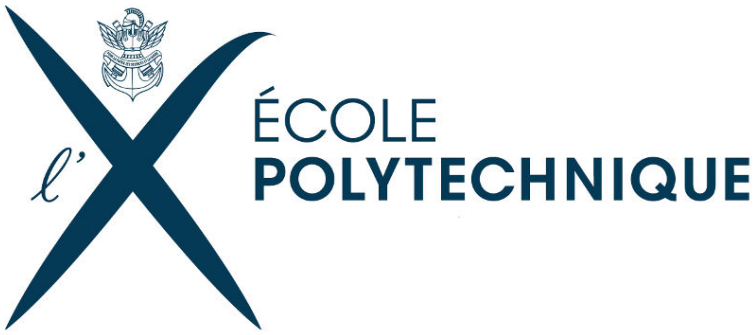

Basketball Tracking with Computer Vision

CSC_51073_EP – Analyse d'Image et Vision par Ordinateur

BY

Prakhar Tiwari

Sarah Rouphael

TABLE OF CONTENTS

LIST OF FIGURES

1 Introduction

Understanding broadcast basketball videos is not an easy computer vision task. The game involves fast player movements, quick camera changes, frequent occlusions, and crowded scenes, all of which make visual analysis difficult. More broadly, recent surveys highlight that such challenges are common in sports video analysis and motivate the growing use of AI-based computer vision techniques in the sports industry [2]. In order to analyze a basketball match properly, important elements such as the players, the ball, and the court need to be detected and tracked across time. However, in real broadcast footage, this is far from straightforward. Small and fast-moving objects like the ball are often missed by detectors, players can look very different from one frame to another, and classical geometry-based methods for court estimation are highly sensitive to noise and imperfect segmentations.

In this project, we explore different approaches for analyzing broadcast basketball videos, with a focus on detecting and tracking players and the ball, assigning players to teams, and estimating the visible court region. Instead of aiming for a fully perfect solution, the goal of this work is to understand the practical challenges that arise when applying existing computer vision techniques to real broadcast data. Multiple methods were implemented and tested, and many of them showed clear limitations under realistic conditions. These failures and observations guided iterative changes to the pipeline and helped us better understand which approaches work in practice, which ones do not, and the reasons behind their success or failure.

2 Related Work

Prior work on broadcast sports analytics often follows the same tracking-by-detection paradigm adopted in this project: key objects such as players, the ball, and the hoop are detected frame-by-frame using a fast object detector, linked over time with a multi-object tracker, and then exploited for higher-level reason-

ing such as team assignment and court geometry estimation. For tracking, many pipelines combine a detector with DeepSORT, which extends SORT by incorporating appearance-based association to reduce identity switches under occlusions, an issue that is particularly prominent in basketball footage, as shown by [8].

Our team detection strategy was additionally inspired by a computer-vision pipeline described in a Roboflow blog post[4]. While not a peer-reviewed source, this reference provided useful implementation insights that shaped our design choices for team assignment.

Inspired by prior work on learning basketball motion patterns from trajectory data [6], we investigated ball trajectory modeling. However, since their approach relies on deep learning and our goal was to keep the pipeline lightweight and training-free, we instead used interpolation to fill short gaps in the detected ball track.

For court understanding, the literature commonly formulates the problem as sports field or court registration, where a planar homography or camera calibration is estimated from visual cues. We attempted to implement the approach proposed by Sha et al. [5]; however, the method did not perform reliably from its initial segmentation stage on our dataset. Despite this limitation, the paper strongly influenced our design choices and motivated the simplified color-based court segmentation strategies described in Section 4.

Finally, we selected YOLOv8 for player and ball detection because recent surveys on ball detection in sports highlight YOLO-based detectors as a strong practical choice due to their real-time performance and robustness to small, fast-moving objects [1], and recent studies further analyze YOLOv8's architecture and effectiveness in demanding detection settings [3].

3 Methods and Results

3.1 Dataset, YOLO Fine-tuning, and DeepSORT

3.1.1 Dataset and Model Training

We fine-tuned two separate YOLOv8-medium models using datasets from Roboflow Universe to address the distinct detection challenges. The finetuning parameters were inspired based on the repository made publicly available by Tarek [7].

