# AI –Video Object Detection

## Problem Statement

Video object detection involves finding and identifying objects in each frame of a video and optionally tracking them over time. Unlike still image detection, videos require handling moving objects, camera motion, and sometimes blurry frames.  
  
This project will help beginners learn how to apply basic object detection techniques to video input.

## Objective

Use beginner-friendly object detection models to detect objects in video files or webcam streams, compare their results, and understand their strengths and weaknesses.  
Suggested beginner-level approaches:  
- YOLOv5 Pre-trained Model – Run frame-by-frame detection on video.  
- MobileNet SSD – Lightweight and fast for low-compute environments.  
- Haar Cascade Classifiers – Traditional method for simple object detection (faces, eyes, etc.).  
(Optional for tracking):  
- OpenCV Tracking API – Track detected objects between frames.

## Dataset

You can use:  
- Sample videos from OpenCV (e.g., pedestrian detection video).  
- Any short video from YouTube (download and use for testing).  
- Custom recorded videos from your phone or webcam.

## Acceptance Criteria

- At least 2 different beginner-friendly detection methods implemented.  
- Model outputs saved as:  
 - Annotated videos (bounding boxes drawn around detected objects).  
 - Summary of detection counts per object type.  
- Basic performance comparison between models:  
 - Detection accuracy (visual inspection is enough for beginners).  
 - Processing speed (frames per second).

## Project Milestones (30 Days)

|  |  |  |
| --- | --- | --- |
| Milestone | Duration | Description |
| Kick-off & Setup | 1 day | Install Python, OpenCV, PyTorch, and download models |
| Dataset / Video Selection | 3 days | Find or record sample videos |
| Model 1: YOLOv5 Implementation | 1 week | Load pre-trained YOLOv5, run on video |
| Model 2: MobileNet SSD Implementation | 1 week | Load MobileNet SSD, run on video |
| Evaluation & Comparison | 3 days | Compare visual results & FPS |
| Final Presentation | 3 days | Show demo videos & results in PPT |

## Protocols

- Use pre-trained models to avoid heavy training.  
- Keep videos short (10–30 seconds) for faster processing.  
- Submit:  
 - Python code files  
 - Annotated output videos  
 - A short PPT summarizing results