Cricket Ball Tracking System – Detailed Report

1. Problem Overview The goal of this project was to build a robust cricket ball tracking system for broadcast-style videos. The system outputs a processed video showing the ball trajectory and frame-wise annotations indicating the ball's position and visibility. Major challenges included the small size of the ball, motion blur, occlusions, and visually similar white objects such as shoes, gloves, pads, crease lines, and umpire accessories.

2. Initial Baseline: YOLO-only Detection The first version of the system relied solely on YOLOv8 (pretrained on COCO) to detect the sports ball class. For each frame, the highest-confidence sports ball detection was selected and its center was treated as the ball position.

Issues Observed: - Frequent false positives on shoes, gloves, pads, and crease lines - Inconsistent detections across frames - Lack of temporal continuity

3. Spatial Constraint: Cropping the Processing Frame Cricket broadcasts follow a predictable camera layout. To leverage this domain knowledge, detection was restricted to a fixed region of interest (ROI): - Width: 10% to 90% of frame width - Height: 15% to 45% of frame height

Impact: - Removed most ground-level false positives - Reduced detection noise significantly - Improved efficiency by processing fewer pixels

4. Introducing Temporal Tracking with Kalman Filter To improve stability and continuity, a Kalman filter was introduced. The filter smoothed frame-to-frame detections and allowed short gaps in detection to be bridged, resulting in a smoother trajectory.

5. Problems Introduced by Naive Tracking Unconstrained tracking caused drift when the ball was not visible and false lock-on if initialized incorrectly. This highlighted the need for stricter control over when tracking should be active.

6. State-Based Tracking Logic The tracker was redesigned as a state-driven system: - Unconfirmed: detections exist but tracking is not trusted yet - Confirmed: consistent detections observed, tracking enabled - Lost: ball missing for too long, tracker reset

7. Fallback Logic A hierarchical fallback mechanism was used: - Primary detection via YOLO - Secondary fallback using color-based detection (HSV and circularity)

8. Visualization Strategy - Blue dots show past confirmed ball positions - Green dot shows the current ball position - No bounding boxes or ROI overlays were drawn

9. YOLO-only Ablation Study A YOLO-only version was implemented to validate design choices. This eliminated drift but produced noisier trajectories, confirming the benefit of temporal tracking for smoothness.

10. Final System Characteristics Strengths: - Robust against white-object false positives - Stable and interpretable tracking - Clean reset and recovery behavior

Assumptions: - Camera remains largely static - Ball appears within a predictable region - Broadcast-style cricket footage

11. Conclusion The final system evolved from a simple object detector into a domain-aware tracking pipeline through careful iteration. Spatial constraints, state-based logic, and fallback mechanisms significantly improved reliability while keeping the system efficient and explainable.