mhv/2026
ACM MILE HIGH VIDEO
Leading Video Coding and Streaming Event Since 2016

Enabling Resource-Efficient AI Video Analytics with Hierarchical Codecs

Guendalina Cobianchi, V-Nova Ltd,
Manzoor Razaak, V-Nova Ltd,
Ben Cope, Intel Corp.
Palanivel Guruvareddiar, Intel Corp.

The Disconnect Between Video Formats and AI Requirements

Human Consumption

Aimed at perceptual fidelity

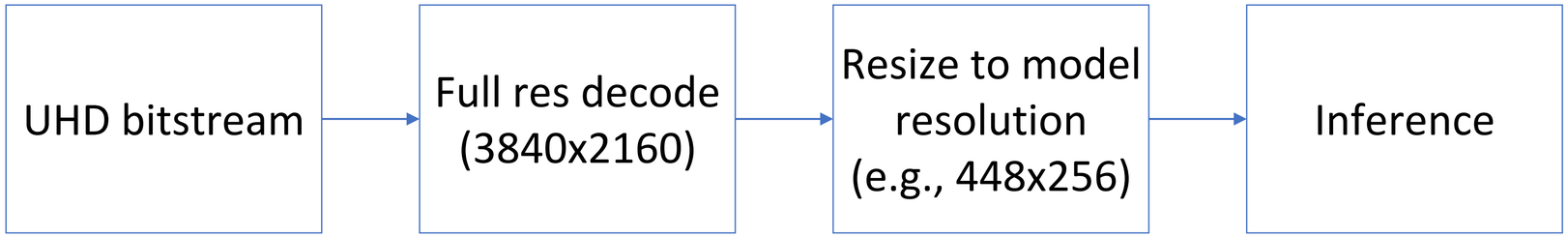
e.g., full resolution, 60 fps, 4:2:0 color

Machine Vision

Aimed at Feature Extraction

e.g., low resolution, sparse access, ROI focus

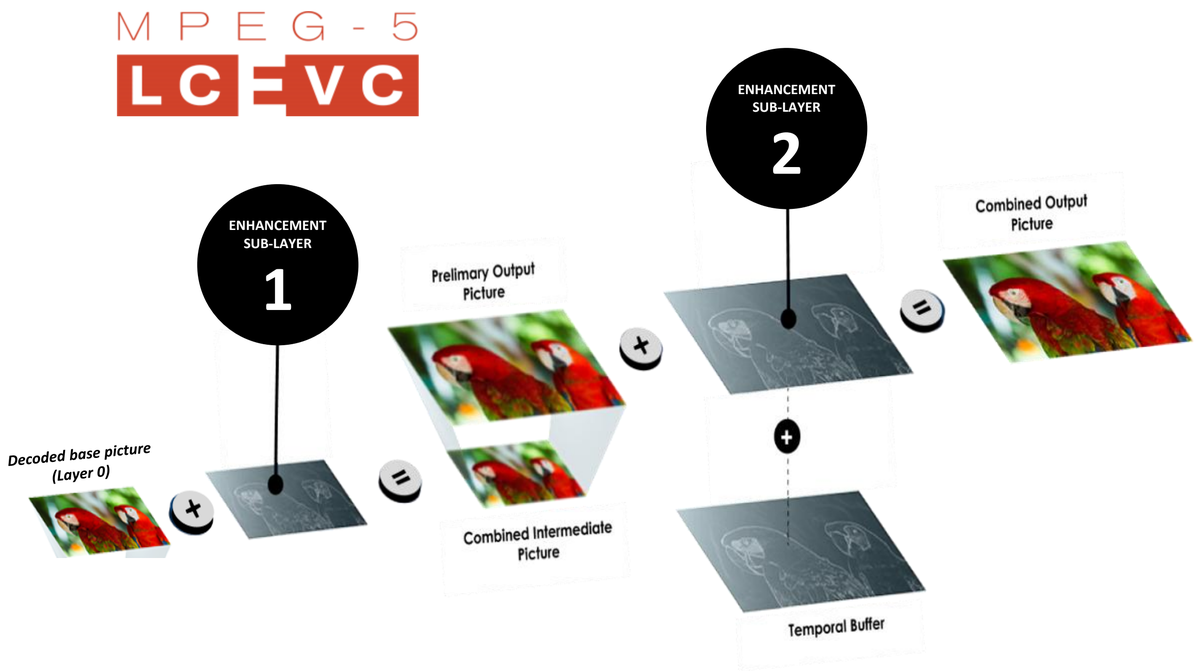
Conventional decoding imposes a full-resolution preprocessing overhead



