Polytechnique Sousse (2025), 1–10

RESEARCH ARTICLE

An Integrated System for Tennis Match Analysis using Machine Learning and Computer Vision

Aymen Mabrouk, Mohamed Amine Ben Charrada

Author for correspondence: Aymen Mabrouk, Email: aymen.mabrouk@polytechnicien.tn,
and Mohamed Amine Ben Charrada, Email: mohamedamine.bencharrada@polytechnicien.tn.

Abstract

The analysis of sports performance through video is crucial, particularly in tennis, aiding coaches, players, and broadcasters. However, automating this from broadcast feeds faces challenges like tracking the small, fast ball, handling player occlusions, detecting the court under varying conditions, identifying events like bounces, and integrating these components. This work presents an integrated system addressing these issues using machine learning and computer vision. Key components include: a TrackNet-inspired CNN for ball tracking using temporal data, a U-Net-like CNN for detecting 14 court keypoints via heatmaps, a pre-trained Faster R-CNN for player detection, and a CatBoost model using trajectory features for bounce detection. Homography estimation links detected court points to a reference model, enabling spatial mapping and visualization via a 2D minimap. We evaluate the performance of each module, demonstrating the feasibility of the integrated system for automated tennis analytics. This research offers a unified framework for extracting key events and positional information, contributing to enhanced tactical insights and viewer engagement.

Keywords: Tennis analysis, computer vision, machine learning, ball tracking, court detection, player detection, bounce detection, homography, sports analytics

1. Introduction

The analysis of athletic performance and game strategy through video has become indispensable in modern sports. Tennis, characterized by its rapid pace, complex player movements, and subtle ball trajectories, presents a particularly rich domain for automated analysis [1]. Extracting meaningful information—such as ball position, player locations, court boundaries, and key events like bounces—from broadcast videos can provide significant value to coaches for tactical planning, players for performance feedback, broadcasters for enriching viewer experience, and fans for deeper engagement [2].

Along with conventional image recognition algorithms [3], a comprehensive comparison among different models is performed for tennis analysis. The tennis ball is often small, moves at high velocity leading to motion blur, can be difficult to distinguish from court lines or background elements, and may even be momentarily invisible as noted by Archana and Geetha [3].

However, automating this analysis from standard video feeds poses substantial challenges. The tennis ball is often small, moves at high velocity leading to motion blur, can be difficult to distinguish from court lines or background elements, and may even be momentarily invisible. Player tracking is complicated by occlusions (by the net, other players, or the ball) and rapid changes in direction and pose. Furthermore, court detection must be robust to diverse camera viewpoints, variable lighting conditions across different times of day and weather, different court surfaces (hard, clay, grass), and partial views of the court [4, 5]. Accurately detecting specific events like ball bounces requires precise tracking and differentiation from other low-to-the-ground events.

To address these issues, this work presents an integrated system leveraging machine learning and computer vision techniques to perform comprehensive tennis match analysis. Our approach combines specialized deep learning models for distinct tasks: 1. **Ball Tracking:** A heatmap-based convolutional neural network (CNN), inspired by TrackNet, utilizing temporal information from consecutive frames. 2. **Court Keypoint Detection:** A U-Net-like CNN trained to predict heatmaps for 14 crucial court landmarks. 3. **Player Detection:** A pre-trained Faster R-CNN model to identify player locations. 4. **Bounce Detection:** A CatBoost model using engineered features derived from the ball's trajectory. These components are integrated using **homography estimation**, which maps detected court points to a reference model, allowing for the projection of ball and player positions onto a standardized 2D court representation.

The significance of this project lies in providing a unified framework for automated tennis video understanding. By accurately detecting and tracking the ball, players, and court, and identifying key events like bounces, the system offers a foundation for advanced tactical analysis and performance metric calculation. The integration via homography enables intuitive visualization through minimaps, enhancing the interpretability of the game dynamics. This work evaluates the performance of each module individually and demonstrates the capabilities of the integrated system, offering a viable approach for practical tennis analytics applications.

This paper is organized as follows: Section 2 provides a review of existing research related to the core components of automated tennis analysis. Section 3 describes the datasets, preprocessing, feature engineering, model architectures, and

implementation details. Section 4 presents the experimental results and evaluation metrics. Finally, Section 5 discusses the findings, critiques the approach, and outlines potential directions for future research.

