live) (UAP 6.5)

AP 6 Demonstration und Evaluation
UAP 6.1 Entwicklung einer Verarbeitungskette
UAP 6.2 Implementierung der Schnittstellen
UAP 6.3 Aufzeichnung und Anonymisierung von Daten
UAP 6.4 Evaluation der Assistenzfunktionen
UAP 6.5 Hardware-Implementierung

EVCS Dataset Generation: Overview

Planning

- EVCS Map
- On-Site Scouting
- Routes Planning

Recording

- May 24, 2024
- July 23, 2024

Annotation

- Anonymization
- EVCS Labelling

[VISCODA Dataset: Electric Vehicle Charging Station Detection](#)

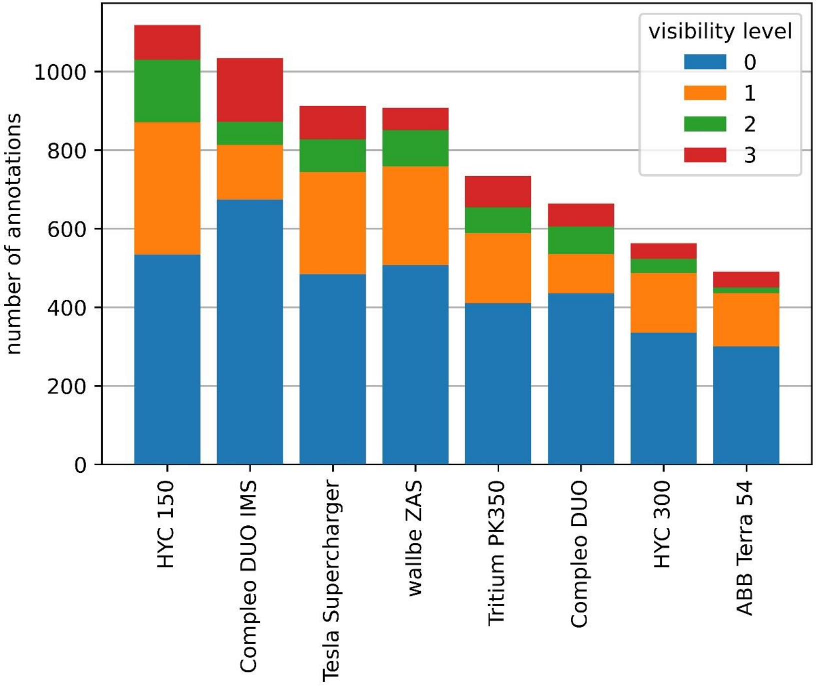
