Real-time AI-Integrated Security Camera System

A Video Surveillance as a Service approach

Trung Nguyen Tran Anh Kent Bui Cong Khoi Tran

Instructor: Trong Nhan Le, PhD

Reviewer: Duc Dung Nguyen, PhD

Faculty of Computer Science and Engineering
Ho Chi Minh University of Technology, Vietnam National University

June 05, 2024

Table of Contents

1. Introduction
   - Related Works
   - Objectives
2. Architecture
   - Camera Cluster
   - Local Transcoder Device
   - Computer Vision Service
   - Cloud Services
3. Demonstration
4. Evaluation
   - Stream Latency
   - Bandwidth Usage
   - Stress Test
5. Conclusion
   - Achievements
   - Limitations
   - Future Developments

Table of Contents

1 Introduction
- Related Works
- Objectives

2 Architecture
- Camera Cluster
- Local Transcoder Device
- Computer Vision Service
- Cloud Services

3 Demonstration

4 Evaluation
- Stream Latency
- Bandwidth Usage
- Stress Test

5 Conclusion
- Achievements
- Limitations
- Future Developments

Introduction to Video Surveillance as a Service

Video Surveillance as a Service (VSaaS) is a solution that offers remote management, storage, live streaming, analytics from the cloud

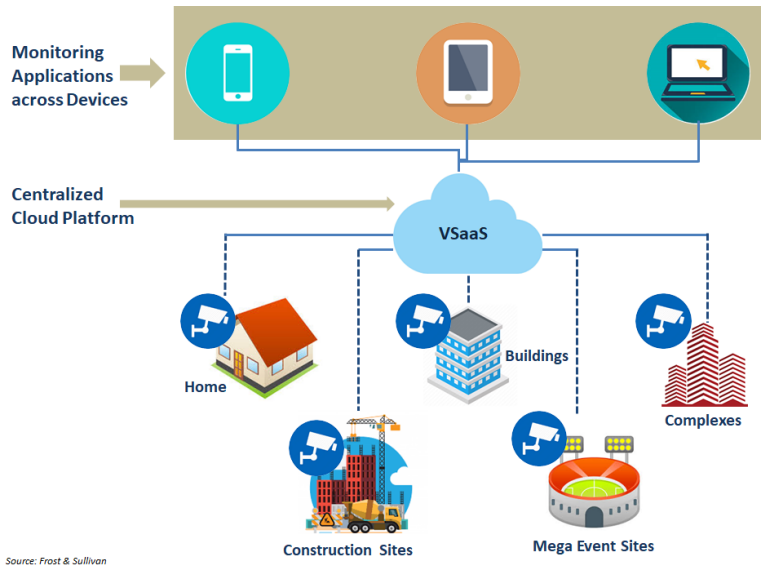
How does it work?

Cameras are connected to the cloud via the Internet

Benefits

VSaaS offers several benefits over traditional self-hosted surveillance systems:

1. Low initial capital cost
2. Secure and reliable
3. Advanced feature set due to the availability of cloud computing
4. Reduced IT staffs



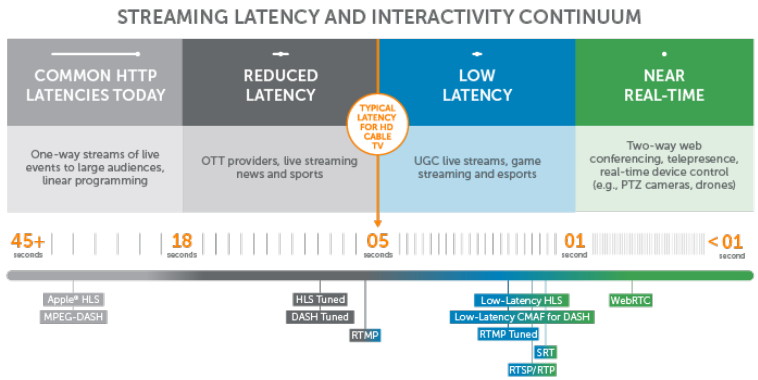
Case study

Highlands Coffee uses VCloudcam (VNG) to centrally manage over 1,500 cameras across 330 locations, monitoring both customers and employees

Pharmacity uses VSaaS to record customer behavior and collect metrics like foot traffic and conversion rates across hundreds of locations

Streaming Protocols

In the last decade, the defacto streaming protocols for transporting live video feeds over the Internet are RTMP (Real-Time Messaging Protocol) and RTSP (Real-time Streaming Protocol)



Limitations

Lack of universal standard for encryption and are more primitive compared to newer protocols

Streaming Protocols

Approach

Favor next-gen protocols over older standards

- Secure Reliable Transport (SRT): UDP-based, encryption comes standard, proven to offer low latency for long distance and low bandwidth[1]scenario due to superior congestion control
- WebRTC: UDP-based, peer-to-peer, designed for low-latency video calls and streaming over the Internet, developed by Google

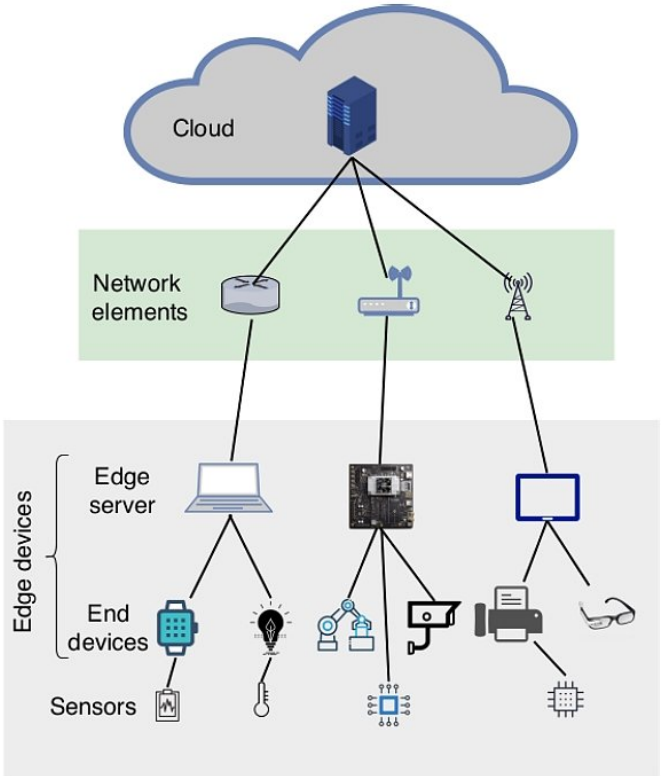
[3]Haivision (n.d.), *RTMP vs. SRT: Comparing Latency and Maximum Bandwidth [Whitepaper]. https://www.haivision.com/resources/white-paper/srt-versus-rtmp/*

Edge AI

Edge AI refers to the combination of edge computing and artificial intelligence to execute machine learning tasks directly on interconnected edge devices [2]

Benefits

- Reduced latency, bandwidth usage and increase real-time responsiveness
- Resilience to Internet connectivity issues
- Reduce the load on centralized cloud processors



[4] IBM (n.d), *What is edge AI?*. https://www.ibm.com/topics/edge-ai

Objectives

The objective is to improve the existing Video Surveillance as a Service model in...