1. **Player Detection Model (`player.pt`):** Trained on the `basketball-mjp0n` dataset to detect players (class 0). This model handles partial occlusions, varying player poses, and scale variations across camera distances. The dataset obtained from Roboflow consisted of 2021 training images, 577 validation images, and 289 test images.

   Model Configuration: We initialized training from the YOLOv8m (medium) pretrained weights, which provide a balanced trade-off between detection accuracy and inference speed. The model was trained to detect a single class (players, class 0).

Table 1: Player Detection Model: Training Configuration

Parameter	**Value**
Base Model	YOLOv8m (`yolov8m.pt`)
Input Resolution	640×640 pixels
Epochs	200
Batch Size	16
Optimizer	AdamW (YOLOv8 default)
Initial Learning Rate	0.01 (cosine annealing)
Data Workers	4 parallel workers
Target Classes	1 (players)

Training Results: The final trained model (`player.pt`) achieved the fol-

lowing performance metrics on the validation set (Table 2). These metrics indicate strong detection performance with high precision and recall, demonstrating the model's ability to reliably detect players across varying conditions while maintaining low false positive rates.

Table 2: Player Detection Model: Validation Performance

Metric	**Value**
Precision	92.79%
Recall	89.76%
mAP@0.5	92.96%
mAP@0.5:0.95	72.84%
Inference Speed (per image)	
Preprocessing	0.18 ms
Inference	9.71 ms
Postprocessing	0.77 ms
Total	**10.66 ms (∼94 FPS)**

2. **Ball and Hoop Detection Model (`ball_hoop.pt`):** Trained on the `basketball-xptln` dataset to detect the basketball (class 0) and the hoop (class 1). This model was specifically optimized for small-object detection to handle the fast-moving, frequently occluded ball. The dataset obtained from Roboflow consisted of 533 training images, 80 validation images, and 80 test images.

   Model Configuration: Similar to the player model, we used YOLOv8m pretrained weights as initialization. The smaller size and higher velocity of basketballs compared to players motivated specific training considerations for this model.

Table 3: Ball and Hoop Detection Model: Training Configuration

Parameter	**Value**
Base Model	YOLOv8m (`yolov8m.pt`)
Input Resolution	640×640 pixels
Epochs	200
Batch Size	16
Optimizer	AdamW (YOLOv8 default)
Initial Learning Rate	0.01 (cosine annealing)
Data Workers	4 parallel workers
Target Classes	2 (ball, hoop)

Training Results: The final trained model (`ball_hoop.pt`) achieved the validation metrics shown in Table 4. The lower mAP@0.5:0.95 compared to the player model reflects the inherent difficulty of small object detection, where precise localization (required for high IoU thresholds) is more challenging. The relatively lower recall (77.64%) indicates that the ball is occasionally missed, particularly during fast motion, severe occlusions, or when the ball appears very small in the frame.

Table 4: Ball and Hoop Detection Model: Validation Performance

Metric	**Value**
Precision	89.90%
Recall	77.64%
mAP@0.5	86.08%
mAP@0.5:0.95	46.75%
Inference Speed (per image)	
Preprocessing	0.11 ms
Inference	5.67 ms
Postprocessing	0.54 ms
Total	**6.32 ms (∼158 FPS)**

The decision to use separate specialized models rather than a unified multi-class detector was motivated by the distinct detection challenges: players are large and numerous, while the ball is small, singular, and requires higher sensitivity. This specialization allowed each model to optimize for its specific task without performance trade-offs.

3.1.2 DeepSORT Tracking

To maintain consistent player identities across frames, we integrated DeepSORT [8], which extends SORT (Simple Online and Realtime Tracking) by incorporating deep appearance features alongside motion prediction via Kalman filtering, which significantly reduces identity switches during occlusions—a frequent occurrence when players cross paths or cluster near the basket. The hyperparameters for the DeepSORT algorithm are mentioned in Table 5.

Table 5: DeepSORT Configuration Parameters

Parameter	**Value**
Maximum Age (`max_age`)	30 frames
Minimum Initialization (`n_init`)	3 frames
Motion Model	Kalman filter (constant velocity)
Association Metric	Motion + Appearance (weighted)
Appearance Features	Deep CNN embeddings
Distance Metrics	Mahalanobis + Cosine distance

For each frame, player detections are associated with existing tracks using both motion prediction and appearance similarity computed from bounding-box crops. While this approach substantially improved tracking consistency compared to motion-only methods, challenges remained during prolonged occlusions or when players with similar appearances (e.g., identical team jerseys) came into close proximity.

