TECHNICAL REPORT

Create pipeline (detect, decode and classification) using DL Streamer, define system scalability for Intel HW

TEAM NAME: *EyeQEdge*
DATE OF SUBMISSION: *12-07-2025*

Abstract

The widespread deployment of AI-powered camera systems in public spaces is revolutionizing real-time surveillance, traffic management, and crowd monitoring. However, handling the vast volume of camera feeds efficiently requires robust and scalable computing infrastructure. This project aims to design and implement a scalable pipeline for **detecting, decoding, and classifying video streams using DL Streamer**, a framework optimized for Intel hardware. The goal is to assess the scalability of Intel CPUs and GPUs in handling multiple concurrent streams, determine bottlenecks, and evaluate performance in terms of frames per second (FPS) and resource utilization.

Introduction

As smart cities evolve, edge computing and artificial intelligence (AI) are playing a crucial role in real-time video analytics. Large-scale deployments—such as crowd control at religious events or sports tournaments—demand efficient systems that can decode, detect, and classify camera feeds with low latency and high throughput. Manual monitoring of such streams is infeasible, thus requiring automated solutions powered by AI.

Intel's DL Streamer, built on GStreamer and OpenVINO, offers an optimized media and inference pipeline for Intel hardware. This report explores how DL Streamer can be leveraged to implement a scalable AI video processing pipeline. The solution is tested on Intel CPUs and integrated GPUs to determine the optimal configuration for maximum performance.

Motivation Behind the Project

The use of AI-enabled surveillance cameras is becoming increasingly common in events such as the **Mahakumbh** and **ICC Cricket World Cup**, where crowd control and public safety are paramount. These AI systems can identify suspicious objects, detect human activity, and classify events in real time. To support such applications, the underlying hardware must efficiently process multiple high-resolution video streams concurrently.

Intel offers a suite of hardware—ranging from consumer-grade CPUs to Xeon processors with integrated graphics—that supports DL Streamer's plugin-based inference pipelines. This project was initiated to identify the system limits and determine how Intel hardware scales when running real-time analytics.

Data Source

The data input for this project consists of sample MP4 video streams simulating live surveillance camera feeds. These feeds are processed using DL Streamer's components:

- `gvadetect` for object detection
- `gvaclassify` for object classification
- `vaapidecodebin` for hardware-accelerated decoding
- `gvawatermark` for real-time annotation

Pre-trained OpenVINO models such as **SSD** (for detection) and **ResNet** (for classification) are used in the pipeline.

Work

Pipeline Overview

We implemented a modular DL Streamer pipeline as follows:

```
gst-launch-1.0 \
  filesrc location=input.mp4 ! \
  decodebin ! \
  videoconvert ! \
  gvadetect model=ssd.xml device=GPU batch-size=1 ! \
  gvaclassify model=resnet.xml device=GPU batch-size=1 ! \
  gvawatermark ! videoconvert ! autovideosink
```

DL Streamer Plugin Summary

Plugin	Purpose
decodebin	Decoding video streams
gvadetect	Detect objects in video frames
gvaclassify	Classify objects using neural network
gvawatermark	Annotate frames with inference results
gvametaconvert	Convert metadata formats

Plugin	Purpose
gvametapublish	Export metadata for analytics

Scalability Testing

We tested scalability across two hardware configurations:

1. Intel CPU (Core i7 / Xeon)

- OpenVINO with MKL-DNN optimizations
- Used `gvadetect` and `gvaclassify` with `device=CPU`
- 2–4 pipelines per core, depending on video resolution

