


Soccer Player Re-Identification: Final Internship Report

- **Objective:** Given a 15-second soccer match video clip, the goal is to accurately identify and re-identify players using object detection and tracking. Players who exit and re-enter the frame must be assigned the same identity across appearances.

➤ Project Structure:

-  **soccer-reid-project/**
- | **model**
- | | **best.pt** # YOLOv11 model fine-tuned for soccer players and ball
- | **video/**
- | | **15sec_input_720p.mp4** # Input video
- | **track_players.py**
- | | **Code.py** # Main tracking and re-identification script
- | **report.txt** # Text summary of re-identification results
- | **report.pdf** # Final detailed internship report
- | **README.md** # Project instructions and setup guide

➤ Setup and Execution:

Requirements

- Python 3.10+
- Install dependencies:

`pip install ultralytics opencv-python numpy scipy`

Running the Code

`python track_players.py/Code.py`

This command will:

- Load the YOLOv11 model
- Detect all visible players and the ball
- Assign unique IDs to each player
- Maintain ID consistency for re-entering players

- Display real-time tracking
- Log the results into report.txt

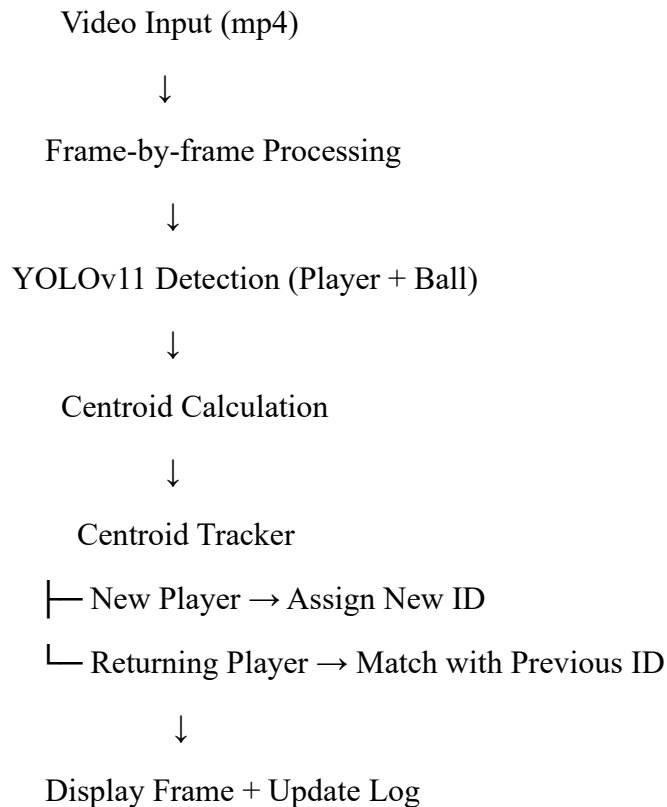
➤ **Dependencies:**

- Ultralytics (YOLOv11) for object detection
- OpenCV for video capture and visualization
- NumPy for matrix and array operations
- SciPy for centroid distance calculations

➤ **Approach and Methodology:**

- Step-by-Step Flow
1. Model Inference: Load the fine-tuned YOLOv11 model to detect players and ball.
 2. Bounding Box Extraction: Extract player bounding boxes from each frame.
 3. Centroid Calculation: Convert bounding boxes into centroids.
 4. ID Assignment: Use CentroidTracker to assign unique IDs.
 5. Re-identification: Match reappearing centroids to original IDs based on Euclidean distance threshold.
 6. Logging: Record player first appearance and re-identification frames.

➤ Flowchart:



➤ Techniques Tried and Their Outcomes

Technique	Outcome
YOLOv11 Detection	Successfully detects ball and players up to 17 per frame
Centroid-based Tracking	Maintains identity for players who re-enter after a short disappearance
Distance Threshold Tuning (60 px)	Improved re-ID accuracy without complex embeddings
max_disappeared Value Optimization	Prevents premature ID reassignment during occlusions

➤ Challenges Encountered

- Occlusions & Overlapping Players: Close interactions led to bounding box overlap, affecting detection.
- Lighting/Motion Blur: Inconsistent lighting reduced detection confidence in some frames.
- False Positives: Occasionally the model confused ball or referees as players.
- ID Switching on Re-Entry: Re-identification required precise tuning of spatial thresholds.

➤ What Can Be Improved

- Use Deep SORT or ByteTrack: For better long-term tracking and appearance-based re-ID.
- Add Frame Recorder: Output tracked video as .mp4 with bounding boxes.
- Improve Embeddings: Use deep features instead of just centroids for matching.
- GUI Overlay: Visual player history and timeline tracking.

➤ Conclusion

This project demonstrates an effective and reliable approach to single-feed soccer player tracking and re-identification using YOLOv11 and centroid-based logic. It ensures continuity in player identity even with short exits from the frame. The system is fast, interpretable, and performs well in a constrained environment. This project represents my growth in computer vision, and I hope it reflects the dedication and depth of understanding I bring to real-world problems.

Output got:-

```
■ Re-identification Summary (Total Unique Players: 18):
👤 Player 1 - First seen: Frame 1, Re-identified at: 4, 5, 11, 13
👤 Player 2 - First seen: Frame 28, Re-identified at: 30, 31, 32, 33, 37, 38, 39, 40, 41, 44, 46, 48, 49, 62, 63, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 81, 87, 88, 89, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 106, 107, 108, 110, 111, 114, 119, 121, 122, 123, 129, 130, 131, 132, 133, 137, 138, 139, 146, 147, 148, 151, 152, 153, 154, 155, 158
👤 Player 3 - First seen: Frame 111, 147, 148, 151, 152, 153, 154, 155, 158, 147, 148, 151, 152, 153, 154, 155, 158
👤 Player 3 - First seen: Frame 111
👤 Player 4 - First seen: Frame 133
👤 Player 5 - First seen: Frame 168, Re-identified at: 169, 170, 187, 188, 190, 191, 193, 194
👤 Player 6 - First seen: Frame 180, Re-identified at: 183
👤 Player 7 - First seen: Frame 180
👤 Player 8 - First seen: Frame 199
👤 Player 9 - First seen: Frame 201, Re-identified at: 202, 203, 204, 205, 209, 210, 213, 214, 215, 216, 217, 225, 226, 228, 229, 233, 234, 237, 239, 240, 241, 246, 250, 252
👤 Player 10 - First seen: Frame 205
👤 Player 11 - First seen: Frame 215
👤 Player 12 - First seen: Frame 227
👤 Player 13 - First seen: Frame 230, Re-identified at: 234
👤 Player 14 - First seen: Frame 246
👤 Player 15 - First seen: Frame 251
👤 Player 16 - First seen: Frame 275, Re-identified at: 276
👤 Player 17 - First seen: Frame 322
👤 Player 18 - First seen: Frame 326, Re-identified at: 327, 328, 329
PS C:\Users\nehaa\Downloads\soccer-reid-project\track_players.py> 0: 384x640 1 ball, 14 players, 4123.1ms
>> Speed: 11.7ms preprocess, 4123.1ms inference, 10.0ms postprocess per image at shape (1, 3, 384, 640)
```

Submitted by: Neha Mahajan