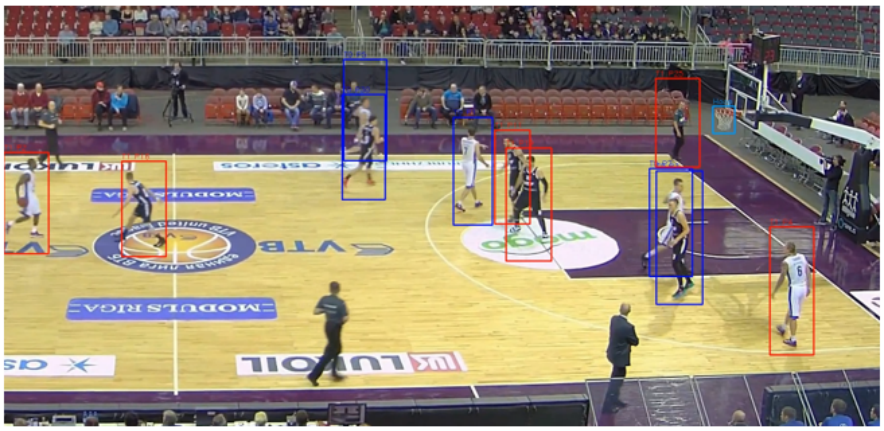

Figure 1: Yolo Tracking players after fine-tuning.

3.2 Ball trajectory post-processing

3.2.1 Parabolic trajectory estimation (initial approach)

As an initial attempt to compensate for imperfect YOLO ball detections, we modeled the ball motion using a physics-inspired parabolic trajectory. The horizontal motion was approximated as a linear function of time, while the vertical motion was modeled as a quadratic function to account for gravity. Model parameters were estimated using least-squares fitting over a short window of recent ball detections, and the fitted trajectory was used to predict future ball positions when detections were missing. Although this approach helped smooth some detection noise, it proved unreliable in practice. Broadcast basketball footage frequently violates ideal projectile assumptions due to camera motion, perspective effects, occlusions, and noisy detections. As a result, the predicted trajectories were often unstable or visually inaccurate. For these reasons, this approach was not retained in the final system.

3.2.2 Interpolation-based gap filling (final approach)

To address missed detections in a simpler and more robust way, we replaced the parabolic model with an interpolation-based strategy. When the ball is not detected, frames are temporarily buffered instead of being written directly to the

output. Once the ball is detected again, the ball centers from the last detection before the gap and the first detection after the gap are used to linearly interpolate the ball position for each buffered frame based on time. A red ball marker is drawn at these interpolated positions, producing a smooth and continuous trajectory across short detection gaps. In addition, the last valid YOLO ball bounding box is kept for a short duration when detections are missing, providing a visual fallback and making detection failures explicit. If the ball remains undetected for too long, interpolation becomes unreliable and the buffered frames are flushed without interpolation. This final approach avoids strong physical assumptions and provides a stable, visually consistent representation of the ball trajectory despite imperfect detections.

3.3 Team Detection

3.3.1 Edge-based player segmentation for team detection (discarded)

We initially explored an edge-based segmentation approach to support team detection by isolating player silhouettes within YOLO bounding boxes. For each detected player, Canny edge detection followed by morphological operations was used to extract and fill the largest contour, producing a binary player mask intended for cleaner appearance feature extraction. In practice, this method proved unreliable. Player edges were often weak or fragmented, and background structures frequently interfered with contour extraction, leading to incomplete or inaccurate player masks. As a result, the extracted features were too noisy for consistent team classification, and this approach was not retained in the final pipeline

3.3.2 GrabCut-based player segmentation and color clustering (discarded)

We also explored a GrabCut-based segmentation approach to improve team detection by isolating the player from the background within YOLO bounding boxes.