- **Connectivity**: Using new and improved streaming protocols to increase reliability and security between cameras and the cloud
- **Scalability**: Implementing an architecture based on the use of Edge AI to decentralize analytic features
- **Security**: Avoid exposing cameras to the public Internet, reducing risks from outdated firmware vulnerabilities

Table of Contents

1 Introduction
- Related Works
- Objectives

2 Architecture
- Camera Cluster
- Local Transcoder Device
- Computer Vision Service
- Cloud Services

3 Demonstration

4 Evaluation
- Stream Latency
- Bandwidth Usage
- Stress Test

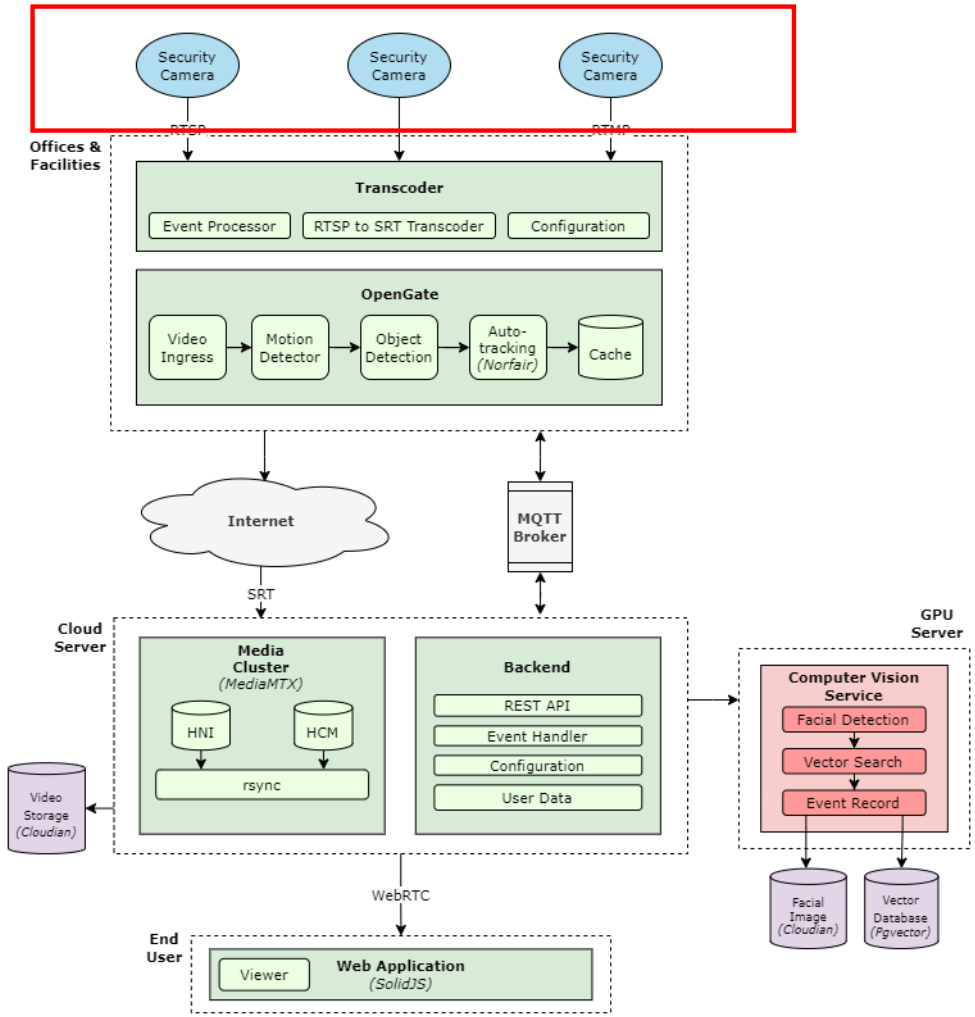
5 Conclusion
- Achievements
- Limitations
- Future Developments

Camera Cluster

Definition

A camera cluster includes 1 or more IP cameras that shares the same network.

- The demo featured Hikvision PTZ cameras
- Hikvision cameras can be accessed through its internal network portal & integration APIs

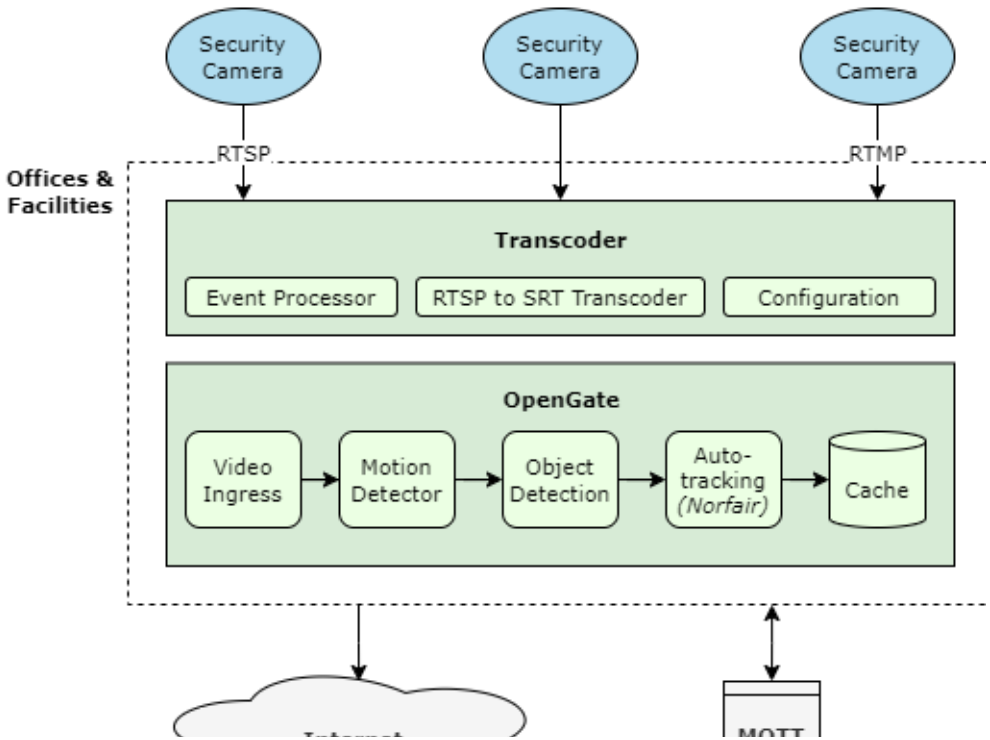


Camera Cluster

Definition

A camera cluster includes 1 or more IP cameras that shares the same network.

