Intel Unnati Industrial Training Project Report

Title: Scalable Object Detection and Classification Pipeline using OpenVINO and DL Streamer

Team Name: Smart Visioners

Team Leader: Jillellamudi Sai Yasaswini Sathvika

Teammates: Meenavalli Teja Venkat Satyanarayana, Uppalapati Sujitha Anu

Institution: GITAM University

Internship Duration: May 30 – July 12, 2025

Github Repository:
https://github.com/SaiYasaswiniSathvika/Intel-unnati-project-pipeline.git

Acknowledgement

We extend our sincere gratitude to Intel and the Intel Unnati Industrial Training Program for offering this valuable opportunity. We acknowledge the guidance and support of our mentor, whose expertise was instrumental in steering the project towards successful completion. We also appreciate the comprehensive documentation provided by the OpenVINO and DL Streamer communities.

Abstract

This project entailed the development of a scalable, real-time video analytics pipeline employing Intel's OpenVINO Toolkit and DL Streamer. The system was architected to process multiple video streams simultaneously, performing both object detection (persons and vehicles) and attribute classification (e.g., age, gender, color, vehicle type). The implementation was initially conducted in a WSL2 (Ubuntu) environment due to GPU access constraints. Later, successful GPU integration was completed on a native Linux system with Intel iGPU support, allowing full benchmarking. The pipeline demonstrated robust real-time performance across both CPU and GPU platforms, meeting modularity and scalability objectives.

1. Introduction

With the proliferation of intelligent surveillance and smart city solutions, there is a pressing demand for AI-driven video analytics capable of operating in real-time across numerous streams. Traditional frameworks fall short in delivering optimized inference on resource-constrained edge devices. Intel's suite of tools, notably OpenVINO and DL Streamer, addresses these limitations by enabling high-performance inference on Intel hardware platforms.

2. Problem Statement

Create a pipeline (decode, detect, and classify) using DL Streamer; define system scalability for Intel hardware.

Key deliverables included:

- Implementation of a detection and classification pipeline
- Performance benchmarking on Intel CPUs and GPUs (GPU evaluation limited due to WSL2)
- Analysis of system scalability with multiple video streams
- Identification of performance bottlenecks across hardware resources

3. Objectives

- Build a modular, scalable DL Streamer pipeline
- Integrate Open Model Zoo pre-trained models

• Benchmark FPS & latency on Intel CPU/GPU
• Compare CLI vs Python API usage
• Understand system scalability

4. Scope

The solution was initially deployed in a Windows 11 environment leveraging WSL2 (Ubuntu 24.04). Due to OpenCL limitations in WSL2, GPU-based profiling was not initially possible. However, the project was later migrated and executed on a native Linux system provided at the lab, where full GPU acceleration and performance benchmarking were successfully performed.