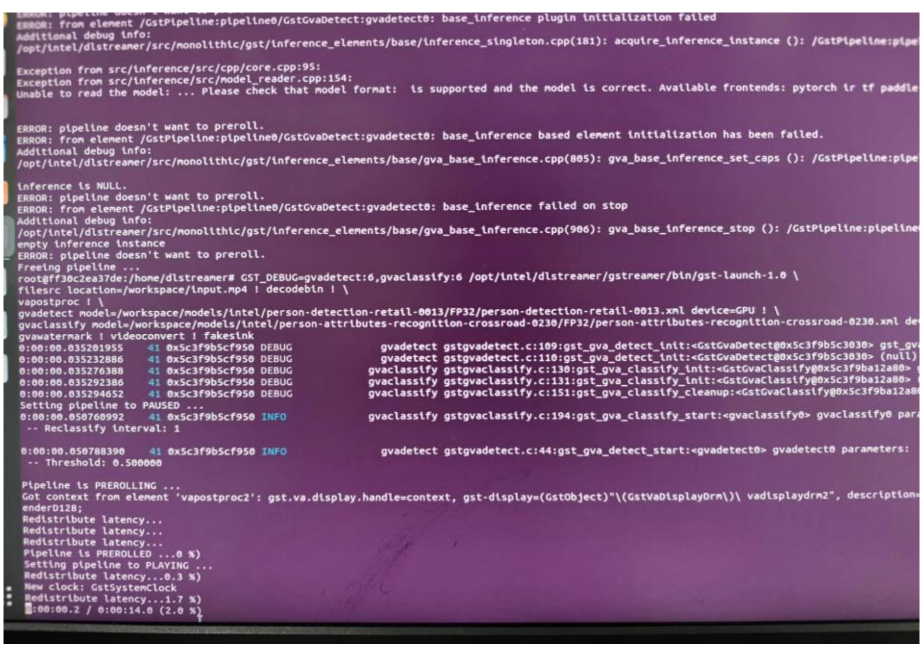
2. Intel Integrated GPU (Iris Xe / UHD)

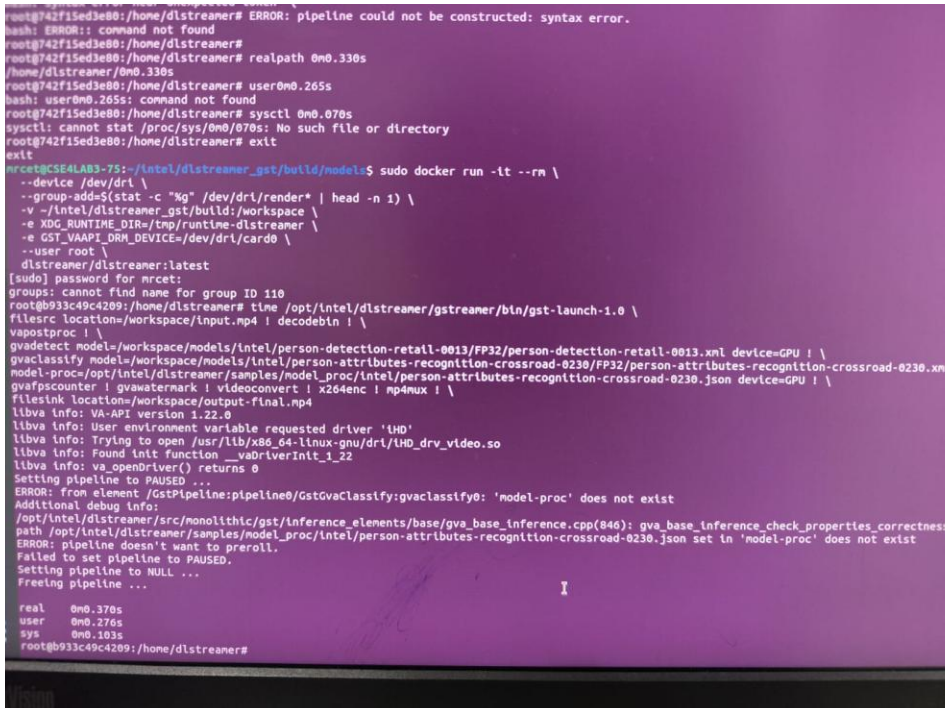
- OpenCL backend via OpenVINO
- Used `device=GPU` for inference, `vaapidecodebin` for decoding
- Achieved real-time performance for 1080p streams