- Typical analytics pipeline: full-frame decode → color convert / scale → inference
- Inference operates on reduced representations (downscaled inputs and/or sparse frame sampling)
- Most decoded pixels are discarded, yet still incur decode, resize, and memory traffic
- Example: 3840×2160 → 448×256 uses **~1.4% of pixels** (the remainder is discarded after resizing)

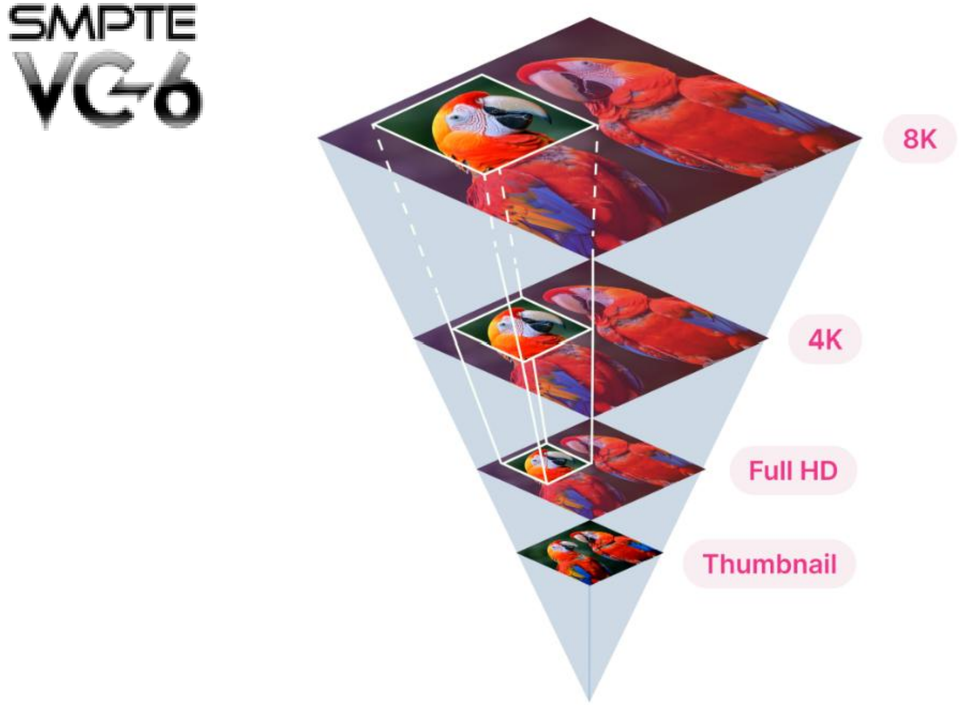
This mismatch shifts bottlenecks to decode/scale and memory movement, leading to **accelerator under-utilisation** when inference cannot be fed at line rate

Hierarchical Formats Enable 'Compute-Aware' Data Access



MPEG-5 LCEVC (ISO/IEC 23094-2)

Base-First Decoding (HD) for AI, selectively pull (UHD) if needed



VC-6 (SMPTE ST 2117)

Hierarchical intra-structure allows fetching specific resolution or ROIs

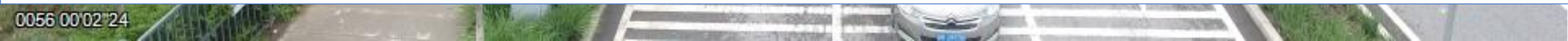
3840x2160, 622 KB

A practical VC-6 example – UHD (0.6 BPP)