- The demo featured Hikvision PTZ cameras
- Hikvision cameras can be accessed through its internal network portal & integration APIs

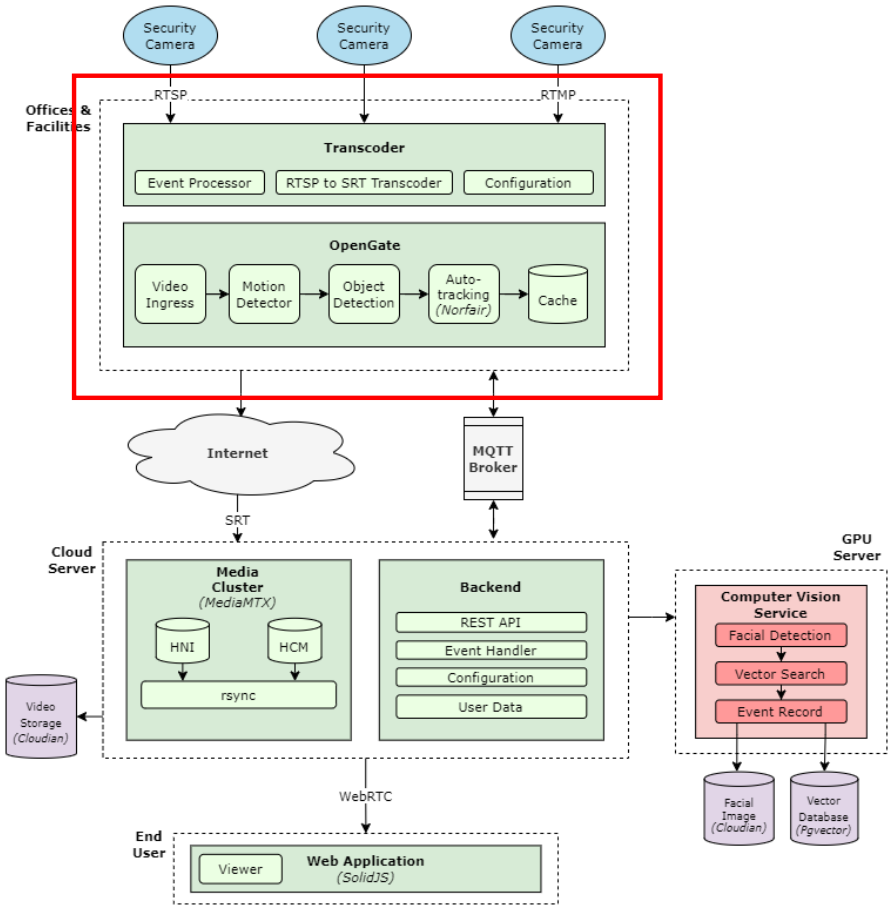


Local Transcoder Device (LTD)

Definition

A low-powered server sits in the same network with the cameras, act as an edge computing node. It includes 2 components: Transcoder & OpenGate

- Demo & testing use a laptop equivalent to an Intel NUC

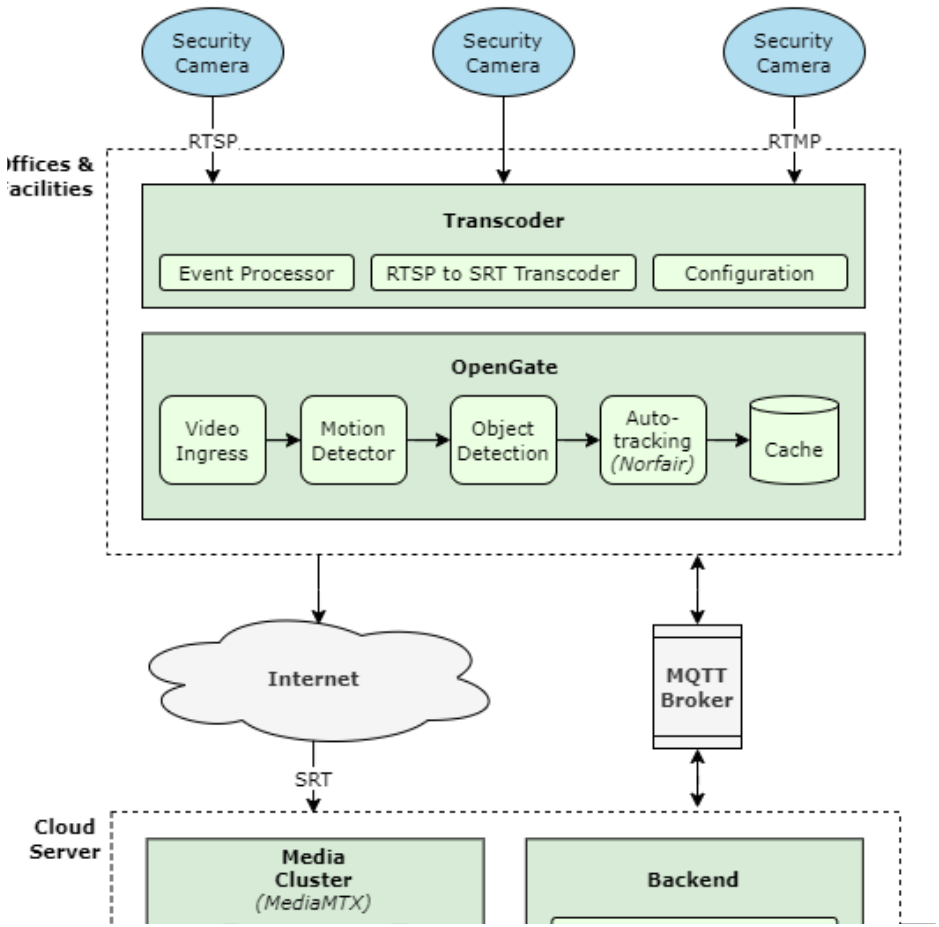


Local Transcoder Device (LTD)