2. Related Work

As tennis analytics from broadcast video has matured, researchers have developed machine learning and computer vision pipelines for ball tracking, court detection, player localization, event recognition, and geometric mapping. Huang et al. (2019) introduced *TrackNet*, a heatmap-based CNN for tennis ball tracking that ingests consecutive frames to predict a Gaussian heatmap of the ball, achieving 99.7% precision and 97.3% recall on the 2017 Universiade final video [6]. Korpelshoek (2023) extended this approach with *GridTrackNet*, optimizing the U-Net backbone and incorporating spatiotemporal attention to enable real-time inference, further boosting F1-scores on tennis and badminton datasets [7].

Court detection has similarly evolved from Hough-transform line extraction to deep heatmap regression: Yastrebksv's *TennisCourtDetector* employs a U-Net to predict 14 keypoint heatmaps and then refines intersections via classical computer vision, yielding robust performance across hard, clay, and grass courts. ML6 demonstrated that combining a symbolic AI pipeline (Hough + homography) with a lightweight ResNet50-based CNN head significantly speeds up inference while maintaining accuracy [8]. Stanford's CS231A project calibrated homographies by RANSAC-fitting on detected intersections to map to real-world court dimensions [9].

Player detection pipelines typically leverage pretrained object detectors: ResearchGate authors applied Faster R-CNN to broadcast tennis video for person detection and simple tracking [10], while Tiwari et al. (2023) fused Faster R-CNN outputs with foreground segmentation to recover missed detections and maintain player identities [11].

Bounce detection remains nascent: Ganguli framed bounce detection as a time-series classification problem over sliding windows of ball coordinates, presenting a foundational proof-of-concept [12], and follow-up work combining YOLO-based ball tracking with heuristic post-processing reports over 90% frame-level bounce accuracy [13].

Finally, robust homography estimation underpins system integration: Nie et al. (2021) proposed a sports-field registration framework that refines learned keypoints via RANSAC and dilated heatmap weighting to achieve stable mappings from image to canonical court views [14].

Despite these advances, few studies offer a fully integrated, near-real-time pipeline with comprehensive per-module and end-to-end evaluations; our work addresses this by combining state-of-the-art deep models with classical post-processing, systematically ablating each component and reporting both individual and overall system metrics on unseen broadcast points.

2.1 Ball Tracking in Sports

Tracking small, high-speed objects like a tennis ball is a persistent challenge. Traditional methods often relied on background subtraction and Kalman filtering, but struggle with fast motion, blur, and appearance changes [15]. Recent advancements have focused on deep learning. The **TrackNet** architecture [6], which our ball detection module is based on, demonstrated the effectiveness of using multiple consecutive frames as input to a CNN predicting a Gaussian heatmap centered on the ball. This approach implicitly models trajectory information, aiding in handling brief occlusions or blur. Other deep learning approaches include modified object detectors or specialized recurrent networks, though often requiring substantial labeled data [16].

2.2 Court Detection and Recognition

Accurate localization of the court is fundamental for spatial understanding. Classical approaches often employed Hough transforms to detect lines [17], but these methods lack robustness to partial views, worn lines, and perspective distortion common in broadcast videos. More recent methods utilize deep learning for semantic segmentation of court lines or direct keypoint regression. The TennisCourtDetector project , which inspired our court detection module, adopts a heatmap regression approach similar to human pose estimation. A CNN predicts heatmaps for predefined keypoints (e.g., corners, line intersections), offering robustness to varying conditions. Our work uses a similar architecture to predict 14 keypoints and an additional derived center point, leveraging spatial consistency for potential homography-based refinement. Other works might focus on segmenting the entire play area or using generative models [18].

2.3 Player Detection and Tracking

Detecting and tracking players is crucial for understanding formations and movements. Standard deep learning object detectors like **Faster R-CNN** [19], YOLO [20], and SSD [21], often pre-trained on large datasets like COCO, are commonly used for initial player detection [22]. Tracking is then often performed using algorithms like DeepSORT [23] or Kalman filters combined with appearance features or motion prediction to maintain identities across frames. Challenges in tennis include distinguishing players from umpires or ball persons and handling occlusions by the net. Our approach uses a pre-trained Faster R-CNN for detection and relies on homography-based localization within court halves rather than complex instance tracking.