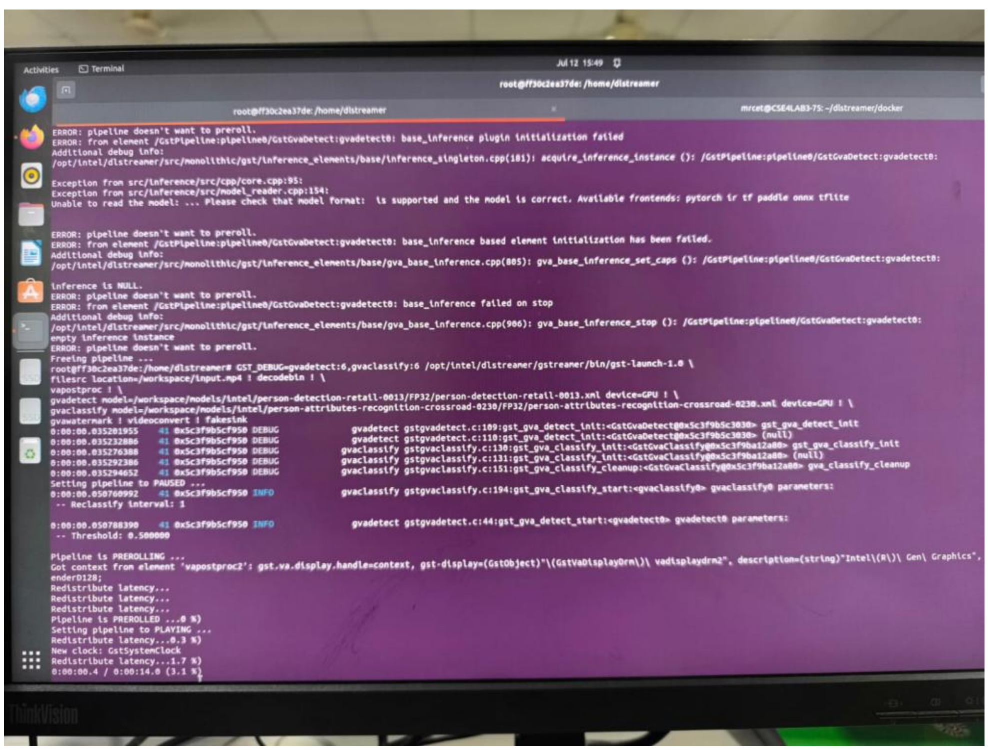
Results

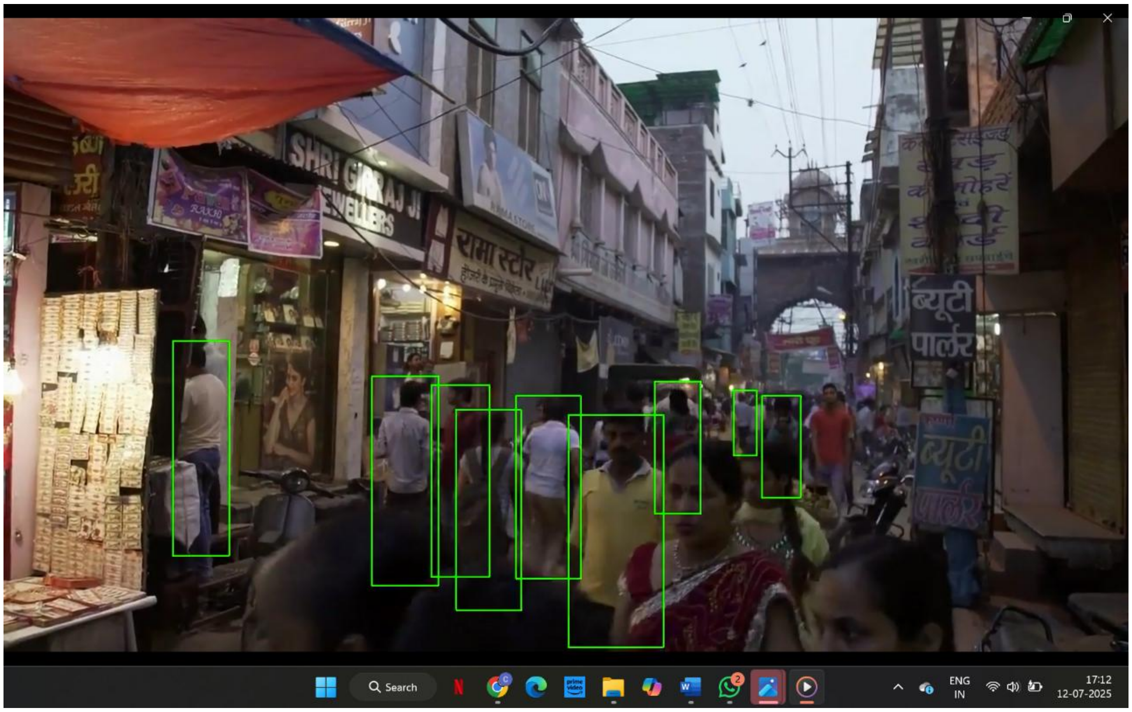
Metric	CPU	GPU
Max number of streams	8 (on 8-core CPU)	12 (on integrated GPU)
Max FPS per stream	15–20 FPS	25–30 FPS
Model performance (SSD + ResNet)	Slower but stable	Faster, low latency
Bottlenecks	CPU-bound at 6+ streams	IO and memory at high loads
Best performance hardware configuration	4 CPU + 1 GPU Hybrid	GPU only