EVCS Detection Dataset: Major Types

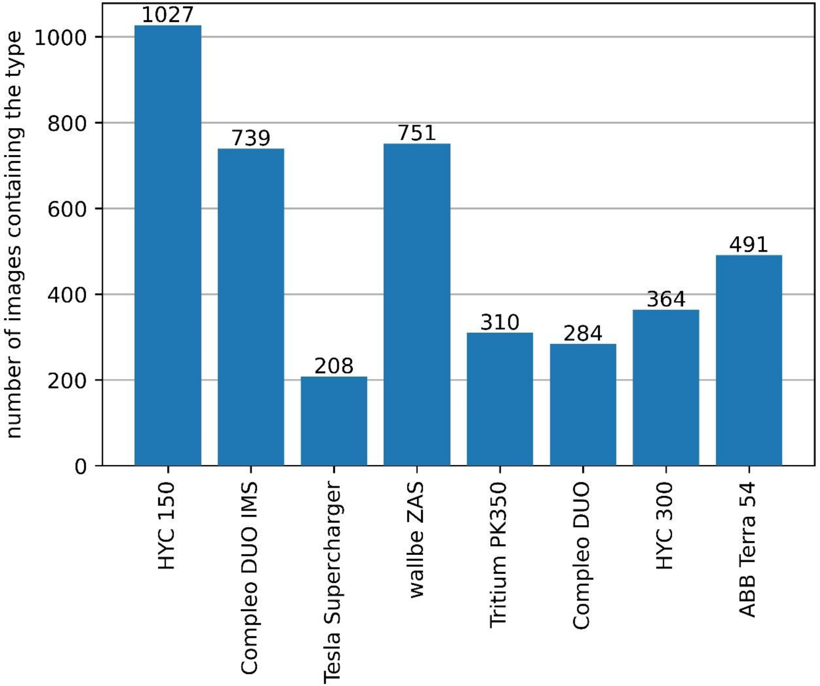
VISCODA

- Those types are popular in general

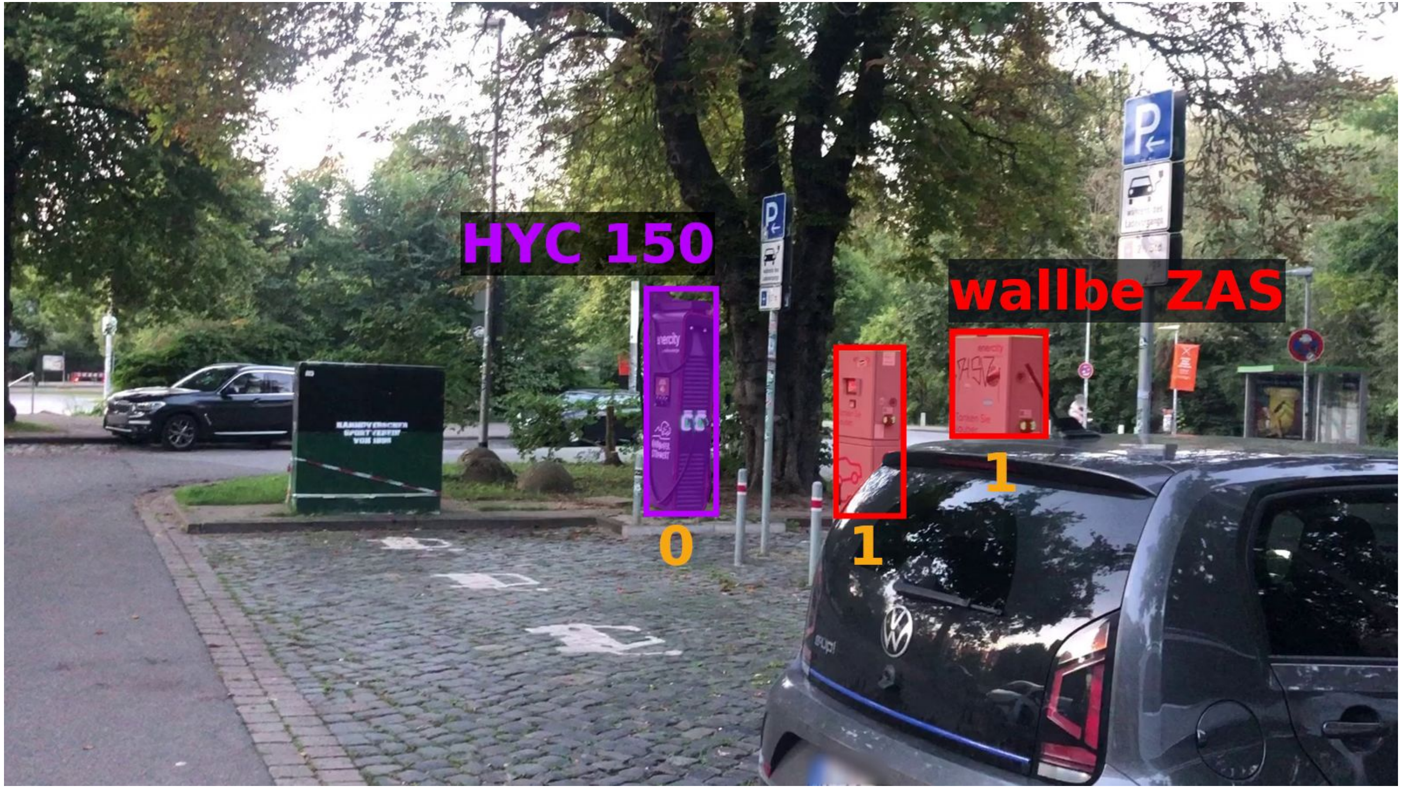
EVCS Detection Dataset: Major types

- Distribution of annotation per class: 400 – 1000





EVCS Detection Dataset



EVCS Detection Dataset

Evaluation

- Result for eight-class object detection

Detector	Backbone	
Faster-RCNN-R101	R101	65.2
YOLOX-l	CSP DarkNet Larger	63.0
DINO-L	Swin-L	69.7
Grounding-DINO-t	Swin-t	75.0
Grounding-DINO-ZS*	Swin-t	0.3