960x540

226 KB (-64%)

LOQ2, 540p



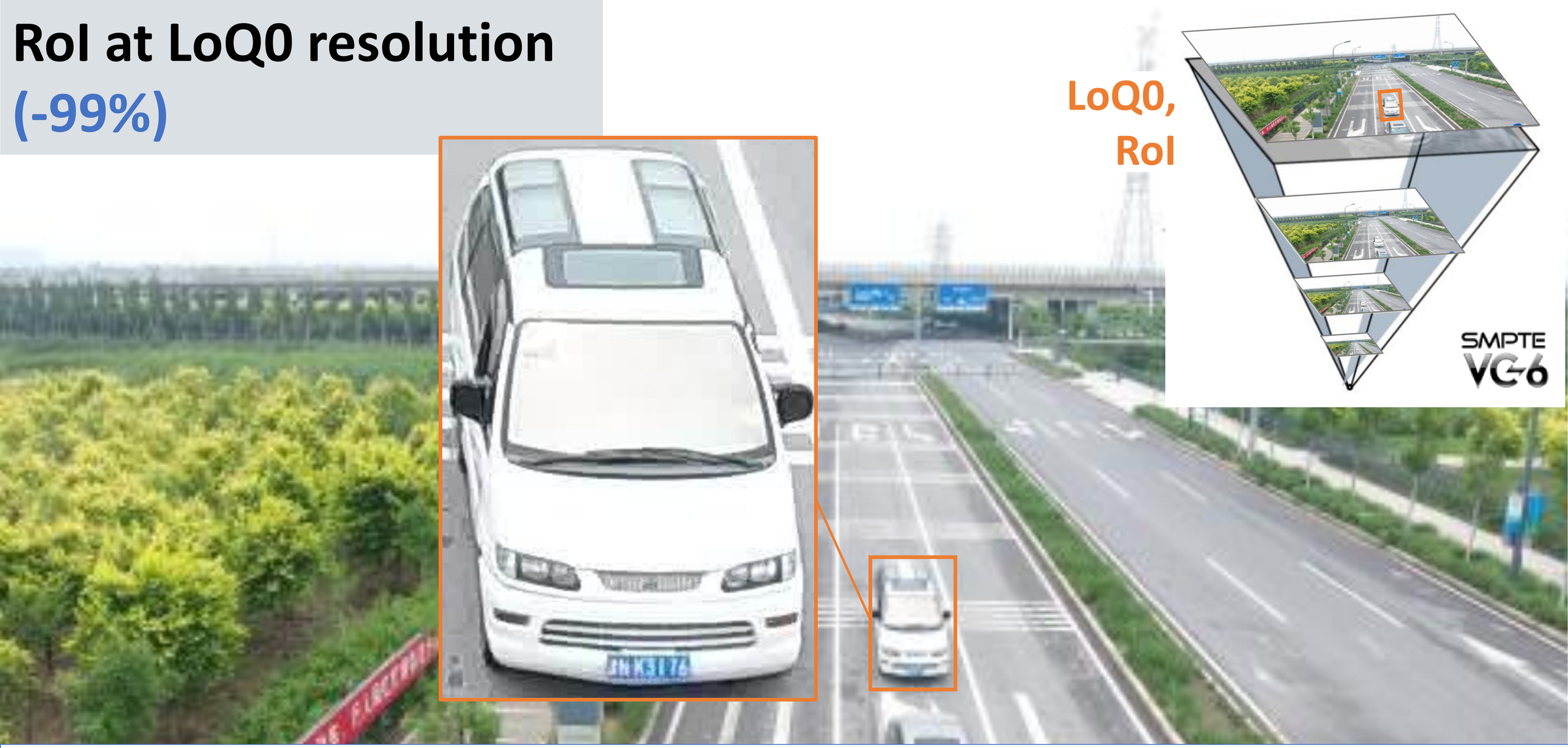
480x270

53 KB (-91%)

LOQ3, 270p

RoI at LoQ0 resolution

(-99%)

