**Soccer Player Re-Identification Assignment**

**Option 2: Re-Identification in a Single Feed**

**In this assignment I have used 2 different approaches :**

**Approach 1: Naive approach (without any algorithms)**

* The video is of 15 seconds, which has 25 FPS, and hence **15 x 25 = 375** being the total number of frames.
* After going through the video frame-by-frame , all the players in each frame are detected using the **YOLOv11 model.**
* Each player is given a **bounding box**, which gives us a set of values of **(x, y, width, height)** for each detected player. This helps us in identifying the position of the player.
* Center of the box is calculated as **((x1 + x2) / 2)**  and **((y1 + y2) / 2)**  and returned.
* Each player is assigned a new ID.
* The **ID, center position, jersey color, bounding box and last frame number when the player was seen** are all stored in a dictionary called **“detected\_players”.**
* For every detected bounding box, three things are checked and compared, i.e., is the detected object a **player, referee or a ball**  and then extracts the bounding box and its center coordinates.

**➤ If it's a player:**

* Estimate jersey color (red or blue) from cropped region.
* Check all existing **detected\_players** and try to match based on **center proximity.**

If matched (close enough):

* Reuse the same ID, update bbox, center, and last\_seen.

If not matched:

* Assign a new ID from **vacant\_ids**, store it in **detected\_players.**

**➤ If it's a referee:**

* If **referee\_info** is empty (i.e., not assigned yet), assign it now and mark it permanently.
* On future frames, track using center similarity and update **referee\_info** accordingly.
* Always label as "Referee", never change the label or treat it as a new object.

**➤ If it's a ball:**

* If **ball\_info** is empty, assign and store it.
* In later frames, update its position based on proximity.
* Always label as "Ball", don't reassign.
* The bounding boxes are drawn and labelled.
* The tracking results are saved and released and the video obtained as a result is saved.