- Remarks:
  - All fine-tuned variants (first 4) delivered reasonable results
  - Zero-shot detection does not work for eight-class detection
    - However, it works to some extent for one-class detection (MAP 33.1)

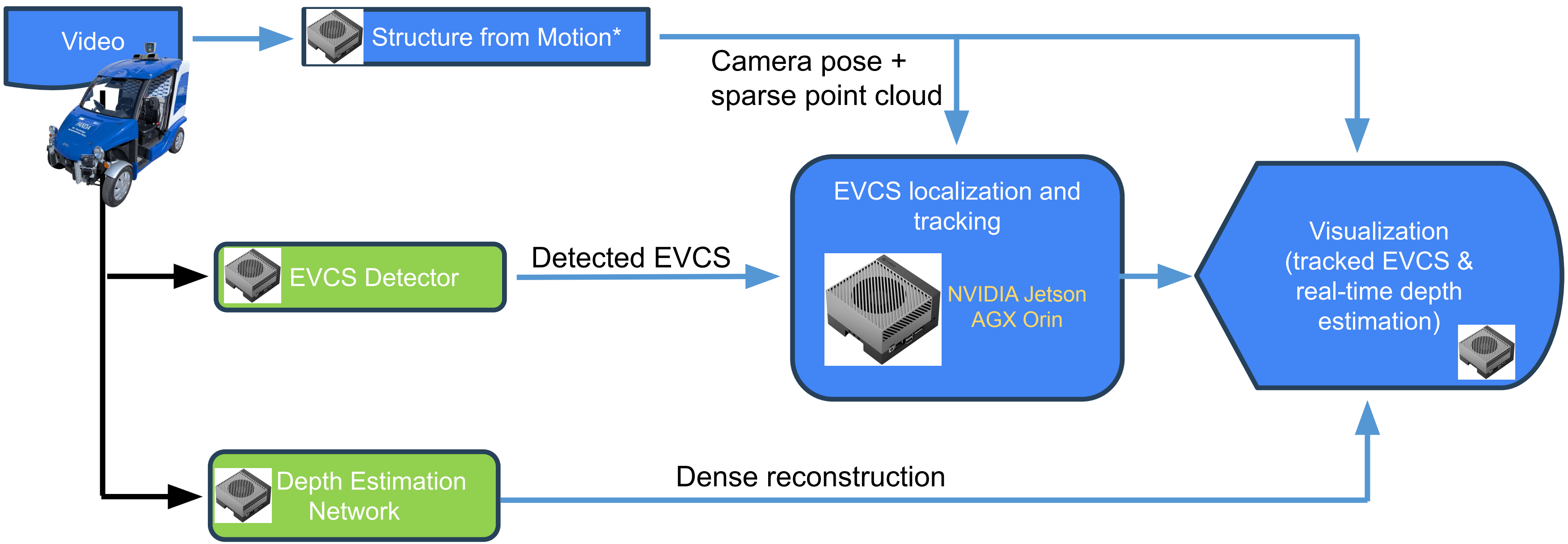
** The employed text prompts are the evcs class name of each type*

Data Usage

- Download the data here: www.evcs.viscoda.com
- Baseline detector development
  - Based on MMdetection framework
    - https://github.com/open-mmlab/mmdetection
  - Dataset readme: https://github.com/VISCODA-git/EVCS_Benchmark_Readme
    - Dataset details
    - Config files for baseline detectors
    - How to train and test baseline detectors
- Details are published via paper: https://doi.org/10.1007/978-3-032-12840-9_6

Chen, L., Südbeck, S., Riggers, C., Geib, T., Cordes, K., & Broszio, H. (2025, September). EVCS: A Benchmark for Fine-Grained Electric Vehicle Charging Station Detection. In *DAGM German Conference on Pattern Recognition* (pp. 73-88).