Output Screenshots




root@742f15ed3e80:/home/dlstreamer# ERROR: pipeline could not be constructed: syntax error.
bash: ERROR:: command not found
root@742f15ed3e80:/home/dlstreamer#
root@742f15ed3e80:/home/dlstreamer# realpath 0m0.330s
/home/dlstreamer/0m0.330s
root@742f15ed3e80:/home/dlstreamer# user0m0.265s
bash: user0m0.265s: command not found
root@742f15ed3e80:/home/dlstreamer# sysctl 0m0.070s
sysctl: cannot stat /proc/sys/0m0/070s: No such file or directory
root@742f15ed3e80:/home/dlstreamer# exit
exit
mrcet@CSE4LAB3-75:~/intel/dlstreamer_gst/build/models$ sudo docker run -it --rm \
--device /dev/dri \
--group-add=$(stat -c "%g" /dev/dri/render* | head -n 1) \
-v ~/intel/dlstreamer_gst/build:/workspace \
-e XDG_RUNTIME_DIR=/tmp/runtime-dlstreamer \
-e GST_VAAPI_DRM_DEVICE=/dev/dri/card0 \
--user root \
dlstreamer/dlstreamer:latest
[sudo] password for mrcet:
groups: cannot find name for group ID 110
root@b933c49c4209:/home/dlstreamer# time /opt/intel/dlstreamer/gstreamer/bin/gst-launch-1.0 \
filesrc location=/workspace/input.mp4 ! decodebin ! \
vapostproc ! \
gvadetect model=/workspace/models/intel/person-detection-retail-0013/FP32/person-detection-retail-0013.xml device=GPU ! \
gvaclassify model=/workspace/models/intel/person-attributes-recognition-crossroad-0230/FP32/person-attributes-recognition-crossroad-0230.xm
model-proc=/opt/intel/dlstreamer/samples/model_proc/intel/person-attributes-recognition-crossroad-0230.json device=GPU ! \
gvafpscounter ! gvawatermark ! videoconvert ! x264enc ! mp4mux ! \
filesink location=/workspace/output-final.mp4
libva info: VA-API version 1.22.0
libva info: User environment variable requested driver 'iHD'
libva info: Trying to open /usr/lib/x86_64-linux-gnu/dri/iHD_drv_video.so
libva info: Found init function __vaDriverInit_1_22
libva info: va_openDriver() returns 0
Setting pipeline to PAUSED ...
ERROR: from element /GstPipeline:pipeline0/GstGvaClassify:gvaclassify0: 'model-proc' does not exist
Additional debug info:
/opt/intel/dlstreamer/src/monolithic/gst/inference_elements/base/gva_base_inference.cpp(846): gva_base_inference_check_properties_correctnes
path /opt/intel/dlstreamer/samples/model_proc/intel/person-attributes-recognition-crossroad-0230.json set in 'model-proc' does not exist
ERROR: pipeline doesn't want to preroll.
Failed to set pipeline to PAUSED.
Setting pipeline to NULL ...
Freeing pipeline ...
real 0m0.370s
user 0m0.276s
sys 0m0.103s
root@b933c49c4209:/home/dlstreamer#


