Software Requirements Specification (SRS)

# Project Title: Multi-Source Live Aviation Monitoring and Display System

## Platform:

Linux (C Language with IPC and Multithreading mechanisms)

# 1. Introduction

## 1.1 Purpose

This SRS document defines the requirements for a real-time aviation monitoring system that acquires, processes, synchronizes, and displays live video and sensor data from multiple aircraft, drones, and radar sources. The system ensures minimal latency and accurate visualization for air traffic control and onboard monitoring.

## 1.2 Scope

The system captures live video and sensor data (altitude, speed, GPS) from multiple sources, processes them in parallel, performs object detection and tracking, encodes and streams the data, and displays it on a real-time dashboard. It uses Linux IPC mechanisms (Shared Memory, Semaphores, Message Queues, Sockets) and multithreading for concurrency and synchronization.

## 1.3 Definitions, Acronyms, and Abbreviations

- IPC: Inter-Process Communication  
- RTS: Real-Time Streaming  
- UI: User Interface  
- GPS: Global Positioning System  
- TCP: Transmission Control Protocol  
- POSIX: Portable Operating System Interface

# 2. Functional Requirements

|  |  |  |
| --- | --- | --- |
| ID | Function | Description |
| FR1 | Multi-Source Data Acquisition | Acquire live video and sensor data from aircraft, drones, and radar using multithreading. |
| FR2 | Parallel Data Processing | Process video (e.g., stabilization, overlays) and sensor data (e.g., altitude, GPS) in parallel. |
| FR3 | Object Detection & Tracking | Detect and track moving objects across multiple video feeds. |
| FR4 | Data Encoding & Communication | Encode and compress video/sensor data; transmit via sockets and message queues. |
| FR5 | Real-Time Display & Dashboard | Display synchronized video and sensor data on an interactive UI dashboard. |
| FR6 | Fault Detection & Recovery | Detect failures in modules and recover using signal handling and watchdog threads. |

# 2.1 Modules

**Module 1**: **Multi-Source Video and Sensor Data Acquisition**  
Acquire live video frames and sensor data (altitude, speed, GPS) from multiple sources concurrently.

**Module 2**:  **Parallel Data Processing Pipeline**  
Process video and sensor data in parallel for tasks such as image stabilization, noise reduction, altitude overlay, and flight path visualization. (Manchikanti Apurupa)

**Module 3**: **Object Detection and Tracking**  
Implement real-time detection and tracking of moving objects (other aircraft, obstacles, or drones) across multiple feeds.

**Module 4:** **Data Encoding, Compression, and Inter-Process Communication**  
Encode and compress video streams and sensor data for real-time distribution to control stations or onboard displays. (Ragula Saketh).

**Module 5:**  **Real-Time Display and UI Dashboard**  
Display processed video and sensor data on an interactive UI dashboard for monitoring and decision-making.

**Module 6:**  **Fault Detection, Signal Handling, and Recovery**  
Ensure system stability by detecting failures in video streams, sensor threads, or processing modules. (Adarsh Rai)

# 3. Non-Functional Requirements

- Concurrency: Support multithreaded and multi-process architecture for real-time performance.  
- Reliability: Handle module failures gracefully using signal handling and watchdogs.  
- Efficiency: Optimize CPU and memory usage for real-time responsiveness.  
- Scalability: Extendable to additional aircraft or sensor types.  
- Synchronization: Maintain temporal alignment across video and sensor streams.

# 4. Software and Hardware Requirements

## 4.1 Software Requirements

- OS: Ubuntu Linux  
- Compiler: GCC  
- Tools: ipcs, ipcrm, shmget, semget, msgget, socket, pthread  
- Language: C (with POSIX threads and IPC)

## 4.2 Hardware Requirements

- Simulated aircraft video and sensor data (text-based or emulated)  
- No physical aircraft or cameras required for simulation

# 5. System Overview (Process-Based)

Process Flow:  
1. Data Acquisition Threads: Capture video and sensor data concurrently.  
2. Shared Memory Manager: Stores sensor data for access by multiple processes.  
3. Video Processor: Stabilizes and enhances video frames.  
4. Sensor Overlay Module: Adds altitude, speed, and GPS overlays to video.  
5. Object Tracker: Detects and tracks moving objects across feeds.  
6. Encoder: Compresses video and sensor data for transmission.  
7. Streaming Client: Sends encoded data to the display server.  
8. UI Dashboard (TCP Server): Displays synchronized video and sensor overlays.  
9. Watchdog & Signal Handler: Monitors system health and handles failures.

# 6. Constraints

- Each module must release IPC resources upon exit using cleanup.c.  
- Manual launching of modules in separate terminals is required.  
- Simulated data is used for development and testing.

# 7. Appendices

## A. Assumptions

- Users will run each module manually in separate terminals.  
- Simulated video and sensor data are acceptable for testing.

## B. Glossary

- Frame: A single image or data block from a video stream.  
- Client: The UI dashboard receiving and displaying processed data.  
- Sensor Data: Includes altitude, speed, and GPS coordinates.  
- Watchdog: A monitoring thread that detects and recovers from failures.