**Results and challenges faced:**

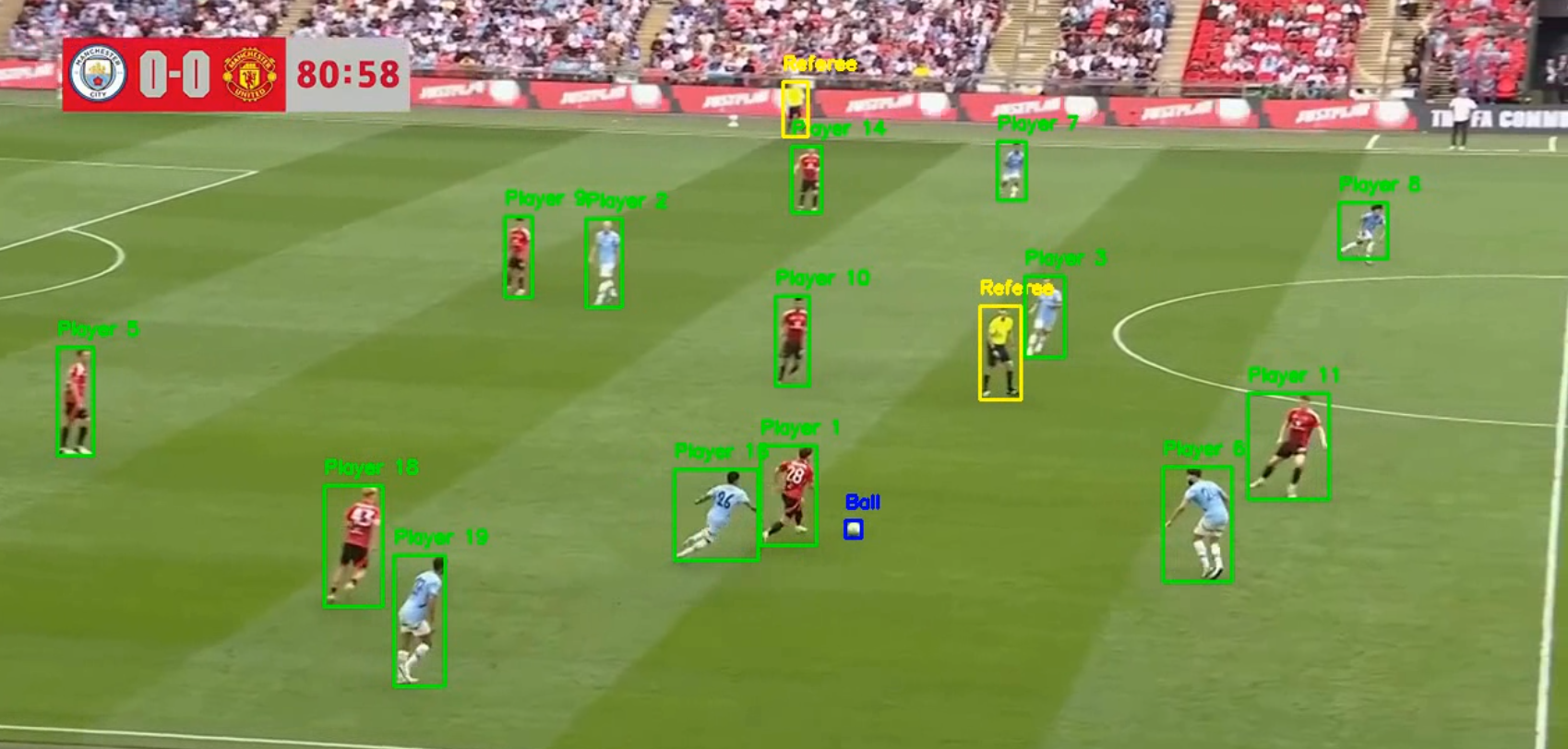
**Results:**

* Each player gets a unique, fixed ID (like “Player 1”, “Player 2”, etc.) and they are constant across all frames.
* Players, referees, and the ball are detected separately. The **ball** is labelled as “Ball” and tracked consistently and the referees is labelled only once and remains fixed (although not in all frames).
* Each detection is annotated on the video with **green (players)**, **yellow (referee)**, and **blue (ball)** bounding boxes.
* When a player disappears temporarily (occlusion or out of frame) and reappears, their ID was properly reassigned based on **proximity and jersey color** and also based on if the player ID already existed in **detected\_players** list or not, which helped in reducing ID duplication or switching slightly.

**Challenges:**

* Despite efforts to fix IDs, if two players **collide or overlap**, the tracker gets confused with them and **reassigns wrong IDs** and **switches IDs between players.**
* **YOLO** bounding boxes fluctuated slightly across frames, causing jitter in labels while affecting center-based matching. This made IDs unstable when players were close.
* If the detection threshold is set too low (DETECTION\_THRESHOLD = 0.3), YOLO might detect benches, crowds, etc. as players/ball/referee. If set too high, real players or the ball may be missed.
* Since we’re relying only on bounding box center and jersey color but not using face, body or number features, similar-looking players in close proximity may confuse the tracker.

**Tried technique:** Tried using OCR to read the jersey numbers of the players in order to maintain a consistent ID for the players, but failed because the jersey numbers were visible only for a few players and not all.



**Approach 2: With algorithms**

**1.DeepSORT + Kalman Filtering**

1. **DeepSORT function :** Assigns consistent IDs automatically using **motion + appearance,** Initializes tracker with **Kalman filter** and **MobileNet.**
2. **MobileNet :** Designed specifically for efficient **image classification and feature extraction.**
3. **Kalman Filter:** Mathematical algorithm used to predict and update the **position of a moving object** over time, even when the measurements are noisy or uncertain.

* Used **YOLOv11** to detect players, referees, and the ball.
* Used **DeepSORT** (with Kalman Filter + MobileNet) to assign consistent IDs to players.

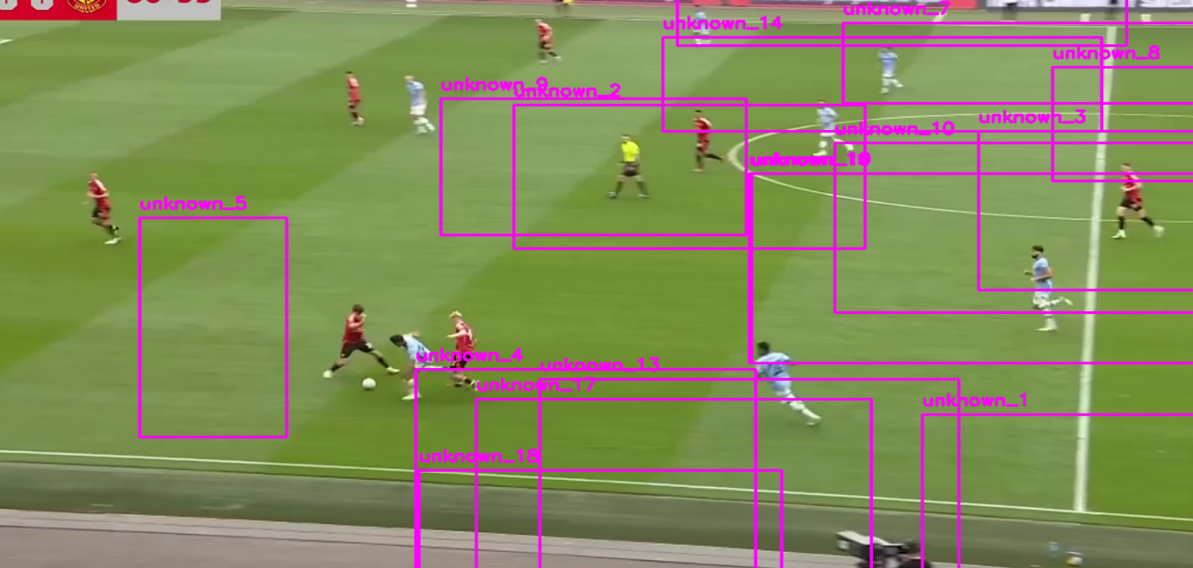
**Results and challenges faced:**

* YOLO misclassifies players as referee or ball (wrong class label), or sometimes even as “unknown”.
* Bounding boxes were too big, including grass/background.
* DeepSORT sometimes gave new IDs when players got occluded or re-entered the frame.