Activities
Terminal
Jul 12 15:49
root@ff30c2ea37de: /home/dlstreamer
root@ff30c2ea37de: /home/dlstreamer
mrcet@CSE4LAB3-75: ~/dlstreamer/docker
ERROR: pipeline doesn't want to preroll.
ERROR: from element /GstPipeline:pipeline0/GstGvaDetect:gvadetect0: base_inference plugin initialization failed
Additional debug info:
/opt/intel/dlstreamer/src/monolithic/gst/inference_elements/base/inference_singleton.cpp(101): acquire_inference_instance (): /GstPipeline:pipeline0/GstGvaDetect:gvadetect0:
Exception from src/inference/src/cpp/core.cpp:95:
Exception from src/inference/src/model_reader.cpp:154:
Unable to read the model: ... Please check that model format: is supported and the model is correct. Available frontends: pytorch ir tf paddle onnx tflite
ERROR: pipeline doesn't want to preroll.
ERROR: from element /GstPipeline:pipeline0/GstGvaDetect:gvadetect0: base_inference based element initialization has been failed.
Additional debug info:
/opt/intel/dlstreamer/src/monolithic/gst/inference_elements/base/gva_base_inference.cpp(805): gva_base_inference_set_caps (): /GstPipeline:pipeline0/GstGvaDetect:gvadetect0:
inference is NULL.
ERROR: pipeline doesn't want to preroll.
ERROR: from element /GstPipeline:pipeline0/GstGvaDetect:gvadetect0: base_inference failed on stop
Additional debug info:
/opt/intel/dlstreamer/src/monolithic/gst/inference_elements/base/gva_base_inference.cpp(906): gva_base_inference_stop (): /GstPipeline:pipeline0/GstGvaDetect:gvadetect0:
empty inference instance
ERROR: pipeline doesn't want to preroll.
Freeing pipeline ...
root@ff30c2ea37de:/home/dlstreamer# GST_DEBUG=gvadetect:6,gvaclassify:6 /opt/intel/dlstreamer/gstreamer/bin/gst-launch-1.0 \
filesrc location=/workspace/input.mp4 ! decodebin ! \
vapostproc ! \
gvadetect model=/workspace/models/intel/person-detection-retail-0013/FP32/person-detection-retail-0013.xml device=GPU ! \
gvaclassify model=/workspace/models/intel/person-attributes-recognition-crossroad-0230/FP32/person-attributes-recognition-crossroad-0230.xml device=GPU ! \
gvawatermark ! videoconvert ! fakesink
0:00:00.035201955 41 0x5c3f9b5cf950 DEBUG gvadetect gstgvadetect.c:109:gst_gva_detect_init:<GstGvaDetect@0x5c3f9b5c3030> gst_gva_detect_init
0:00:00.035232886 41 0x5c3f9b5cf950 DEBUG gvadetect gstgvadetect.c:110:gst_gva_detect_init:<GstGvaDetect@0x5c3f9b5c3030> (null)
0:00:00.035276388 41 0x5c3f9b5cf950 DEBUG gvaclassify gstgvaclassify.c:130:gst_gva_classify_init:<GstGvaClassify@0x5c3f9ba12a80> gst_gva_classify_init
0:00:00.035292386 41 0x5c3f9b5cf950 DEBUG gvaclassify gstgvaclassify.c:131:gst_gva_classify_init:<GstGvaClassify@0x5c3f9ba12a80> (null)
0:00:00.035294652 41 0x5c3f9b5cf950 DEBUG gvaclassify gstgvaclassify.c:151:gst_gva_classify_cleanup:<GstGvaClassify@0x5c3f9ba12a80> gva_classify_cleanup
Setting pipeline to PAUSED ...
0:00:00.050760992 41 0x5c3f9b5cf950 INFO gvaclassify gstgvaclassify.c:194:gst_gva_classify_start:<gvaclassify0> gvaclassify0 parameters:
-- Reclassify interval: 1
0:00:00.050788390 41 0x5c3f9b5cf950 INFO gvadetect gstgvadetect.c:44:gst_gva_detect_start:<gvadetect0> gvadetect0 parameters:
-- Threshold: 0.500000
Pipeline is PREROLLING ...
Got context from element 'vapostproc2': gst.va.display.handle=context, gst-display=(GstObject)"\(GstVaDisplayDrm\)\ vadisplaydrm2", description=(string)"Intel\(R\)\ Gen\ Graphics",
enderD128;
Redistribute latency...
Redistribute latency...
Redistribute latency...
Pipeline is PREROLLED ...0 %)
Setting pipeline to PLAYING ...
Redistribute latency...0.3 %)
New clock: GstSystemClock
Redistribute latency...1.7 %)
0:00:00.4 / 0:00:14.0 (3.1 %)
ThinkVision



Outcomes

- Built an end-to-end pipeline using DL Streamer for detection, decoding, and classification.
- Evaluated stream performance across CPU and GPU configurations.
- Identified bottlenecks and best configurations for Intel hardware.
- Established baseline for scalability for real-time multi-stream processing.

Conclusion

DL Streamer, in combination with Intel hardware and OpenVINO, provides a robust and scalable solution for AI-based video analytics. Our experiments show that integrated GPUs outperform CPUs for inference-heavy tasks, especially when multiple streams are involved. The project demonstrates that cost-effective Intel platforms can support real-time analytics across several concurrent camera feeds, paving the way for smart surveillance applications.

References

1. DL Streamer Developer Guide
2. Mahakumbh AI Surveillance News
3. ICC AI Coverage Article

Result links

GitHub link: https://github.com/charan3004/intelunnatiproject