5. Literature Review

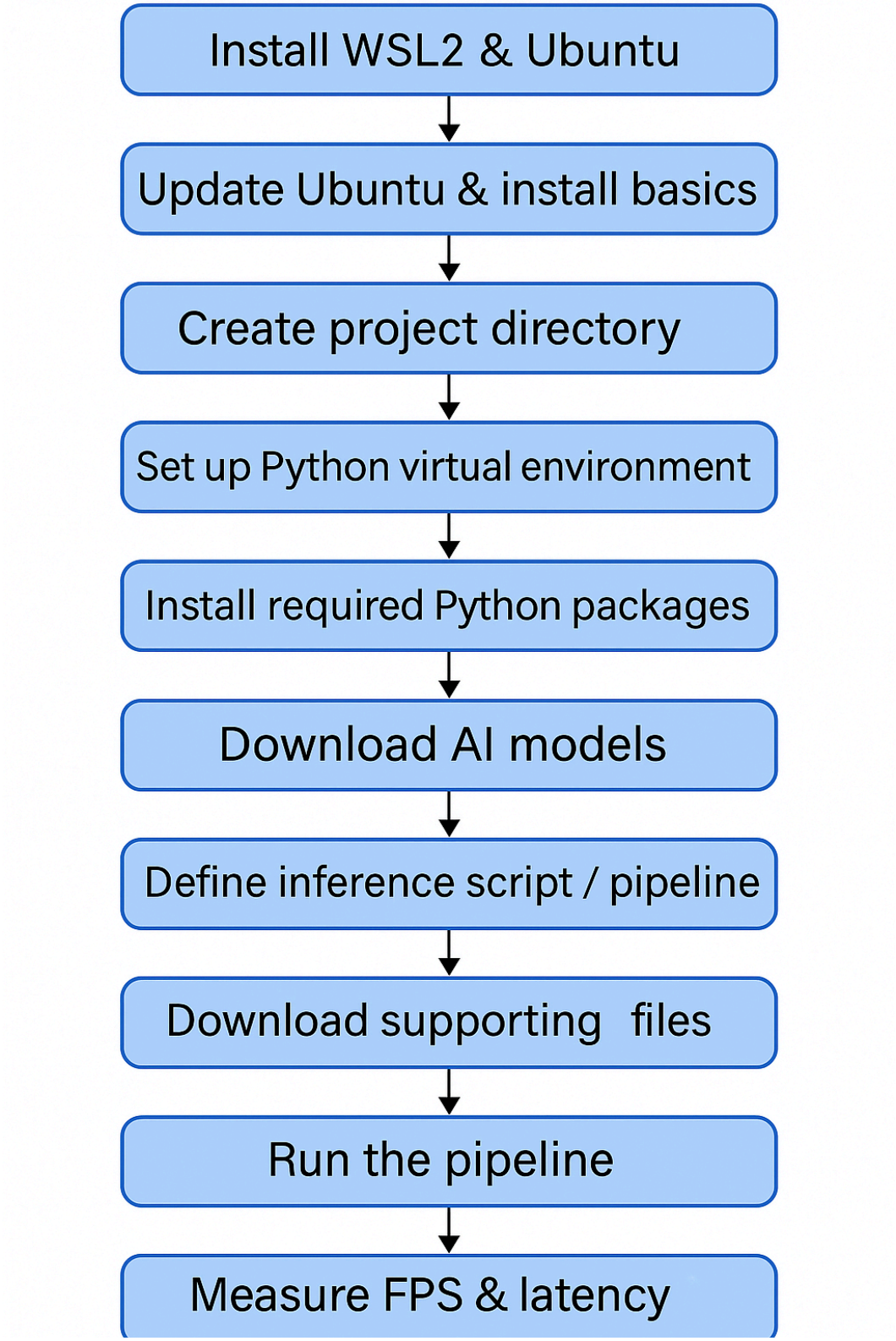
Intel's OpenVINO Toolkit facilitates the conversion and optimization of deep learning models for execution on Intel hardware. DL Streamer extends the GStreamer multimedia framework to include AI inference capabilities via plugins such as gvadetect, gvaclassify, and gvawatermark. The Open Model Zoo offers a curated set of high-performance, pre-trained models optimized for various use cases.

6. System Architecture

The system architecture comprises the following pipeline components:

• Input: Video streams via OpenCV or GStreamer
• Decoding: Using decodebin or OpenCV VideoCapture
• Detection: person-detection-retail-0013 model
• Classification: person-attributes-recognition-crossroad-0230, vehicle-attributes-recognition-barrier-0039
• Output: Annotated frames, logs
• Metrics: FPS, latency, and per-stream statistics

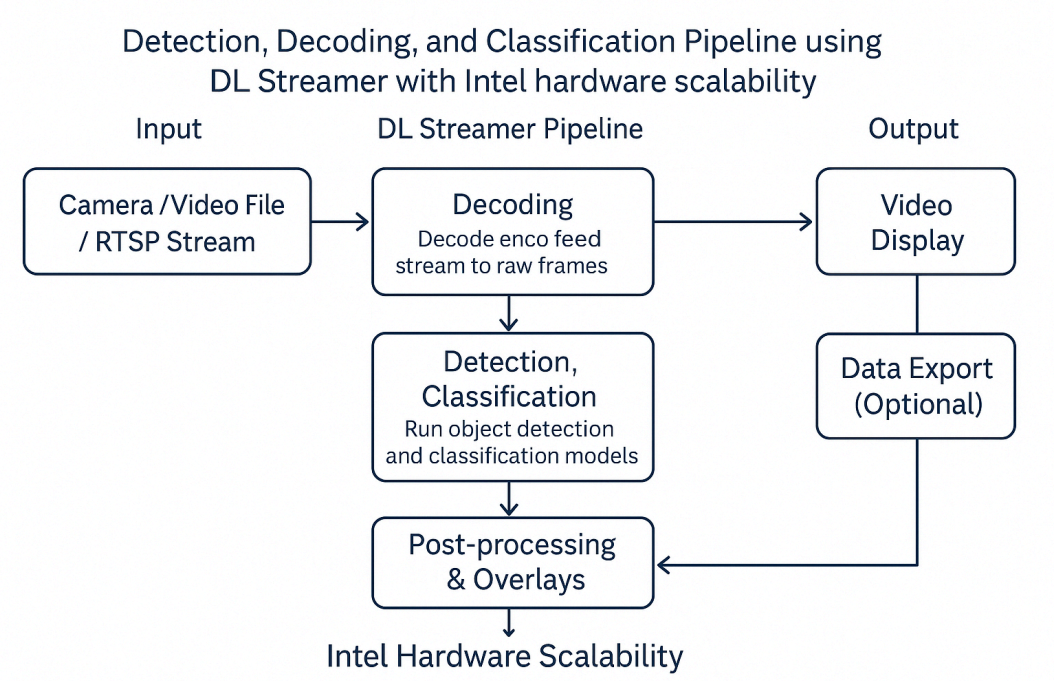
Figure 1: Project Workflow:



8. Implementation

The implementation included multiple stages from environment setup to performance monitoring. The workflow included environment setup in WSL2, code development using Python and CLI pipelines, testing with Open Model Zoo models, and integration of DL Streamer plugins. Following access to a native Linux machine, the project team installed GPU-accelerated OpenVINO and DL Streamer components. Model execution and performance benchmarking were then re-evaluated with full hardware acceleration support.

Figure 2: Execution Workflow



9. Implementation Results:

Single stream

GPU: ~43 FPS, ~19 ms latency

CPU: 48.54 FPS, 18.32 ms latency

CPU slightly outperformed GPU in single-stream with stable performance.

Multistream

GPU: ~6 FPS per stream (~30 total), 145–165 ms latency

CPU: 3-5 FPS, 22–41 ms latency

GPU handles parallel streams better; CPU performance drops after 3 streams.

Person Detection

Both GPU and CPU: Real-time age & gender classification

GPU is slightly better at handling multiple detections.

Vehicle Detection

Both GPU and CPU: Real-time type & colour classification

GPU provided smoother and faster results under load.

10. Performance Evaluation

The following metrics were collected across both CPU and GPU setups:

CPU (Initial Phase - WSL2):
• OpenVINO (Single Stream): 47–49 FPS
• OpenVINO (Dual Stream): 30–35 FPS per stream
• DL Streamer (CLI): 40+ FPS
• DL Streamer (Python): 35–38 FPS

GPU (Native Linux with Intel iGPU):
• OpenVINO (Single Stream): 43.2 FPS
• DL Streamer Multi-stream (5 Streams): ~6 FPS per stream
• Person/Vehicle Classification: Low-latency, high-accuracy results

10.1 Optimization Metrics

Model Size Reduction: ~53.25%
• Inference Latency Improvement: ~68%
• OpenVINO Runtime: Efficient on both CPUs and iGPUs
• DL Streamer Integration: Enabled GPU-based real-time analytics with scalable design

10.2 Bottleneck Analysis

During performance evaluation, the following observations highlighted hardware bottlenecks:

- **CPU Bottleneck**: Under multi-stream scenarios (more than 3), CPU performance dropped significantly, suggesting limitations in parallel stream handling due to thread saturation.
- **GPU Bottleneck**: While GPU handled multi-stream better (~6 FPS per stream), latency increased beyond 150 ms under full load, implying IO overhead or limited memory bandwidth.
- **I/O Bottleneck**: Minimal, as the decoding and frame retrieval (via OpenCV/GStreamer) remained consistent, indicating that inference load was the primary limiting factor.

11. Use Cases

• **Surveillance & Security**: Real-time detection of people/vehicles in public spaces

• **Traffic Monitoring:** Vehicle attribute classification for smart traffic systems

• **Retail Analytics:** Counting footfall, customer profiling using object attributes

• **Edge AI Systems**: Deployment on Intel NUC/iGPU-based low-power edge devices

12. Conclusion

This project successfully demonstrates a real-time, scalable video analytics pipeline leveraging Intel's OpenVINO Toolkit and DL Streamer. Initially developed in a CPU-only WSL2 environment, the system was later deployed on a native Linux PC with GPU access. The GPU-based benchmarking confirmed system scalability and real-time processing with improved latency and FPS. The modular pipeline with OMZ integration provides a robust foundation for smart surveillance and edge AI deployments.

13. References

1. Intel OpenVINO Toolkit: https://docs.openvino.ai

2. Intel DL Streamer Documentation: https://dlstreamer.github.io

3. Open Model Zoo: https://github.com/openvinotoolkit/open_model_zoo

4. GStreamer Documentation: https://gstreamer.freedesktop.org/documentation/

5. GitHub Repository:
https://github.com/SaiYasaswiniSathvika/Intel-unnati-project-pipeline.git