# Software Requirements Specification (SRS)

## Project Title:

Real-Time Multi-Source Video Streaming System

## Platform:

Linux (C Language with IPC mechanisms)

# Introduction

## Purpose

This SRS document defines the requirements for the real-time multi-source video streaming system. The system enables concurrent acquisition, processing, detection, encoding, and streaming of video frames in real time using inter-process communication techniques in C under a Linux environment.

## Scope

The project handles video frames coming from multiple simulated sources (like cameras or drones), processes the frames for basic enhancements, detects objects, encodes them, and streams to clients in real time with low latency. It leverages Linux IPC mechanisms (Shared Memory, Semaphores, Message Queues, Sockets).

## Definitions, Acronyms, and Abbreviations

* + - **IPC**: Inter-Process Communication
    - **RTS**: Real-Time Streaming
    - **POSIX**: Portable Operating System Interface
    - **TCP**: Transmission Control Protocol

# Functional Requirements

|  |  |  |
| --- | --- | --- |
| **ID** | **Function** | **Description** |
| FR1 | Frame  Acquisition | Acquire frames from multiple sources into shared  memory. |
| FR2 | Frame  Processing | Read frames, apply basic processing, and forward for  detection. |
| FR3 | Object  Detection | Detect objects in the processed frame and store  metadata. |
| FR4 | Video  Encoding | Encode and compress frame data; send via message  queues. |
| FR5 | Real-Time  Streaming | Stream encoded frames to a TCP client (viewing server). |

1. **Non-Functional Requirements**

* **Concurrency**: Support at least 5 processing modules (producers/consumers).
* **Reliability**: Should gracefully handle module crashes, memory cleanup using cleanup.c.
* **Efficiency**: Minimal CPU, memory use; real-time frame processing C streaming.
* **Scalability**: Easily extendable to real camera/video data in the future.

# Software and Hardware Requirements

## Software Requirements

* + - **OS**: Ubuntu Linux
    - **Compiler**: GCC
    - **Tools**: ipcs, ipcrm, shmget, semget, msgget, socket
    - **Language**: C

## Hardware Requirements

* + - Simulated video frames (textual dummy frames)
    - No actual camera required

# System Overview (Pocess-Based)

**Process Flow:**

1. **Producer Process**: Writes video frame data to shared memory.
2. **Consumer Process**: Reads from shared memory, processes frame data.
3. **Object Detector**: Detects objects and writes detection info to shared memory.
4. **Detection Consumer**: Reads detection info and prepares for encoding.
5. **Video Encoder**: Encodes the processed data and sends it to message queue.
6. **Encoding Consumer (TCP Client)**: Reads encoded frame and sends to server.
7. **Streaming Server (TCP Server)**: Receives encoded data and displays it.

# Constraints

* Each module/process must release IPC resources upon exit using cleanup.c.
* Manual multi-terminal launching required for separate processes.
* Dummy frames used in simulation.

# Appendices

## Assumptions

* + Users will run each module manually in separate terminals.
  + Simulated data (frames as text) is acceptable in place of real video.

## Glossary

* + **Frame**: A single image or data block representing a video capture.
  + **Client**: The receiver module (TCP server) displaying the video stream.

TCP Socket

Message Queue

Message Queue

Shared Memory

Shared Memory