Demonstration Software: Overview



All the computation and visualization run on one single NVIDIA Jetson AGX Orin

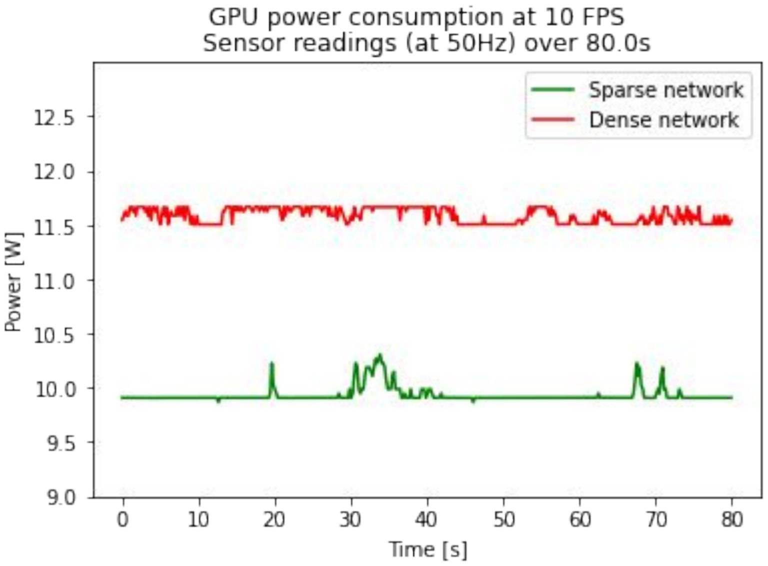
* The structure from motion runs in the online mode

Result: Detection Performance

EVCS Detector: Faster-RCNN-x101 trained with hypersparse loss (TNT, UAP 3.6)
Deploy version: Compiled with sparsity engine enabled on NVIDIA AGX Orin (TNT, VISCODA)

Model	fp32 (PyTorch)	fp16 (TensorRT)		int8 (TensorRT)		relative differences between int8 and fp16	
	mAP	mAP	time [ms]	mAP	time [ms]	mAP	time [ms]
Faster-RCNN-x101-Dense	63.2	63.3	111.3	63.9	70.3	+0.9%	-36.8%
Faster-RCNN-x101-Sparse	61.2	61.2	95.1	61.0	63.5	-0.3%	-33.2%
Relative differences between dense and sparse model	-3.3%	-3.3%	-14.6%	-4.5%	-9.7%		

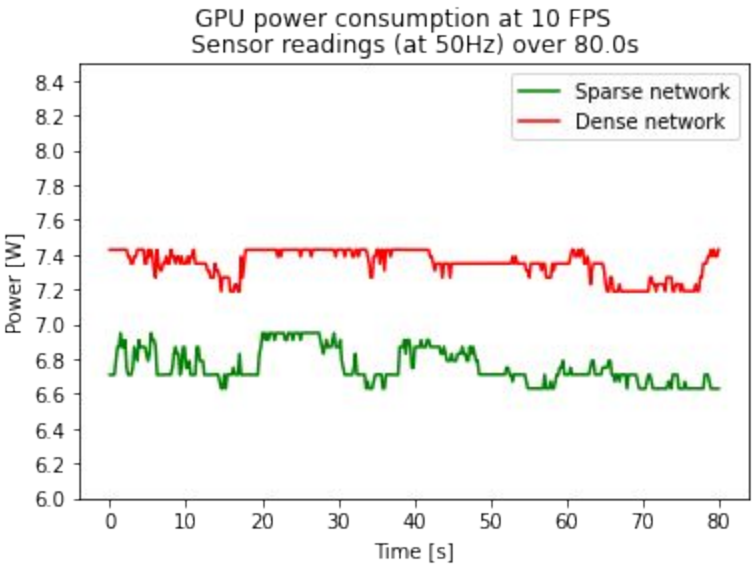
Result: Power consumption



Each data point corresponds to the average over 10 readings (200.0 ms)

FP16

Average power consumption dense network: 11.59 W
Average power consumption sparse network: 9.93 W
Absolute average power consumption reduction: 1.66 W
Relative average power consumption reduction: **14.3%**



Each data point corresponds to the average over 10 readings (200.0 ms)

INT8

Average power consumption dense network: 7.35 W
Average power consumption sparse network: 6.76 W
Absolute average power consumption reduction: 0.59 W
Relative average power consumption reduction: **8.0%**

Single-view Depth Estimation

- A class agnostic object detection is prefered
  - A depth estimation network lays a good foundation

- The state-of-the-art single-view depth estimation network delivers impressive results with a large network
  - Can we make it work on an edge hardware via:
    - Distilling the knowledge from the larger network
  - How to design and development such a network? -> AutoML!