Definition

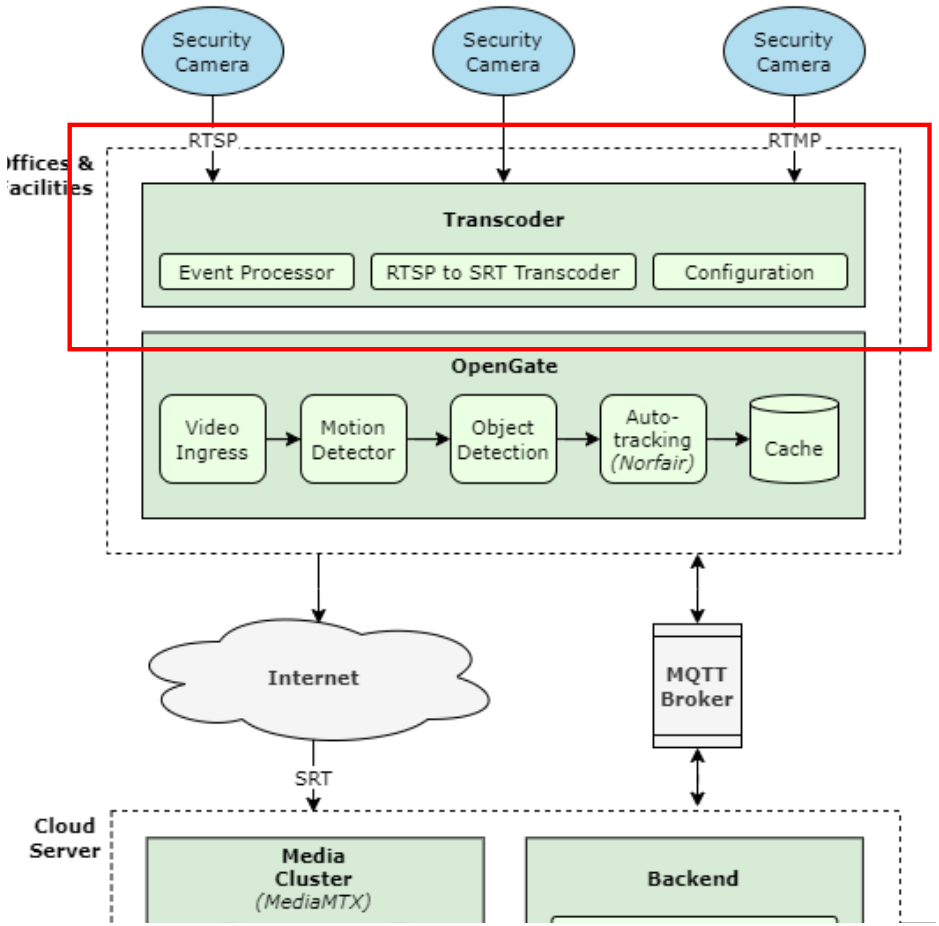
A low-powered server sits in the same network with the cameras, act as an edge computing node. It includes 2 components: Transcoder & OpenGate

- Demo & testing use a laptop equivalent to an Intel NUC



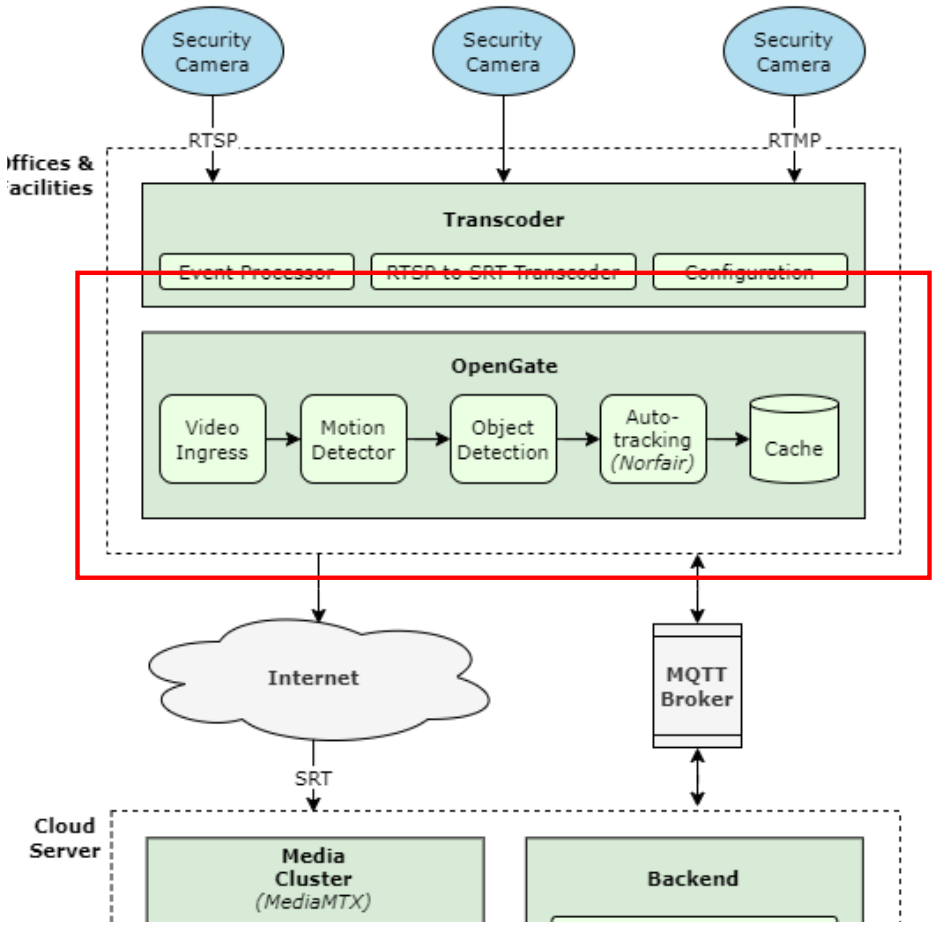
Transcoder

- Transcode insecure RTSP streams to low-latency, reliable SRT for further distance streaming to the cloud
- Enables remote control and configuration of the cameras and collect their metrics via MQTT messages



OpenGate

- Perform analytic on video streams
- Features: Motion detection; Object detection & tracking; Camera auto-tracking
- Utilizes Google Coral TPU to speed up inference
- Fork of an OSS with major modifications



Efficient Edge Inference: Edge TPU

The Coral TPU (Tensor Processing Unit) is put to the test with different object detection models to prove its effectiveness for inference on the edge.

Model name	**Base validation dataset**	**Input size**	**Model size**
YOLOv8 Nano	90 objects COCO	320×320×3	5.3 MB
SSD MobileNet V2	90 objects COCO	300×300×3	6.7 MB
EfficientDet Lite	90 objects COCO	320×320×3	5.7 MB

Table: Specification of the object detection models for evaluation

Efficient Edge Inference: Edge TPU

Devices	Models	Without TPU	With TPU	Diff-erence
NUC[3]	EfficientDet	200	20	-90%
	MobileNetV2	15	4	-73.3%
	YOLOv8	18	8	-55.6%
RP4[4]	EfficientDet	330	210	-27.3%
	MobileNetV2	28	12	-57.1%
	YOLOv8	40	20	-50%