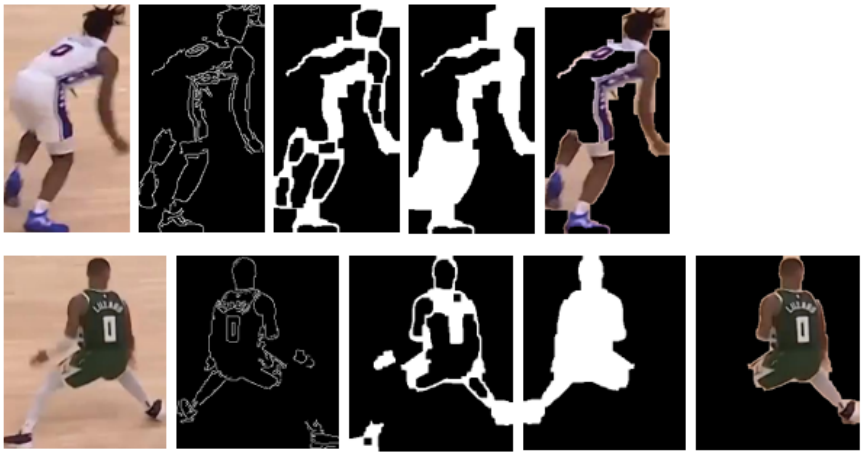

Figure 2: Edge-based algorithm: Case 1 (failure) and Case 2 (successful segmentation).

For each detected player, GrabCut was initialized using the YOLO bounding box (with a small margin) to estimate a foreground mask. Color features were then extracted from the segmented player region in the LAB color space, using the mean of the a and b channels to emphasize jersey color while reducing illumination effects. Finally, players were clustered into two teams using K-means. This approach showed promising results when applied to ground-truth YOLO-annotated bounding boxes, where player localization was accurate and stable. However, when applied to YOLO-predicted bounding boxes, segmentation quality degraded significantly. Inaccurate or noisy detections caused GrabCut to include large background regions or miss parts of the player, leading to unreliable color features and inconsistent clustering. Due to its strong sensitivity to bounding box quality and its lack of robustness under real detection noise, this approach was not retained in the final pipeline.

3.3.3 SigLIP embedding–based team clustering (final approach)

For team detection, we ultimately used an appearance-based clustering pipeline built on SigLIP visual embeddings, which is more robust than edge- or mask-based methods under noisy detections. We first collect a large set of player crops

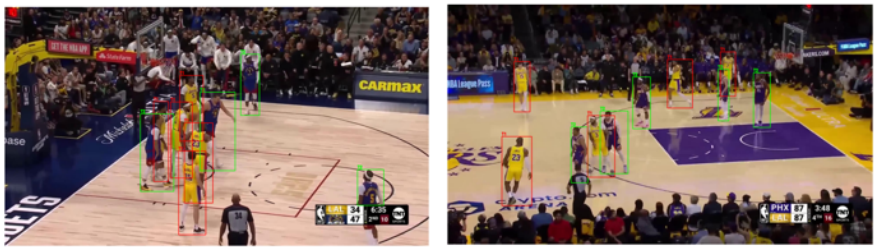

Figure 3: Results of GrabCut-based player segmentation using YOLO-annotated bounding boxes.

from the videos by running YOLO player detection at a fixed stride (e.g., every 30 frames) and shrinking each bounding box (factor 0.4) to focus on the jersey region and reduce background influence. Each crop is then passed through a pre-trained SigLIP vision backbone to produce a high-dimensional feature vector that captures visual appearance beyond raw color. To make clustering easier and reduce noise, we learn a low-dimensional representation using UMAP (e.g., 3D), then fit K-Means with K=2 on these projected features to obtain two team clusters. At inference time, for each frame we extract the shrunk player crops, compute their SigLIP embeddings, project them with the fitted UMAP model, and assign a team label (0 or 1) using the trained K-Means classifier. This approach provided more stable team separation than our earlier segmentation-based attempts, because it relies on strong semantic visual features and does not require accurate pixel-level masks.