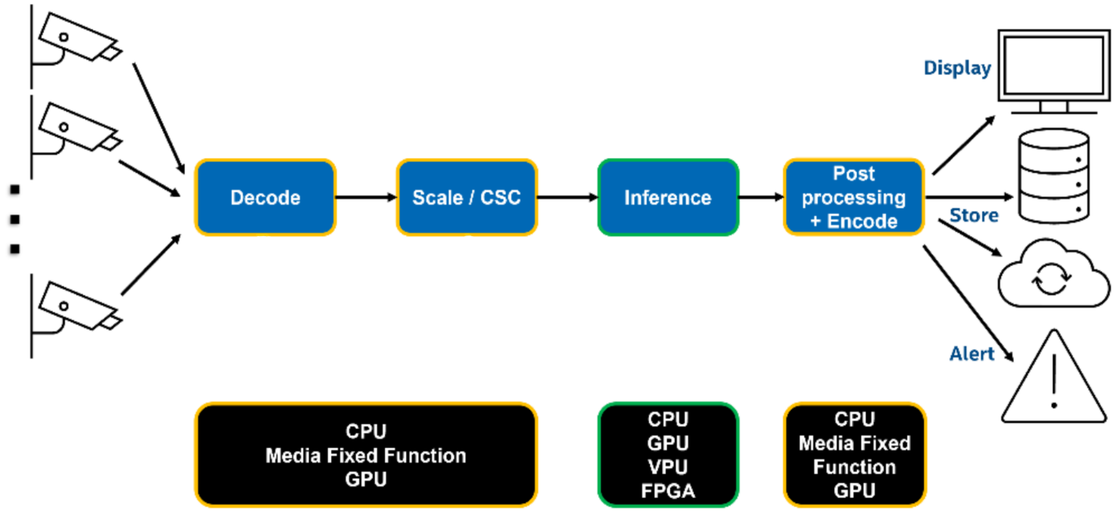


LoQ0,
RoI

Decode & fetch a specific region, LoQ0

Methodology:
Benchmarking on Edge Silicon: Intel® Core™ Ultra 7

Full pipeline



Specs:

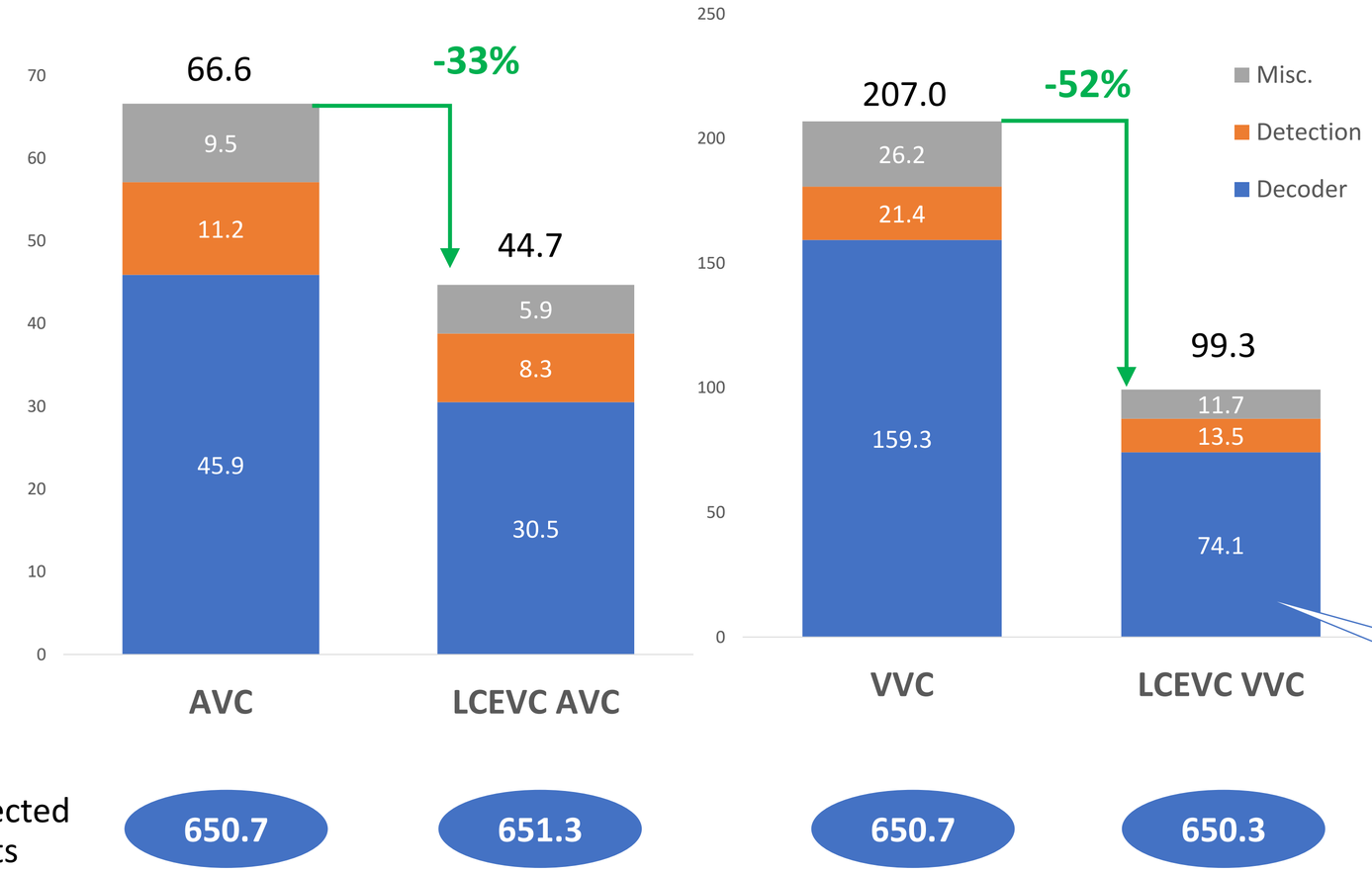
- **Framework**: Intel's Deep Learning Streamer (DL Streamer), an open-source streaming media analytics framework
- **Hardware**: Intel® Core™ Ultra 7 258V (Lunar Lake) w/ Arc 140V GPU. Leveraging hardened AVC decoders.
- **Dataset**: VisDrone DET2018 (UHD Traffic sequences)
- **Model**: SSD MobileNet V2 for obj. detection (448x256 input)
- Codecs: AVC/ VVC/ AVC intra vs LCEVC/ VC-6
- **Evaluation focuses on system-level latency and throughput using standardized, reproducible hardware and software stacks without impacting inference outputs.**

https://docs.openedgeplatform.intel.com/dev/edge-ai-libraries/dlstreamer/index.html

Use case 1: LCEVC Acceleration Results - CPU

Total execution time comparison on CPU (sec.)

AVC/VVC vs. LCEVC AVC/VVC CPU time comparison



# detected objects	650.7	651.3	650.7	650.3

Key insights

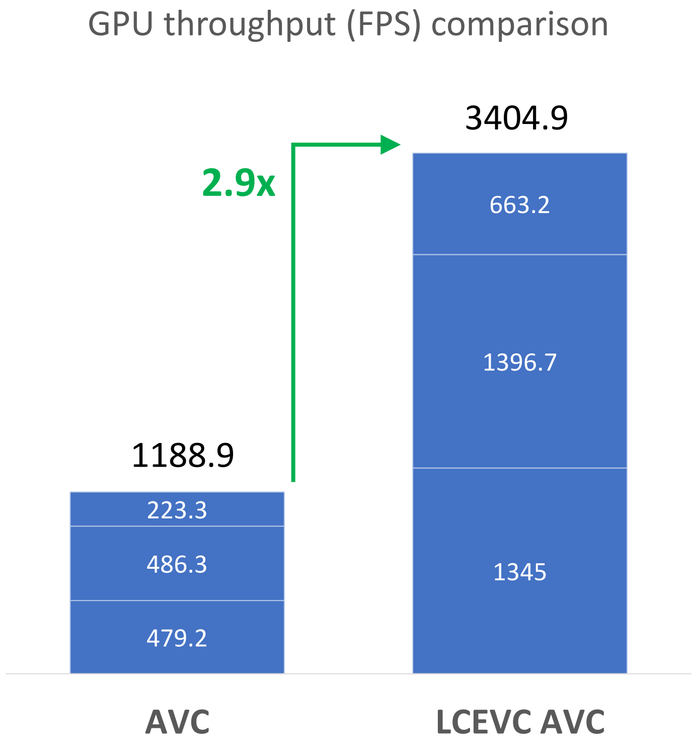
- LCEVC reduced CPU compute time:
  - by 33% for AVC
  - By 52% for VVC
- Performance gains in decoding, scaling, and detection (all stages).
- Accuracy remained consistent across codecs