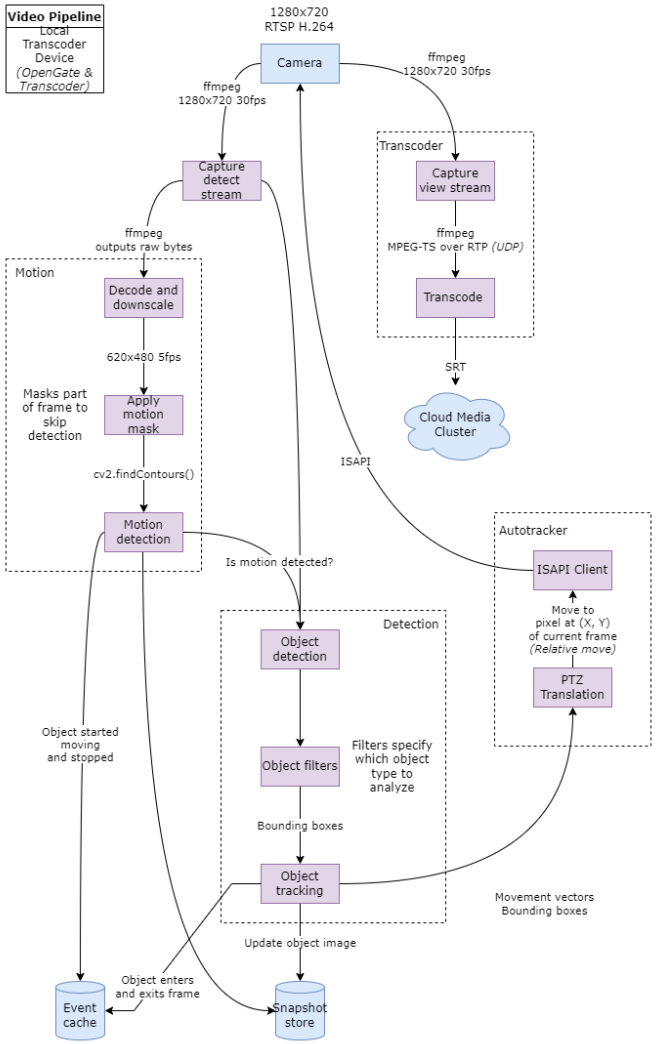
Table: Average inference time (ms) performance comparison between systems with and without an accelerator

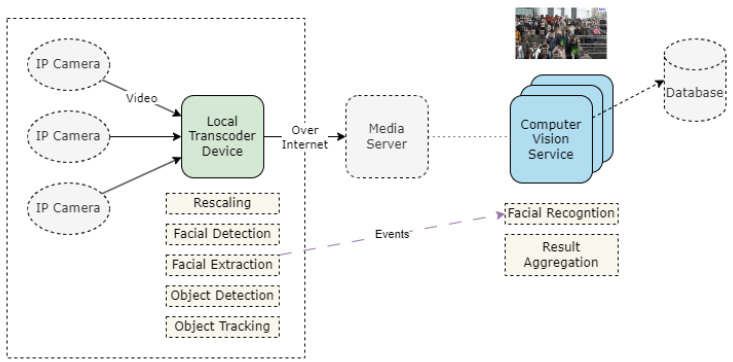
Conclusion

It's evident that utilizing the TPU significantly enhances inference time, in contrast to relying solely on the CPU.

[5] Intel NUC: Intel Core i3-1220P
[6] Raspberry Pi 4

OpenGate: Pipeline





The majority of the detection are performed on the edge. Only facial recognition, which requires access to a vector database, is performed on the cloud.

OpenGate: PTZ Auto-tracking

Naïve approach

Points the camera at the centroid of the moving object. *By the time the camera arrives, the object has already displaced*

Improvement

Move the camera to a predicted point ahead of the current centroid, where the camera will meet the object

OpenGate: PTZ Auto-tracking

On startup, the camera is moved from $(\frac{width}{2}, \frac{height}{2})$ to $(width, height)$ and back to calculate average speed of the camera s_{camera} (calibration)

Predict interception

1. $B_{predict} = B_{curr} + \overrightarrow{v_{obj}}.R.t_{camera}$
2. $(x, y)_{interception} = C_{predict}$

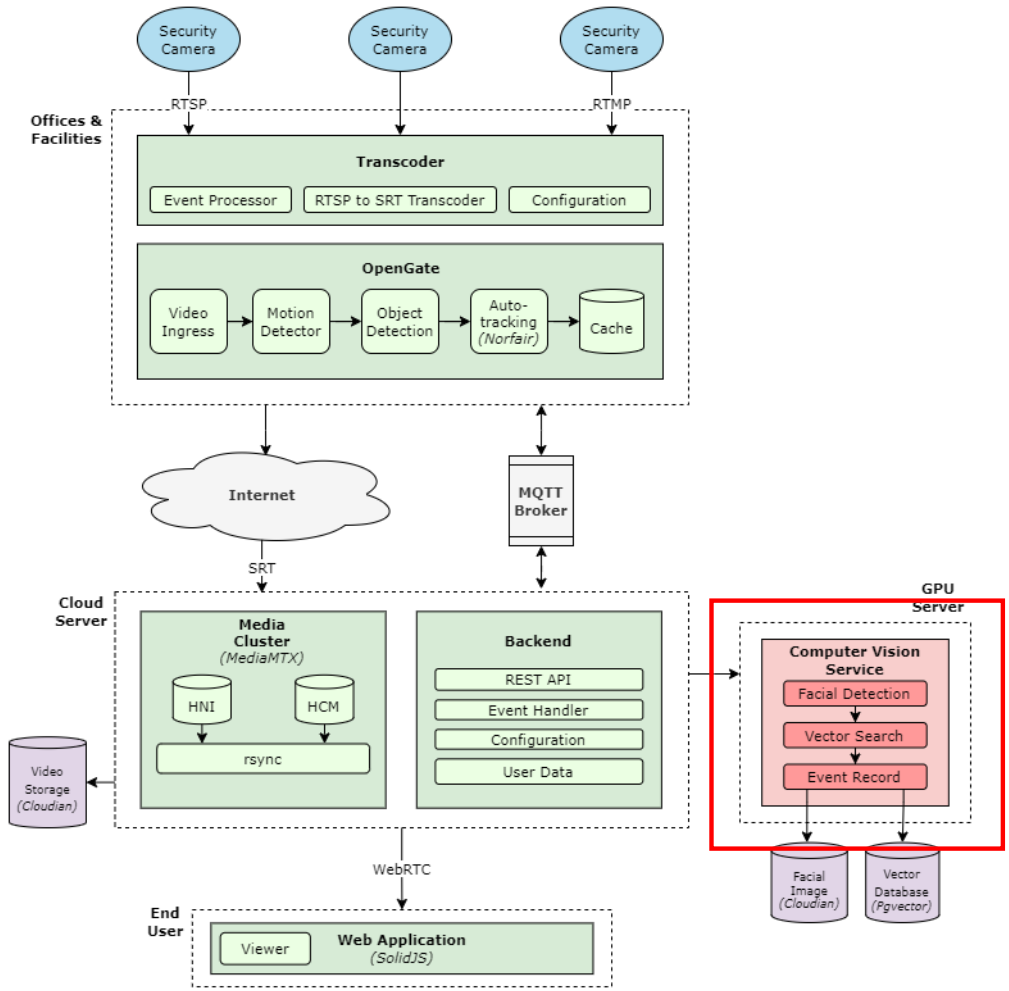
where

- $x \in (0, width); y \in (0, height)$
- $C = (\frac{x_{TL}+x_{BR}}{2}, \frac{y_{TL}+y_{BR}}{2})$ is a centroid
- $C_{frame} = (\frac{0+width}{2}, \frac{height+0}{2})$
- $B = (x_{TL}, y_{TL}, x_{BR}, y_{BR})$ is a bounding box
- $\overrightarrow{v_{obj}} = (v_{x_{TL}}, v_{y_{TL}}, v_{x_{BR}}, v_{y_{BR}})$ is the object's velocity (pixels/frame)
- R is the constant frame rate (frames/seconds)
- $t_{camera} = \frac{\Delta_{C_{frame} \rightarrow C_{curr}}}{s_{camera}}$ is the predicted time of movement between centroid of the frame to the object's