**Tried techniques :**

* Jersey color detection using HSV (to tell red team vs blue team).
* **IoU** – Intersection Over Union (overlap) matching to correctly link detections with tracking.
* **Fixed ID system**: Once a player gets an ID, we keep it the same across frames.

This method worked okay but had issues because the YOLO model’s training was limited. Since I couldn’t retrain it, I worked around it using tracking + color logic.



**2.ByteTrack**

**ByteTrack** is a fast and simple tracking algorithm used in computer vision to track multiple objects (like people, cars, players, etc.) across video frames.

**Results**

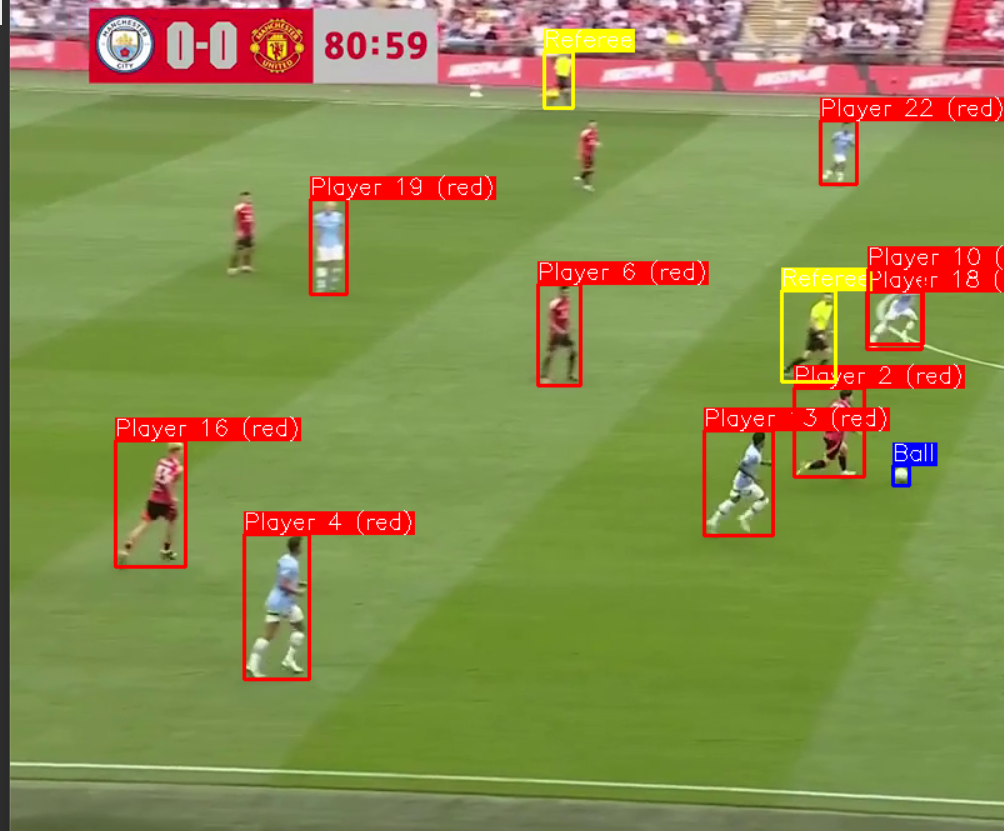
* Players, referees, and the ball were detected with some unique IDs.
* Tracking worked to some extent when players were apart and not overlapping.
* ByteTrack ran faster than DeepSORT and it used little less resources.

**Challenges**

* IDs still switched or increased when players overlapped or left and re-entered the frame.
* No appearance embedding in ByteTrack made it hard to maintain consistent IDs.
* Jersey color-based labelling worked for some players but failed for others.
* YOLO’s bounding box shifts caused instability in center-based tracking.
* Ball and referee were confused with players when detection confidence was low.

**Overall Takeaway**

* Neither ByteTrack nor DeepSORT gave a fully reliable solution for consistent, fixed player ID tracking.

For proper player tracking, we need better ReID features or fine-tune the model — which wasn’t feasible in this case due to limitations with the pre-trained model.

Although I have tried 2 different approaches, I have uploaded the code and results of only the one which gave me the most optimal results i.e., the first approach.

**What could be done with more time and resources ?**

1. Use a better pre-trained model with clearer player/referee/ball detection.
2. Improve jersey color detection using smarter image techniques.
3. Use OCR or other methods to read jersey numbers when visible.
4. Add face/body/jersey features for stronger ID tracking.
5. Test on longer or different quality videos.
6. Handle ID switching better with correction logic or tracking history.
7. Build a small system to track player positions and stats across the match.
8. Explore more better solutions.