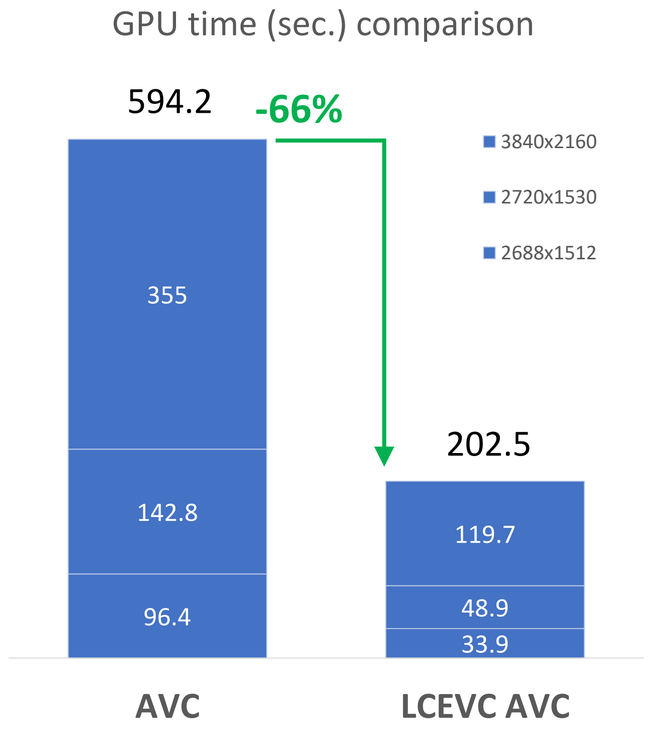
By decoding only the base layer, we remove the most expensive parts of the software decode pipeline

Use case 1: LCEVC Acceleration Results - GPU

Throughput and GPU time comparison

AVC vs. LCEVC AVC (hardened AVC codec support)





Key insights

- LCEVC-AVC achieves faster pipeline execution compared to native AVC:
  - **2.9x** higher throughput
  - **66%** overall time reduction
- GPU decode load decreases ~4x, inference accelerated thanks to lighter scaling to model resolution
- Smaller base layer (~75-80% of total) also reduces memory bandwidth and data transfer
- **Higher channel density, enabling more concurrent AI inference streams on the same infrastructure**

Impact of Base-Layer Decoding on Detector Outputs (VisDrone)