Setup

- Training data: 5K images, EVCS + traffic scenarios
- Ground-Truth: Generated by Depth-Anything 3 (Lin et al., 2025)
- Architecture:
    - MobileNet based backbone (known for good convertablity)
    - Encoder decoder architecture
    - Searching range: 12 hyperparameters including architecture, loss type, learning rate etc.
- Multi Objective AutoDL (UAP 2.2)
    - Accuracy, FLOPS, Energy
    - SMAC hyperparameter optimization
        - ParGEO for multi-objective handling
        - Hyperband for multi-fidelity training (Li et al., 2018)
        - Budget based optimization

Ground Truth Example

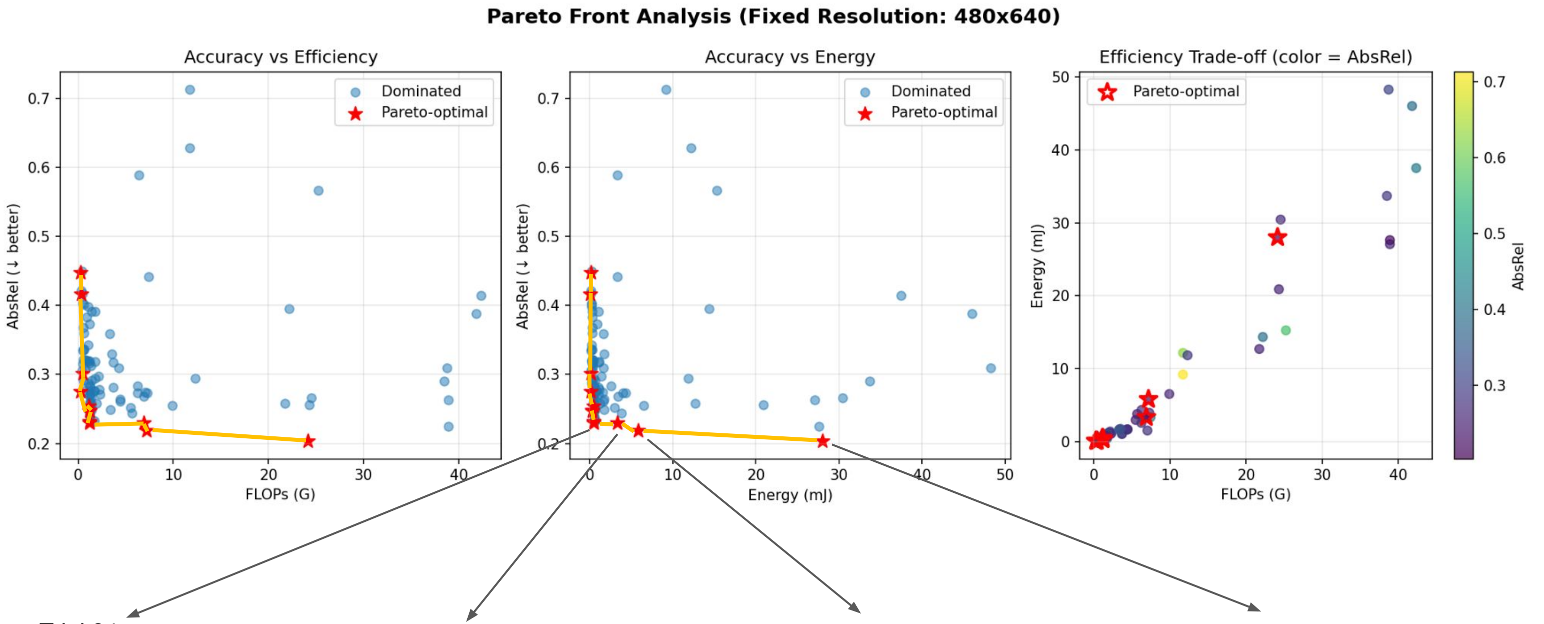
Darker: more distant
Brighter: closer

Lin, H., Chen, S., Liew, J., Chen, D. Y., Li, Z., Shi, G., ... & Kang, B. (2025). Depth anything 3: Recovering the visual space from any views. arXiv preprint arXiv:2511.10647.

