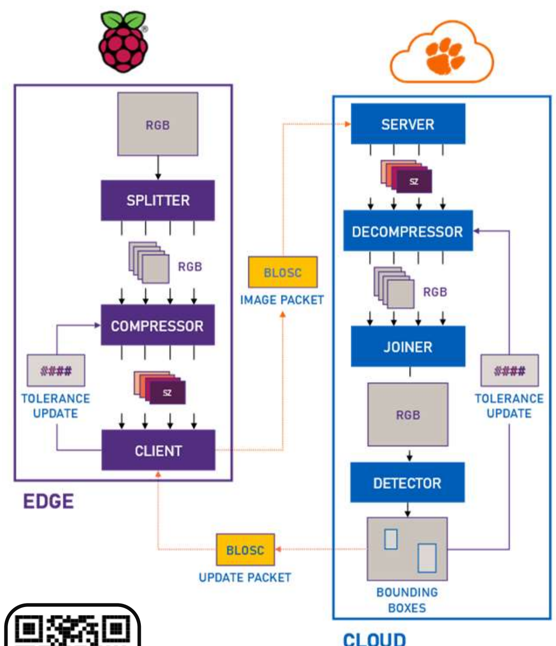
PRACTICAL EVALUATION OF REMOTE ADAPTIVE LOSSY COMPRESSION WITH FEEDBACK

Author: Connor Ennis
Advisor: Dr. Jon C. Calhoun
Email: cjennis@clemson.edu
Github: github.com/ckinvents

ABSTRACT

- When performing object detection, image data is often transmitted to a more powerful cloud system for processing over low bandwidth connections.
- Previously, we considered a simulated architecture for adaptive lossy compression, prioritizing quality in regions where detection occurs.
- **To validate and refine this model for practical use, we propose a revised implementation split between real edge and cloud hardware.**

ARCHITECTURE

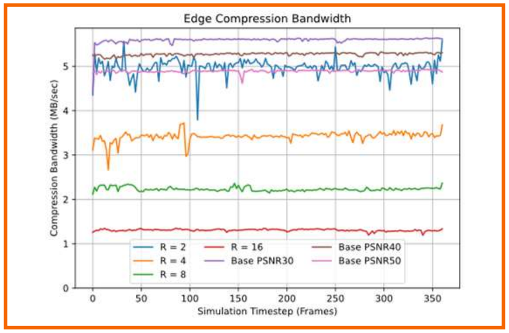


Additional information and data

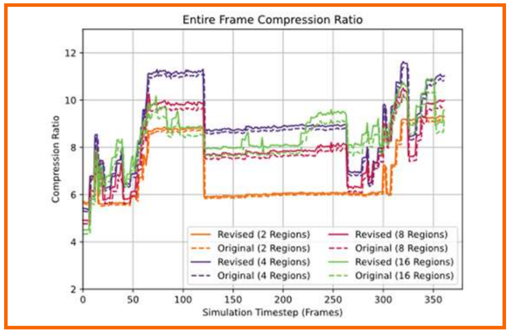
CONTRIBUTIONS

- Revised implementation runs partly on a **low-power device** with a different architecture (ARM vs. x86)
- Consequentially, **client-server communication** can no longer be handled internally using MPI.
- We determine the new **compression bandwidth** (running on the Raspberry Pi) and **communication performance** over a network protocol (in our case, TCP over SOCKS5)
- Additionally, we compare the **performance of a single lossless pass** to that of SZ's internal per-array stage.

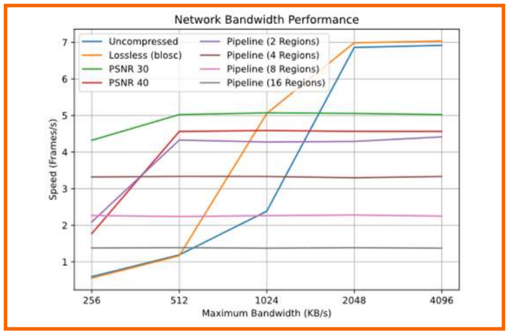
RESULTS



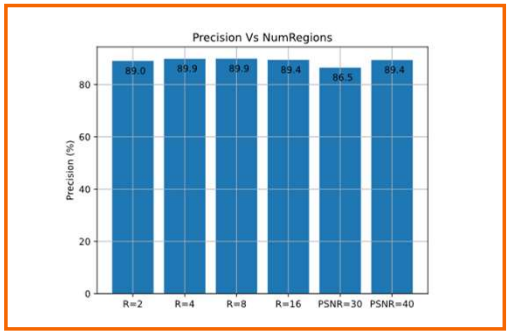
The Pi only has 4 cores, so increasing the number of splits has a much greater performance penalty than on Palmetto (splits become serialized).



By applying the lossless step to the entire frame, rather than individual splits, we achieve improved ratios and performance than with a default SZ configuration.



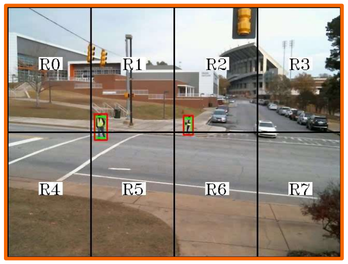
Adaptive compression primarily performs worse than fixed PSNR compression on the Pi. At 256 KB/s, however, there is some framerate improvement for 4 and 8 splits above a fixed PSNR of 40.



To demonstrate that our implementation represents the source model, the detection performance must compare with the simulated results.

RELEVANCE

- Cloud-edge architectures allow for **cost-effective allocation** of HPC resources to remote locations.
- Verifying and improving **real-time data compression** for an actual edge device demonstrates how our model performs **in the field**.
- By identifying bottlenecks and limitations imposed by real communication and hardware, we **learn what limitations exist and how they may be overcome**.
- The task of pedestrian detection provides a practical scenario **invaluable to transportation safety**.



Splits and bounding boxes for adaptive lossy compression (8 regions)

CONCLUSIONS

- As-is, adaptive lossy compression is **generally less performant** on a low-power Arm device than fixed PSNR and lossless compression, **despite comparable accuracy**.
- Based on the reduction in bandwidth as splits increase and the minimal impact of network throttling on SZ-compressed data, **the largest limiting factors** are likely **processing overhead and core count** for parallelization.
- For split SZ compression, applying the lossless pass after joining offers a **notable improvement in compression ratio**, increased as the number of splits rises.

Clemson University is acknowledged for generous allotment of compute time on the Palmetto cluster.
This material is based upon work supported by the National Science Foundation under Grant No. SHF-1910197 and SHF-1943114.