Example frames at identical detection threshold*

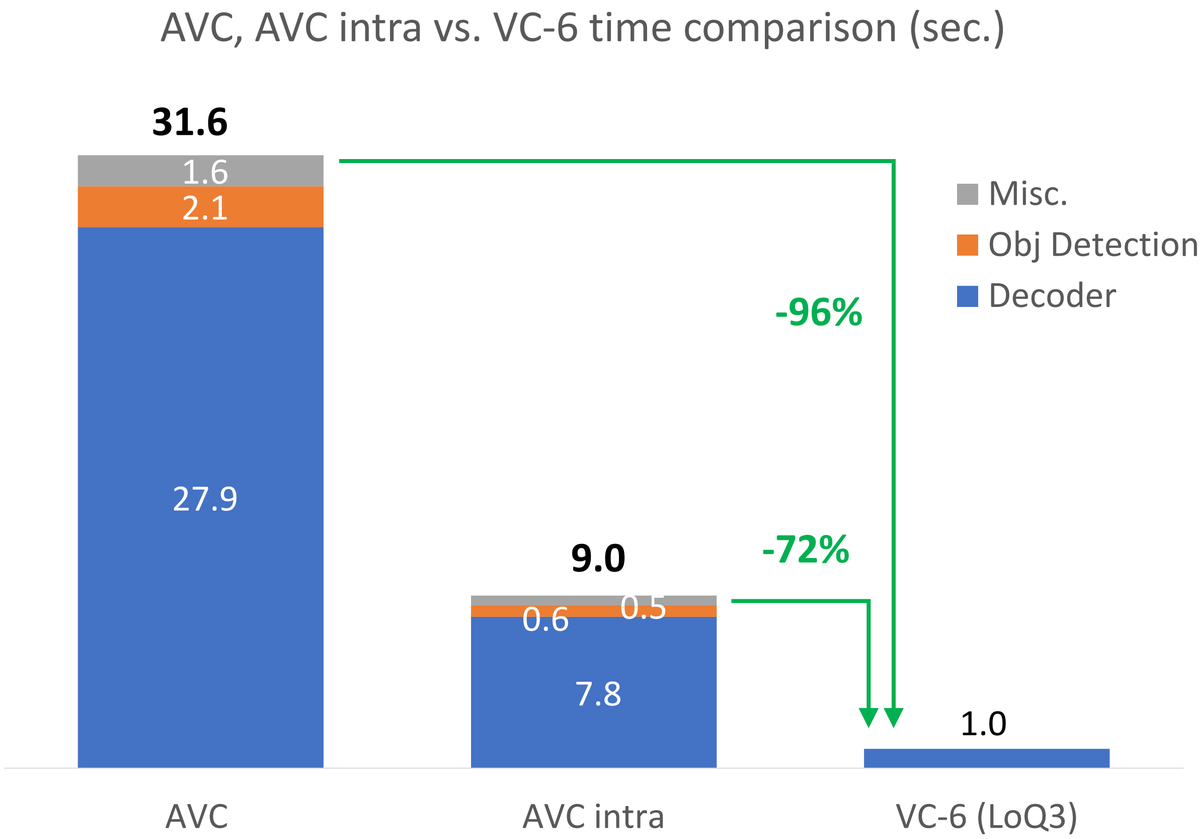
Configuration	Precision	Avg. object count
Native VVC (UHD)	98.6%	650.7
LCEVC Base (HD)	98.6%	651.3

- **Detector outputs remain consistent** when operating on the LCEVC base layer at the target inference resolution
- **Both inputs are downscaled to 448×256** before inference; the comparison therefore reflects the effect of decoding full-resolution frames versus decoding only the base layer before scaling

Note: model used for inference on unseen data to assess system-level performance. Inference outputs were monitored to ensure consistency across codecs; task-level accuracy benchmarking requiring ground-truth annotations was outside the scope.