Theodorakopoulos D, Stahl F, Lindauer M. Hyperparameter Importance Analysis for Multi-Objective AutoML[M]//ECAI 2024. IOS Press, 2024: 1100-1107.

Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A., & Talwalkar, A. (2018). Hyperband: A novel bandit-based approach to hyperparameter optimization. *Journal of Machine Learning Research*, *2018*(185), 1-52.

Pareto Front



Trial 84:
AbsRel: 0.2301
FLOPs: 1.19G
Energy: 0.47mJ
Config: width_mult=1.23,
decoder_ch=96,
stages=4

Trial 76:
AbsRel: 0.2298
FLOPs: 6.89G
Energy: 3.34mJ
Config: width_mult=1.16,
decoder_ch=128,
stages=5

Trial 71:
AbsRel: 0.2182
FLOPs: 7.18G
Energy: 5.80mJ
Config: width_mult=1.06,
decoder_ch=128,
stages=5

Trial 48:
AbsRel: 0.2032
FLOPs: 24.14G
Energy: 27.99mJ
Config: width_mult=0.72,
decoder_ch=96,
stages=5

Energy Consumption

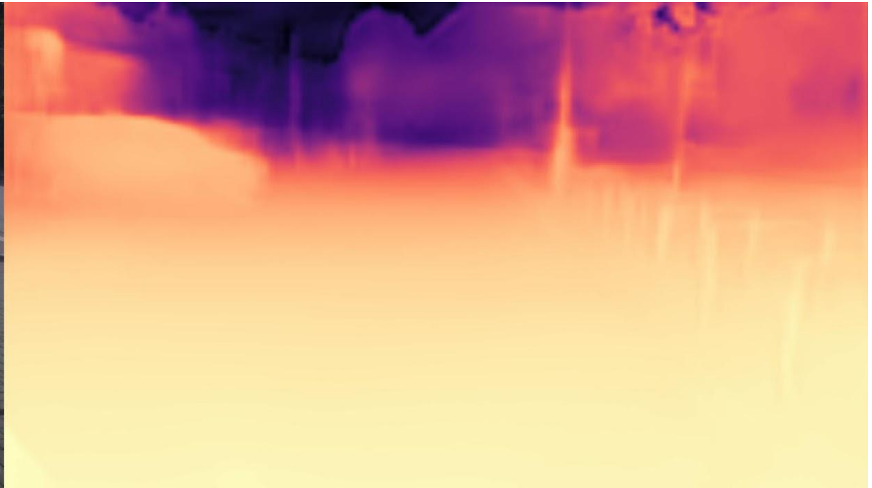
Trial	Platform	Precision	Throughput (FPS)	Energy/Frame (mJ)	Energy saving (from Trial_71 to Trial_84)
Trial 71	Jetson Nano	FP16	11.4	1181.9	**83.9%**
Trial 84	Jetson Nano	FP16	74.7	190.3	
Trial 71	Jetson AGX Orin	FP16	336.0	195.2	**30.5%**
Trial 84	Jetson AGX Orin	FP16	1396.0	135.7	

- Successfully compiled on multiple edge devices
  - NVIDIA Jetson Nano
  - NVIDIA Jetson AGX Orin
  - Qualcomm QCS 6490 (rb3 gen2 vision kit)

Result Image

Input Image

Estimated Depth



Darker: more distant
Brighter: closer

Demonstration

- Data: Live camera data from PANDA

- VISCODA GreenAutoML4FAS demonstration software
  - Pruned EVCS Detector (UAP 3)
  - AutoML (SMAC) trained single-view-depth estimation network (UAP 2)
  - Optimized structure from motion (VISCODA)
  - Detection, localization and tracking algorithm (VISCODA)
  - Visualization (VISCODA)