Technical Report: Player Tracking in Cricket Footage

Date: 9 Dec, 2025

1. Executive Summary

This project implements a computer vision pipeline to detect cricket players and maintain unique IDs across broadcast footage. The solution leverages **YOLOv5** and **DeepSORT**, designed specifically to handle sports-specific challenges like occlusion and rapid motion within the constraints of Google Colab's free tier (Tesla T4 GPU).

2. System Architecture

2.1 Object Detection: YOLOv5s

- **Model:** YOLOv5s (Small) pretrained on COCO.
- **Configuration:** Filtered for **Class 0 (Person)** only.
- **Justification:** Selected for the optimal balance between inference speed (~6ms) and accuracy (mAP), preventing timeout errors common with larger models on cloud runtimes.

2.2 Tracking Algorithm: DeepSORT

We utilized DeepSORT to assign and maintain consistent IDs, integrating two core mechanisms:

1. **Kalman Filtering:** Predicts player positions based on velocity, bridging gaps when detection fails momentarily.
2. **Visual Re-Identification (ReID):** Uses deep appearance embeddings to re-assign IDs to players returning to the frame after leaving.

2.3 Handling Occlusion (ID Consistency)

To address players passing behind umpires or wicket-keepers, we tuned the tracker memory:

- **Setting:** max_age=30 frames.
- **Impact:** This provides a **1-second buffer** (at 30 FPS). It is long enough to maintain the ID during temporary occlusion but short enough to prevent "Zombie Tracks" (assigning old IDs to new players) if a player genuinely leaves the field.

3. Challenges & Technical Solutions

Challenge	Technical Solution
Runtime Disconnects	**Chunk-wise Processing:** Video processed in small batches saved to Drive. A custom **Resume Logic** module detects the last saved chunk and fast-forwards to the exact frame to continue.
Drive Latency	**Local Merging:** To prevent file corruption from network lag, chunks are copied to the local VM (/content) before FFmpeg concatenation.
Memory Constraints	**Stream-based Processing:** Frames are processed sequentially without accumulating in RAM. The final merge uses FFmpeg's copy codec to stitch videos without re-encoding.

4. Limitations

1. **Crowd Occlusion:** In tight huddles, Intersection-over-Union (IoU) metrics fail, and visual features are obscured, leading to occasional ID flickering.
2. **Resolution Trade-off:** Input frames are resized (scale 0.75) to maintain real-time processing, slightly reducing ReID accuracy for distant players.
3. **Detector Recall:** The lightweight YOLOv5s model occasionally misses small objects in wide-angle shots, breaking tracks if the miss exceeds 1 second.

5. Future Improvements

- **Stronger ReID Backbone:** Upgrade DeepSORT's feature extractor to **ResNet50** to better distinguish players in identical uniforms.
- **Offline Global Tracking:** Implement post-processing to optimize trajectories across the entire video timeline rather than frame-by-frame, smoothing out erratic paths.