1. Experimental Setup / Simulation, Performance Parameters, and Efficiency Issues (10 Marks)

Experimental Setup / Simulation

The proposed system is designed to perform **real-time face recognition for surveillance** using drone-captured aerial images and videos. The overall experimental setup is divided into two primary segments:

A. Aerial Data Acquisition Module (Drone Unit)

- A drone equipped with a high-resolution RGB camera is used to capture aerial images and video footage from varying altitudes and angles.
- The drone is connected to the ground control station through a secure wireless communication channel (e.g., Wi-Fi).
- It continuously streams image/video data to the ground system for real-time processing.

B. Ground Processing Unit (Face Recognition System)

- A workstation or edge device receives the drone data stream.
- The system processes incoming frames using the **face detection and face recognition pipeline**.
- Based on the expected number of people, the system dynamically switches between:
    - **Single-Person Recognition Module:** Optimized for identifying a single individual in a close-up image (faster, less computationally expensive).
    - **Multi-Person Recognition Module:** Designed to handle group scenarios, detects all faces in an image, and performs recognition for each detected face (more computation, higher parallel processing).
- Recognition results (e.g., identified names or “unknown” labels) are stored locally in a database for record-keeping and future retrieval.

This setup simulates a real-world security/surveillance situation where drones are patrolling an area, capturing data, and sending it for real-time identification.

Performance Parameters

Several key performance indicators are being considered to evaluate the system:

- **Detection Accuracy:**
  Measures the percentage of faces successfully detected from input frames. High accuracy is crucial for effective surveillance.

- **Recognition Accuracy:**
  Percentage of correctly recognized faces out of the total detected faces. It indicates the reliability of the recognition pipeline.
- **Processing Latency:**
  The average time taken to process a single frame from input to output. This affects real-time responsiveness.
- **Frame Processing Rate (FPS):**
  Number of frames processed per second. Higher FPS ensures smooth real-time operation.
- **Scalability:**
  Ability of the system to handle an increasing number of faces in an image/video frame without significant performance degradation.
- **False Positive / False Negative Rates:**
  Important for evaluating how often the system wrongly recognizes or fails to recognize known faces.

Efficiency Issues

Some efficiency-related challenges currently observed:

- **High Computational Load for Multi-Face Frames:**
  Processing images with multiple faces significantly increases computation time and resource usage.
- **Network Bandwidth Dependency:**
  Real-time video streaming from drone to ground system can suffer from latency or quality degradation in low-bandwidth environments.
- **Lighting and Angle Variability:**
  Aerial imagery often has inconsistent lighting, shadows, or side profiles which may lower recognition accuracy.
- **Model Generalization:**
  Recognition accuracy may decrease if the faces in the dataset differ significantly from real-time captured faces (pose, distance, resolution).
- **Storage Constraints:**
  Continuous storage of high-resolution frames and recognition results requires efficient data management and compression.

2. Tools and Techniques Used (10 Marks)

The project uses a combination of **software frameworks, programming libraries, and techniques** for image processing, face detection, and face recognition:

- **Programming Language:**
  - Python (due to its strong ecosystem for image processing and machine learning)
- **Libraries and Frameworks:**
  - **OpenCV:** For image and video frame processing, drawing bounding boxes, face region extraction, and handling video streams.
  - **face_recognition (dlib-based):** For face detection (HOG/CNN-based) and face recognition (128-dimensional face embeddings using deep metric learning).
  - **NumPy and Pandas:** For efficient data handling, embedding storage, and computations.
  - **SQLite / Pickle Files:** For maintaining known face encodings and associated identity data.
- **Techniques and Methodologies:**
  - Face Detection using HOG (Histogram of Oriented Gradients).
  - Face Embedding generation using pre-trained deep learning models.
  - Face Matching using Euclidean distance between embeddings.
  - Modular architecture to split processing between single-person and multi-person pipelines to optimize performance.
  - Real-time video stream handling using frame-by-frame processing.
  - Preprocessing techniques like face alignment, resizing, and color normalization.
- **Hardware Tools:**
  - Drone with mounted high-resolution camera.
  - Ground system (laptop/desktop) with GPU acceleration for faster processing.

3. Implementation Status (30 Marks)

Current Progress: ~60% Complete

Completed (~60%)

1. System Architecture and Module Setup

- Designed a modular architecture with two separate pipelines:
  - **Single-Person Recognition Module:** optimized for close-up recognition.
  - **Multi-Person Recognition Module:** for detecting and recognizing multiple faces simultaneously.
- Set up the directory structure, data flow, and switching mechanism between the two pipelines.

2. Data Acquisition and Communication

- Configured the drone camera to capture high-resolution images and video.
- Established real-time data transfer from drone to ground processing system via wireless communication.
- Integrated OpenCV to receive and handle video streams on the ground system.

3. Basic Face Detection & Recognition Pipeline

- Implemented face detection using the face_recognition library (HOG-based model for speed).
- Generated face embeddings (128-D) for known individuals and stored them.
- Implemented initial face matching logic using Euclidean distance comparison.

4. Initial Database of Known Faces

- Created a structured local database of known individuals with labeled images.
- Enabled adding new entries to the database for continuous learning.

5. Preliminary Testing & Validation

- Conducted small-scale tests with sample datasets and real drone-captured images.
- Verified that detected faces are recognized correctly and labeled in the output frames.
- Recorded initial performance metrics like FPS, latency, and detection accuracy.

In Progress / Planned (~40%)

1. Advanced Accuracy Optimization

- Improving handling of difficult conditions (lighting, side profiles, varying distances).
- Adding face alignment and preprocessing techniques for better recognition accuracy.

2. Multi-Person Recognition Speed Optimization

- Parallelizing the recognition pipeline to reduce processing time when multiple faces are present.
- Profiling computational bottlenecks to improve FPS and reduce latency.

3. Surveillance-Specific Enhancements

- Automating drone flight paths for continuous area coverage.
- Adding time, date, and GPS metadata to each captured frame.

4. UI and Alert System

- Designing a dashboard to show real-time recognized faces, timestamps, and identity tags.
- Planning to integrate real-time alert generation (notifications on detecting known/suspect faces).

5. Final System Integration and Field Testing

- Integrating all modules into a single unified system for real-world use.
- Conducting large-scale testing under real surveillance conditions.