Then, normalize $(x, y)_{interception} \rightarrow x \in (-1, 1), y \in (-1, 1)$ to use *Relative Move* API of ONVIF/ISAPI. The camera locks its auto-tracking for a single object at a time until that object disappears, stopped moving, or camera cannot move anymore.

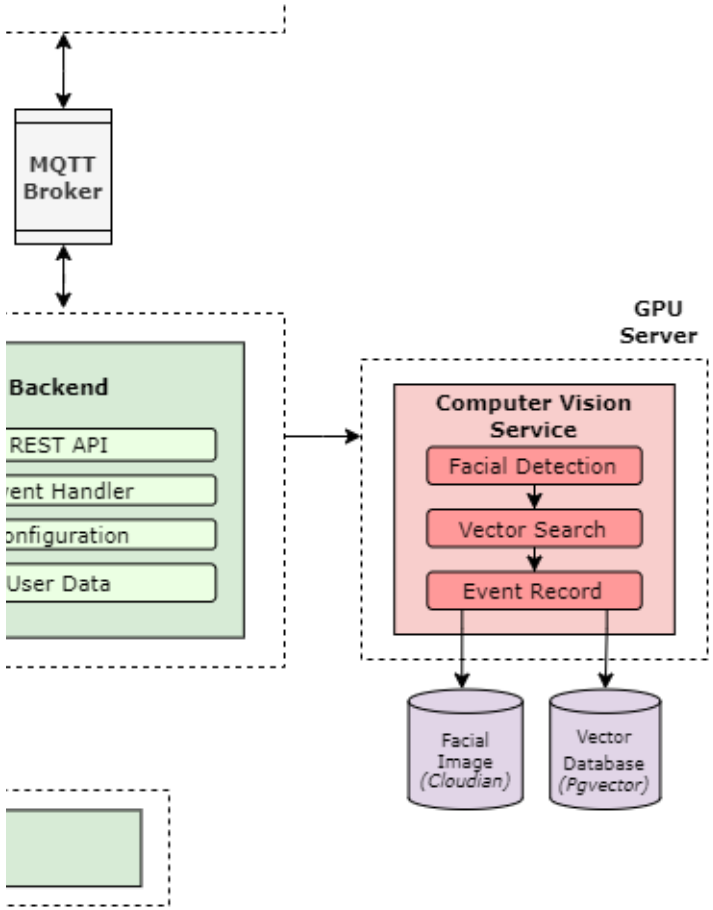
Computer Vision Service

- Performs analytics on-cloud for workloads requiring external data (e.g., faces)
- Features facial detection and recognition
- Utilizes pgvector, a Postgres-based vector database
- Stores facial data in object storage (e.g. Cloudian/S3)



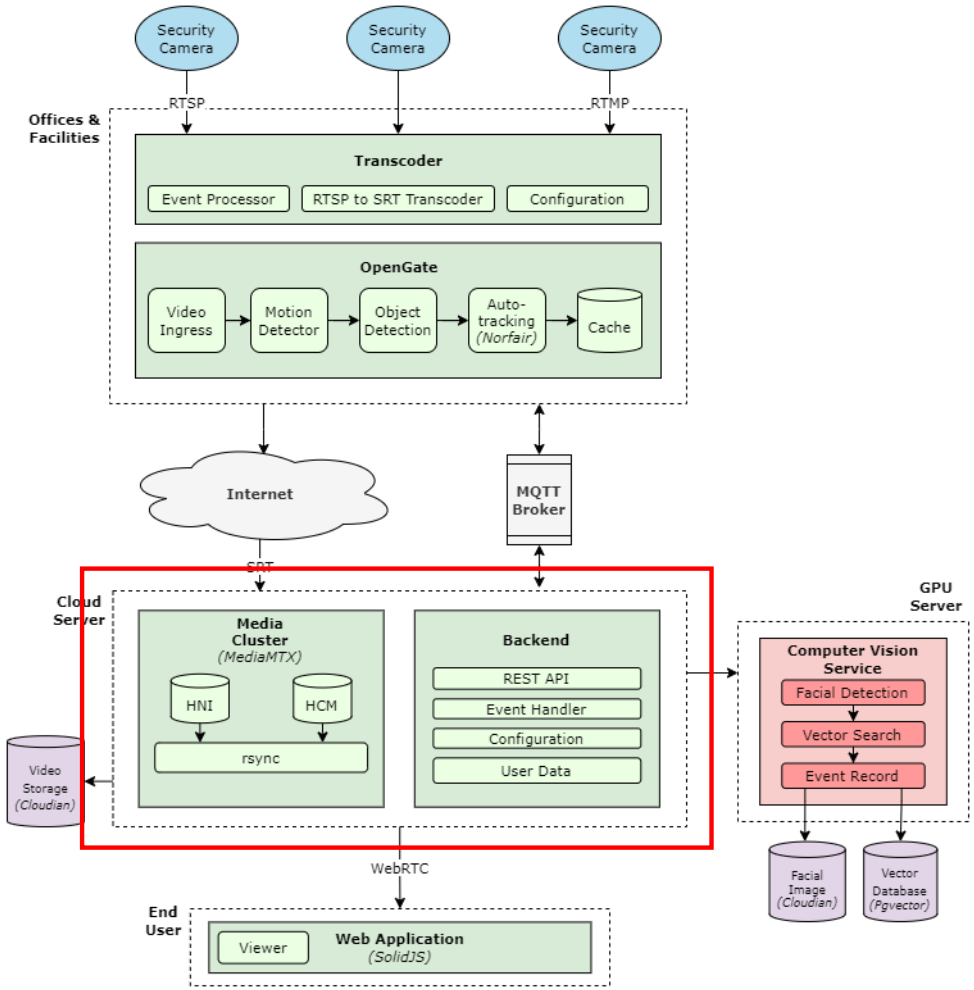
Computer Vision Service

- Performs analytics on-cloud for workloads requiring external data (e.g., faces)
- Features facial detection and recognition
- Utilizes pgvector, a Postgres-based vector database
- Stores facial data in object storage (e.g. Cloudian/S3)