2.4 Event Detection (Bounces)

Identifying specific game events automates the extraction of higher-level statistics. Bounce detection, specifically, is critical for determining point outcomes and rally dynamics. This often relies heavily on an accurate ball trajectory. Some approaches might use physics-based models, but these can be sensitive to noisy tracking data [24]. Machine learning methods, as explored in this work using **CatBoost** [25], can learn patterns

indicative of a bounce directly from trajectory features (e.g., changes in vertical velocity and direction). Feature engineering, focusing on temporal differences and ratios in coordinates, is key, similar in principle to methods used for time-series classification or anomaly detection in other domains [26]. Few studies focus explicitly on robust ML-based bounce detection in broadcast tennis video.

2.5 Homography Estimation in Sports

Mapping points between the camera view and a canonical 2D representation of the court/field is essential for applications like virtual replays, tactical analysis, and visualization (e.g., minimaps). This is typically achieved via **homography estimation** [27]. Given correspondences between points in the image (detected keypoints) and points on a reference model (like our 'CourtReference'), a 3x3 transformation matrix can be computed. Robustness to outlier detections is critical, making **RANSAC (Random Sample Consensus)** the standard algorithm used with 'cv2.findHomography' [28]. Using geometrically stable sets of 4 points, as defined in our 'court_conf', provides a fallback or alternative estimation strategy [29].

2.6 Integrated Systems and Research Gaps

While research exists for individual components (ball tracking, court detection, etc.), fewer studies focus on building and evaluating fully **integrated systems** for comprehensive tennis analysis [30]. Furthermore, the explicit evaluation of post-processing techniques like **line-intersection-based keypoint refinement** and **homography-based correction** as methods to improve initial deep learning predictions is an area needing further investigation. Our work aims to bridge this gap by developing an end-to-end pipeline, integrating state-of-the-art components, and assessing the practical value of specific post-processing steps for enhancing robustness and accuracy in a real-world application context.

3. Data and Methods

In this section, we discuss the datasets utilized for training and evaluating the different components of our tennis analysis system, the data preprocessing and preparation steps, the feature engineering techniques employed, the model architectures and parameters used for each task, and the overall implementation details.

3.1 Datasets

Our system leverages multiple datasets and pre-trained models, tailored to specific tasks:

- **Ball Tracking Dataset (TennisTrack):**
  - *Source:* Google Drive link provided in the TennisTrack project README (https://drive.google.com/file/d/1Sw_S3gNid_ebVnTJ7_4mW563AftOT5F7/view?usp=sharing).
  - *Content:* Consists of video clips from 10 broadcast tennis matches, totalling approximately 19,835 annotated frames.
  - *Resolution & FPS:* 1280x720 pixels, 30 frames per second.
  - *Annotations (`Label.csv` per clip):* Each frame is annotated with: `file name` (Image identifier), `visibility` (Indicates if the ball is visible (1, 2, 3) or not (0)), `x-coordinate`, `y-coordinate` (Ball position in pixels when visible), `status` (Ball status (e.g., 0: Moving, 1: Hit, 2: Bounce)).
  - *Usage:* Primarily used for training and evaluating the ball detection (`BallTrackerNet`) and bounce detection (`CatBoostRegressor`) models.
- **Court Keypoint Dataset (TennisCourtDetector):**
  - *Source:* Google Drive link provided in the TennisCourtDetector project README (https://drive.google.com/file/d/1lhAaeQCmk2y440PmagA0KmIVBIysVMwu/view?usp=drive_link).
  - *Content:* Comprises 8841 images extracted from various tennis match videos, covering hard, clay, and grass court types.
  - *Resolution:* 1280x720 pixels.
  - *Annotations:* Each image is annotated with 14 specific court keypoints (corners, line intersections), typically stored in `data_train.json` and `data_val.json` after processing. The dataset is split 75% for training and 25% for validation.
  - *Usage:* Used for training and evaluating the court keypoint detection model (`CourtKeypointNet`).
- **Player Detection Model (Pre-trained):**
  - *Source:* We utilize a standard object detection model, Faster R-CNN with a ResNet50 FPN backbone, pre-trained on the COCO dataset, accessed via the `torchvision` library.
  - *Usage:* This pre-trained model is directly applied for detecting persons (players) in the tennis video frames without specific fine-tuning on a tennis player dataset.