3.4 Court segmentation

3.4.1 Court segmentation using color-based region growing

To segment the basketball court in each frame, we implemented a color-based region-growing approach in the Lab color space. The objective was to extract the court area while excluding players and other non-court elements. Players were first detected using YOLO and converted into an ignore mask, ensuring that player pixels were not considered during court segmentation. The input frame was

Figure 4: Team classification using SigLIP, UMAP, and K-Means.

then converted from BGR to Lab color space, and the chroma channels (a and b) were smoothed with a Gaussian filter to reduce noise and small color variations. A bottom-central region of the image, which is likely to contain visible court pixels, was used to estimate the typical court color. From this region, a small set of seed pixels with chroma values closest to the estimated court color was selected. Starting from these seeds, the court region was grown by iteratively adding neighboring pixels whose chroma values were similar to the current estimated court color, while ignoring player regions and strong image edges. The mean court color was updated during the process to adapt to gradual color changes across the court surface. Although this method produced reasonable court masks in many cases, it was not fully reliable. Variations in lighting, shadows, court markings, camera perspective, and partial occlusions sometimes caused the region to leak into non-court areas or fail to cover the entire court. Due to these limitations, this approach was not sufficient on its own, which motivated the exploration of additional and more robust methods for court detection.

Figure 5: Court mask with color-based region growing.

3.4.2 Court segmentation using scanline analysis and dominant color estimation

As an alternative to region growing, we explored a scanline-based approach for court segmentation that relies on color smoothness and dominant color estimation in Lab chroma space. The method processes the image row by row and groups consecutive pixels into segments as long as their chroma values change smoothly. Player regions, obtained from YOLO detections, act as barriers that stop segment growth, preventing the algorithm from crossing through players. Only long and internally smooth segments are kept, under the assumption that the court surface appears as large, uniform regions along horizontal scanlines. From these candidate court pixels, a two-dimensional histogram over the Lab chroma channels is constructed. The most populated histogram bin is assumed to correspond to the dominant court color. This dominant color estimate is then applied globally by selecting all pixels in the frame whose chroma values fall into the same bin, producing a full-frame court mask. While this approach is simple and computationally efficient, it is sensitive to lighting variations, shadows, court markings, and reflections, which can break chroma continuity or introduce competing dominant colors. As a result, the segmentation quality was inconsistent across different

scenes, motivating the exploration of additional court detection strategies.

Figure 6: From left to right: the original court image, the mask obtained from row-based segmentation, the dominant chroma-bin mask restricted to the court region, and the dominant chroma-bin mask applied to the full image.

3.4.3 Court mask refinement using connected components and convex hull (exploratory)

After obtaining a global court mask based on dominant chroma, we applied an additional refinement step to enforce spatial consistency. The binary mask was first decomposed into connected components, and only the largest component that extended sufficiently toward the bottom of the image was retained, under the assumption that the visible court occupies the lower part of the frame. This step helped remove isolated false positives located in the stands or background. From this largest connected component, a convex hull was computed to obtain a single, compact approximation of the court area. The resulting hull provided a simplified geometric representation of the court surface, which was visually more coherent than the raw pixel-level mask. However, this refinement was still not sufficiently accurate. The convex hull often overfilled regions outside the true court boundaries or failed to respect perspective distortions. We nonetheless used this representation as a basis to experiment with classical geometric methods such as Hough line detection, KLT tracking, and corner detection to extract court

keypoints. In practice, these methods did not yield reliable or stable results due to noise in the mask and variability in camera viewpoints. As a result, this approach was ultimately not retained and motivated the move toward the final method described in the following section.

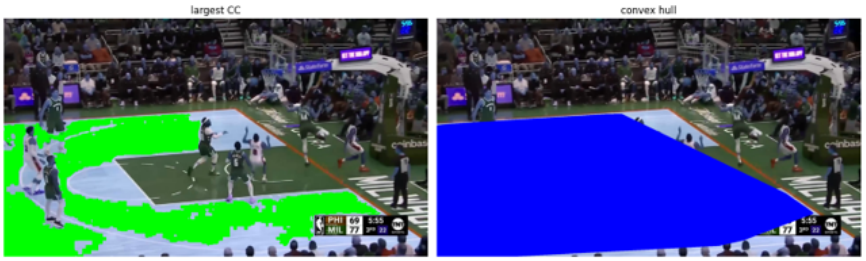

Figure 7: The left image show the connected components mask, and the right image the convex hull

3.5 Court Homography Estimation and Player Mapping

As automatic court geometry estimation methods based on segmentation, line detection, and keypoint extraction proved unreliable under broadcast conditions, we adopted a simplified and robust homography-based approach to enable player-to-court mapping.