Cloud Services

- Provides APIs for front-end applications and LTDs
- Hosts the media servers required for delivering video streams
- Offers WebRTC and HLS live streams to users



Cloud Services

- Provides APIs for front-end applications and LTDs
- Hosts the media servers required for delivering video streams
- Offers WebRTC and HLS live streams to users

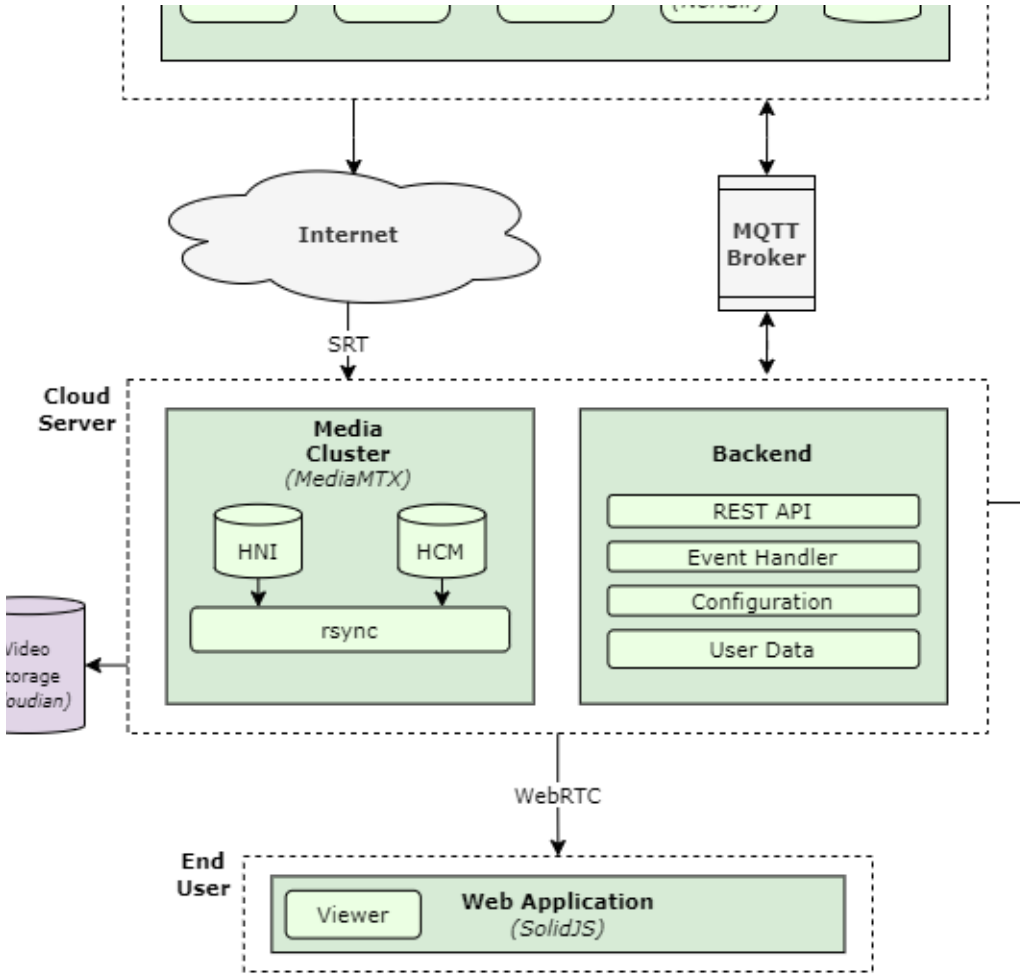


Table of Contents

1 Introduction
- Related Works
- Objectives

2 Architecture
- Camera Cluster
- Local Transcoder Device
- Computer Vision Service
- Cloud Services

3 Demonstration

4 Evaluation
- Stream Latency
- Bandwidth Usage
- Stress Test

5 Conclusion
- Achievements
- Limitations
- Future Developments

Demonstration

Table of Contents

1 Introduction
- Related Works
- Objectives

2 Architecture
- Camera Cluster
- Local Transcoder Device
- Computer Vision Service
- Cloud Services

3 Demonstration

4 Evaluation
- Stream Latency
- Bandwidth Usage
- Stress Test

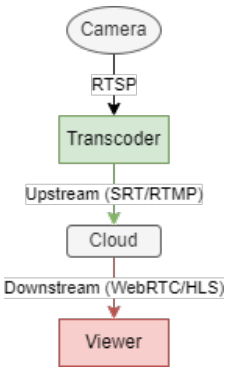
5 Conclusion
- Achievements
- Limitations
- Future Developments

Measuring: Stream Latency

Figure: Image with arrows pointing to the tablets

Stream Latency: Standard

25Mbps upload (cable Internet). Viewer use cable Internet. 720p30



Setup	Latency	Description
SRT & WebRTC	~1.3s	Random artifacts sometimes appear
RTMP &We-bRTC	~2.5s	Random artifacts sometimes appear
SRT & HLS	~8s	No artifacts, stable stream with variable quality
RTMP & HLS	~10s	No artifacts, stable stream with variable quality

Table: Stream latency between protocols in the Standard condition

Conclusion

SRT streams have lower latency than RTMP. HLS streams are stable and reliable due to buffering, but WebRTC streams have significant latency.

Stream Latency: 4G

6Mbps upload (MobiFone 4G). Viewer use cable Internet. 720p30

Setup	Latency	Description
SRT & WebRTC	∼2.6s	Stream freezes and artifacts appears frequently
RTMP & WebRTC	∼4.2s	Stream freezes and artifacts appears frequently
SRT & HLS	∼9s	Stable stream, no artifacts, variable quality
RTMP & HLS	∼12s	Stable stream, no artifacts, variable quality

Table: Stream latency between protocols in the 4G condition

Conclusion

RTMP streams experience greater latency increases than SRT on weaker networks.

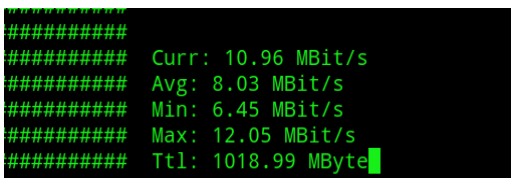
Bandwidth Usage: 720p Video Resolution

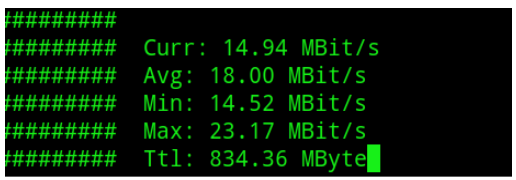
Resolution	Protocol	Quality	Bandwidth Usage
720p	SRT	Medium	6.5 to 8.9 Mbps
		Faster	5.5 to 7.6 Mbps
	RTMP	Medium	6.9 to 10.2Mbps
		Faster	5.6 to 7.5Mbps

Table: Bandwidth usage between protocols at 720p

Conclusion

At both quality settings, the bandwidth usage of SRT is only slightly less than that of RTMP.