* Visualization at display quality, as outputted by DL Streamer

Use case 2: SMPTE VC-6, Sparse Inference



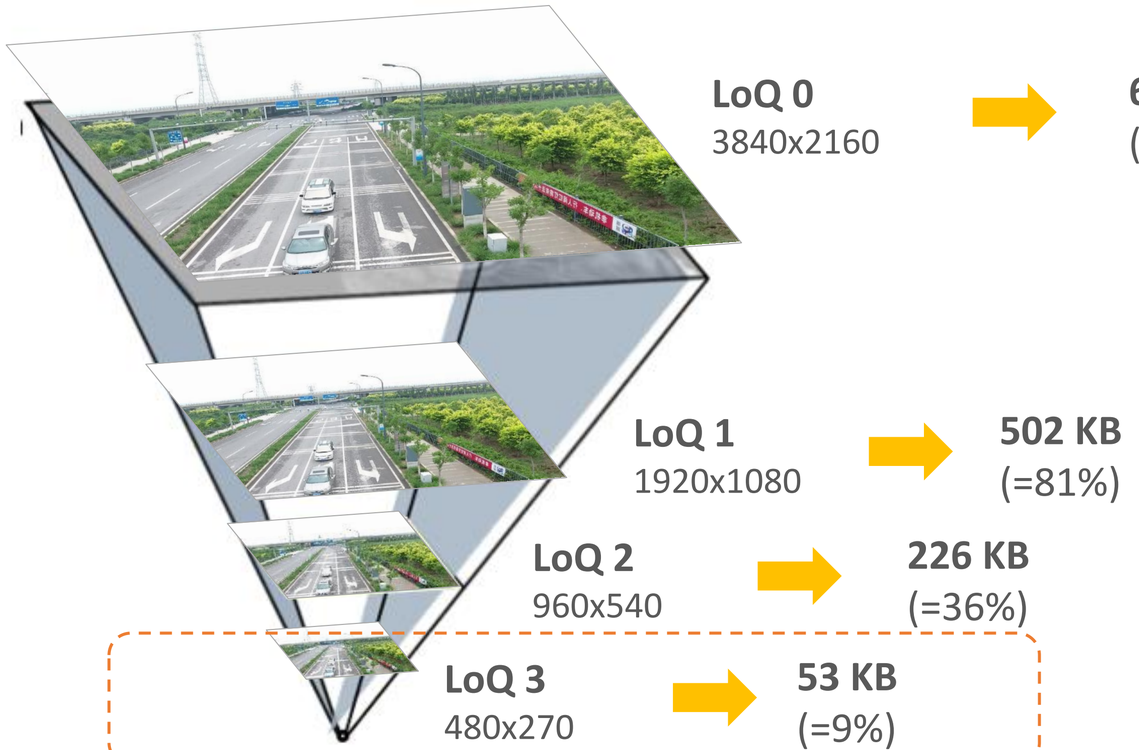
Key insights

- **Selective decoding for sparse inference (1 frame/ second)**: VC-6 decodes only the required LoQ and sampled frames.
- **Large pipeline time reduction**:
  - **-72%** vs AVC intra (1-in-10 frame sampling)
  - **-96%** vs AVC inter.
- **Savings dominated by decode and I/O**: Improvements come from avoiding unnecessary decode and memory movement, not model execution.

Use case 2: Data Transfer comparisons

VC-6 LoQ Size and Data Transfer Savings

0.6 bpp example, Visdrone clip



SSD MobileNet V2 requires 448x256 resolution

Data Transfer requirements

- **AVC inter** → **100% transfer:** decodes and transmits all frames, *4x smaller file*
- **AVC intra** → **10% transfer:** samples 1/10 frames, transfers them at full resolution
- **SMPTE VC-6** → **0.9% transfer:** transmits only the **LoQ3 layer** (**8.6%**) on 1/10th of the frames

VC-6 selective access and partial fetching allows to achieve:

- **>30x** I/O savings vs AVC
- **>10x** I/O savings vs AVC intra

Conclusion & future work

Conclusions

- **Data format choice** is a system-level optimisation lever for video analytics, not only a storage decision.
- **LCEVC (retrofit path):** base-layer decoding reduces CPU time by **33% - 52%** and improves GPU throughput by **2.9x,** with consistent detector outputs.
- **VC-6 (selective path):** selective LoQ and frame access reduces processing time by **up to 96%** and enables order-of-magnitude I/O reductions through partial transfer

Limitations and next steps

- Evaluation focuses on system-level latency and throughput; ground-truth accuracy metrics (e.g. recall, F1) were not reported.
- **Future work**: extend to labelled datasets, additional tasks (e.g. tracking, segmentation), and GPU-accelerated VVC decoding for direct comparison with AVC.

System-level optimisation lever:

"from file to API"

- Decode what inference needs (resolution, sampling, ROI)
- Higher throughput and lower latency
- Reduced data transfer and I/O
- No changes to model or hardware