A small set of corresponding points was manually selected between a representative video frame and a static top-down basketball court template. The court template image used for this mapping was obtained from an open-source basketball analysis repository made publicly available by Tarek [7]. These points correspond to visually distinctive and geometrically stable locations on the court, such as line intersections and corners. Using these correspondences, a planar homography was estimated using a RANSAC-based method to reduce sensitivity to point selection noise.

Once the homography was computed, each tracked player was projected onto the court template using an estimated ground contact point. Specifically, the bottom-

center point of each player's bounding box was used as a proxy for the player's foot position, which provides a reasonable approximation of the physical location on the court. These ground points were transformed via the homography and rendered on the court template, producing a top-down visualization of player positions.

Although this approach requires manual initialization, it proved significantly more stable than fully automatic court detection methods in our experiments. It enabled consistent spatial mapping of players across frames and allowed downstream visual analytics, such as team occupancy and movement visualization, despite the challenging conditions of broadcast basketball footage.

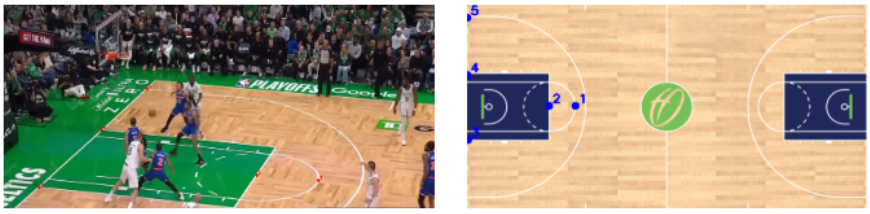

Figure 8: Manual point correspondences between the broadcast frame and the canonical court template used to estimate the homography.

3.6 Ball Possession Estimation and Temporal Smoothing

In addition to object tracking, we implemented a lightweight ball possession estimation module to infer which team controls the ball at each time step. This module operates purely on spatial and temporal information derived from detections and tracks, without requiring additional learning.

For each frame in which the ball is detected, its bounding box is compared against all tracked player bounding boxes. Possession is assigned to the player whose bounding box overlaps the largest fraction (>0.6) of the ball area. The corresponding team label of that player is then used to infer team possession.

To mitigate rapid fluctuations caused by detection noise, brief occlusions, or ambiguous ball-player interactions, a temporal cooldown mechanism is applied. A change in possession is only accepted if it persists consistently for a minimum

number of frames. If the ball is not detected or no player sufficiently overlaps with it, possession is set to undefined.

The estimated possession state is visualized directly on the video frames using color-coded labels corresponding to each team. This simple temporal logic significantly improves visual stability and provides an intuitive representation of ball control dynamics during gameplay.