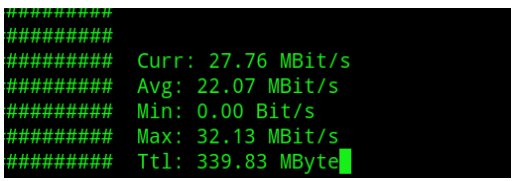


Figure: Outgoing bandwidth usage for 1 to 3 cameras connected to the LTD (left to right)

Bandwidth Usage: 1080p Video Resolution

Resolution	Protocol	Quality	Bandwidth Usage
1080p	SRT	Medium	14.5 to 16 Mbps
		Faster	13.8 to 15Mbps
	RTMP	Medium	18 Mbps to 20Mbps
		Faster	15 to 17.9Mbps

Table: Bandwidth usage between protocols at 1080p

Conclusion

At higher resolution, although SRT is more reliable over unstable connections, the bandwidth usage of two protocols is quite similar.

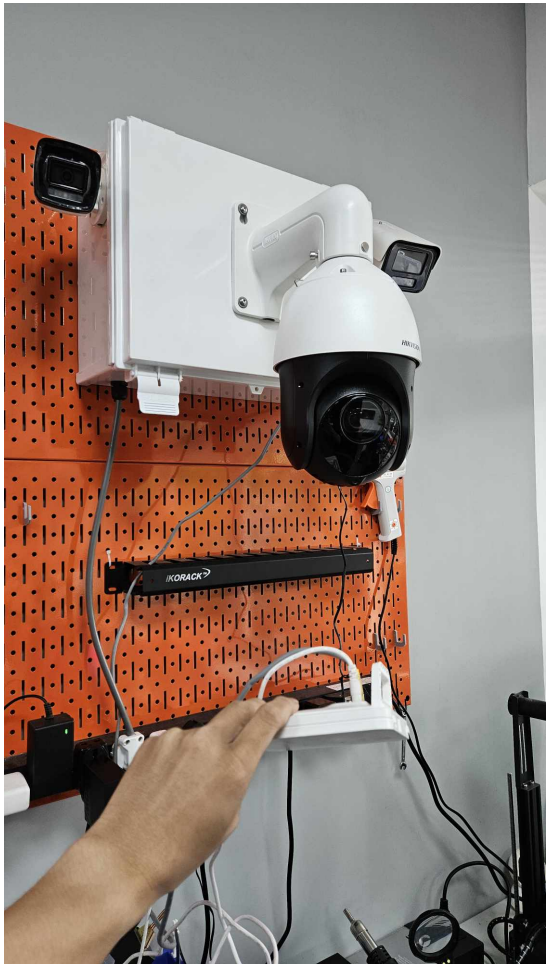
Stress Test: LTD

Tested on a laptop (4 cores 8 thread & 16GB memory + Google Coral TPU)

Incrementally add cameras to the system until instability is observed.

Conditions

Camera origin stream is 1080p25, OpenGate processes at 5FPS/stream, restream to cloud is 720p30 via Mobifone 4G connection. The cameras are connected to the router via Ethernet, while the LTD connects via 2.4GHz Wi-Fi.



Stress Test: Result

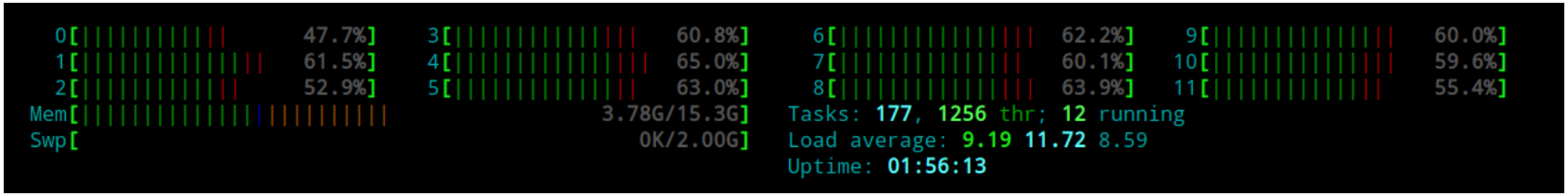


Figure: With 6 cameras, the CPU utilization is typically at 60%, and memory usage is just under 4GB

With more cameras, more streams freezing and timeouts → Device is limited by network bandwidth due to 2.4GHz Wi-Fi

Remark

In our testing conditions, the device can manage up to 6 simultaneous cameras and 7 with a few instability. This is adequate for use cases like retail stores labs, offices or small facilities.

Table of Contents

1 Introduction
- Related Works
- Objectives

2 Architecture
- Camera Cluster
- Local Transcoder Device
- Computer Vision Service
- Cloud Services

3 Demonstration

4 Evaluation
- Stream Latency
- Bandwidth Usage
- Stress Test

5 Conclusion
- Achievements
- Limitations
- Future Developments

Achievements

This thesis has successfully addressed key challenges and developed a more advanced modern VSaaS system compared to existing systems in these aspects:

1. Implemented SRT and WebRTC protocols to significantly reduce latency.
2. Deploying AI processing capabilities on edge devices (LTD) to reduce the traffic and workload on back-end servers
3. Enhanced security by not exposing camera feeds to the public Internet but through LTDs.
4. Developed AI and object-tracking features to detect and track moving people.
5. Conducted testing on the system and have provided guaranteed improvements

Limitations

There are still limitations that affect the performance of built system. Some of significant ones are

1. The system still requires extra LTD devices $\rightarrow$ Still have some upfront cost.
2. Video stream from cameras to LTD devices is not encrypted $\rightarrow$ internal network man-in-the-middle attacks.

Ethical Concerns

We acknowledge the potential misuse of surveillance systems for unauthorized collection of personal data, including faces, movements, and locations, without consent.

Future Developments

There are important areas for future development to enhance the capabilities and performance of VSaaS models:

1. Develop more useful features to utilize collected detection data and gain insights.
2. Research the idea of developing integrated firmware to cameras.
3. Develop a video storage system.
4. Optimize object tracking algorithms.
5. Optimize the Local Transcoder Device to reduce network footprint.

Paper Contribution

Accelerate Emerging Computer Vision Models for Video Surveillance As A Service (VSaaS) using Edge TPU

Abstract

In order to establish a reliable and sustainable edge AI system for remote alerts in a VSaaS system, several crucial factors need to be considered. The performance and accuracy aspects are of utmost importance, particularly as the system tackles increasingly complex tasks. One effective solution to address these challenges is to optimize the existing devices with an Edge TPU (Tensor Processing Unit) to accelerate the inference speed as well as inference accuracy. This paper will consider deeply into these categories, and provide insightful statistics and analysis on the topic

Submitted: 3rd International Conference of Intelligence of Things (ICIT)
May 12, 2024 - Status: In Review

Thanks for listening!