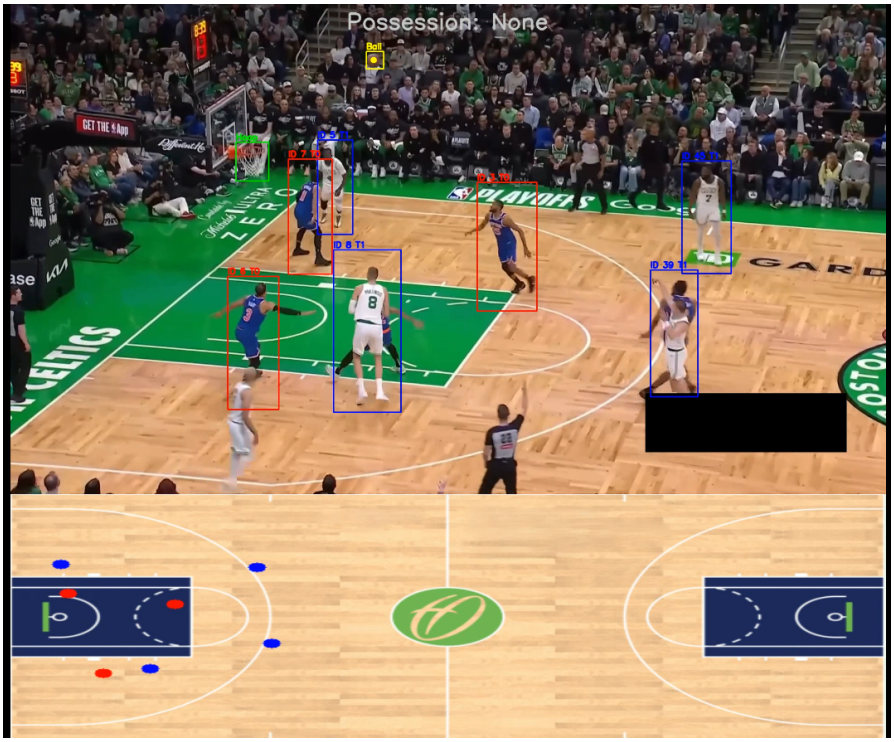

Figure 9: Visualization of final model output with temporal smoothing applied.

4 Limitations and Future Work

Despite the progress made in this project, several limitations remain that present opportunities for future improvement.

4.1 Dataset Scale and Generalization

The current models were trained on relatively small, specialized datasets from Roboflow Universe. While these datasets provided sufficient annotations for initial development, training on larger and more diverse basketball datasets would significantly improve model generalization across different broadcast environments, lighting conditions, camera angles, and playing venues. The limited training data likely contributed to detection inconsistencies observed in footage with unusual perspectives or challenging visual conditions.

4.2 Video Quality and Resolution Robustness

The pipeline currently lacks robustness to videos of varying quality, resolution, and aspect ratios. Broadcast basketball footage can range from high-definition professional broadcasts to lower-quality recordings with different frame sizes and compression artifacts. Future work should focus on developing preprocessing strategies or adaptive model architectures that can handle this variability without requiring separate fine-tuning for each video format.

4.3 Advanced Segmentation Models

The color-based and scanline court segmentation methods explored in this project showed limited reliability under challenging conditions. Modern foundation models such as SAM2 (Segment Anything Model 2) offer significantly more robust semantic segmentation capabilities and could provide more accurate court masks without requiring hand-crafted heuristics. These models could also improve player segmentation for team detection, reducing the sensitivity to bounding box quality that limited our GrabCut-based approach.

4.4 Ball Depth and 3D Trajectory Estimation

The current ball tracking operates purely in 2D image space with simple interpolation for missed detections. Incorporating depth perception and 3D trajectory estimation would enable more physically accurate ball tracking and better handling of situations where the ball moves toward or away from the camera. This would require either stereo vision, depth estimation from monocular cues, or integration with court geometry for metric reconstruction.

4.5 Improved Court Homography Estimation

Court geometry estimation remains one of the weakest components of the pipeline. Future work should explore using YOLO-based keypoint detection models specifically trained to identify court markings (e.g., three-point lines, free-throw lines, center circle). These detected keypoints could then be matched to a standard court template to robustly estimate the homography transformation, enabling accurate metric measurements and proper perspective correction. This approach would be more reliable than the color-based segmentation methods that proved sensitive to lighting and visual clutter.

Addressing these limitations would transform the current exploratory pipeline into a more robust and deployable system for real-world basketball analytics.

References

[1] A. Alhammadi et al. A comprehensive review of ball detection techniques in sports, 2024. ResearchGate preprint.

[2] A. Author and B. Author. How ai-based computer vision algorithms are impacting the sports industry: A survey. *Journal / arXiv*, 2021. URL `https://arxiv.org/abs/xxxx.xxxxx`.

[3] Glenn Jocher et al. Ultralytics yolov8. `https://github.com/ultralytics/ultralytics`, 2023. Accessed 2025-12-14.

[4] Roboflow. Identify basketball players. `https://blog.roboflow.com/identify-basketball-players/`. Accessed: 2025-12-14.

[5] Lei Sha, Jennifer Hobbs, Panna Felsen, Xinyu Wei, Patrick Lucey, and Suman Ganguly. Court reconstruction for camera calibration in broadcast basketball videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[6] Rajiv Shah and Rob Romijnders. Applying deep learning to basketball trajectories, 2016. URL `https://arxiv.org/abs/1608.03793`.

[7] Abdullah Tarek. Basketball analysis. `https://github.com/abdullahtarek/basketball_analysis`, 2023. Accessed: March 2025.

[8] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, 2017.