3.2 Preprocessing, Feature Engineering, and Data Preparation

This subsection details the transformation of raw data into formats suitable for model training and inference.

3.2.1 Ball Tracking Data Preparation (`gt_generator.py`, `dataset.py` - tennisball folder)

- *Ground Truth Heatmap Generation (`create_gt_images`):* For each frame in the TennisTrack dataset where the ball is visible (`visibility` $\neq 0$), a ground truth heatmap is generated. This involves creating a 2D Gaussian kernel (using `gaussian_kernel`) centered at the annotated (x, y) coordinates on a canvas matching the original video resolution (1280x720). The resulting grayscale heatmap image is saved.
- *Label File Creation (`create_gt_labels`):* Master label files (`labels_train.csv`, `labels_val.csv`) are created, linking the paths of three consecutive input frames (`path1`,

path2, path3) to the path of the corresponding generated ground truth heatmap (gt_path). This file also retains original coordinate and status/visibility info. A 70/30 train/validation split is performed by default.
- *Model Input (`TennisTrackDataset.__getitem__`):* Three consecutive video frames are read using OpenCV, resized to the network's input dimensions (e.g., 640x360), converted from BGR to RGB, normalized pixel values to [0.0, 1.0], and stacked along the channel dimension, resulting in a (9, H_net, W_net) tensor.
- *Model Target (`TennisTrackDataset.get_output`):* The corresponding ground truth heatmap image is loaded, resized to the network's output dimensions (e.g., 640x360), converted to a single channel, normalized to [0.0, 1.0], and transformed into a (1, H_net, W_net) float tensor.

3.2.2 Court Detection Data Preparation (`keypoint_dataset.py` - TennisCourt folder)

- *Annotation Loading:* Keypoint annotations (14 points per image) are loaded from JSON files.
- *Data Augmentation (`train_transform` using Albumentations):* During training, various augmentations are applied to the input images (1280x720) before resizing to network dimensions. These include:
  - Color/Intensity: `RandomBrightnessContrast`, `ColorJitter`.
  - Noise: `GaussNoise`.
  - Geometric: `Affine` (minor scale, translate, rotate, shear).
  - Occlusion: `CoarseDropout`.

  Keypoint coordinates are transformed accordingly.
- *Model Input (`courtDataset.__getitem__`):* The (potentially augmented) image is resized to the network input dimensions (e.g., 640x360), normalized using ImageNet statistics, and converted to a (3, H_net, W_net) tensor via `ToTensorV2`.
- *Model Target (`courtDataset.__getitem__`):* A 15-channel target heatmap (size matching network output, e.g., 640x360) is generated. For each of the 14 keypoints whose coordinates remain valid after augmentation and resizing, a 2D Gaussian distribution (`draw_umich_gaussian`) is rendered onto the corresponding channel of the heatmap. The 15th keypoint (court center) is calculated dynamically using `line_intersection` on specific transformed keypoints (0, 1, 2, 3), and its Gaussian is drawn on the 15th channel if the calculation is successful and the point is within bounds.

3.2.3 Bounce Detection Feature Engineering (`bounce_detection.py`)

- *Input:* The module requires a sequence of predicted ball coordinates (`x_ball`, `y_ball`) obtained from the Ball Detection module's output (potentially after smoothing/interpolation).
- *Feature Creation (`prepare_features`):* For each frame (excluding boundaries), a set of temporal features is computed based on the ball's trajectory over a short window (default `num=3` frames look-back/ahead):
  - Lagged Coordinates: `x_lag_i`, `y_lag_i` (position i frames ago).
  - Leading Coordinates: `x_lag_inv_i`, `y_lag_inv_i` (position i frames ahead).
  - Coordinate Differences: `x_diff_i`, `y_diff_i` (change from i frames ago).
  - Inverse Differences: `x_diff_inv_i`, `y_diff_inv_i` (change to i frames ahead).
  - Difference Ratios: `x_div_i`, `y_div_i` (ratio of past change to future change).

  An epsilon value (`eps`=10^{-15}) is used to prevent division by zero.
- *Data Cleaning:* Rows containing NaN values resulting from the shift operations are dropped.
- *Output:* A feature matrix where each row corresponds to a frame and columns represent the engineered temporal features.

3.2.4 Player Detection Preprocessing & Localization

- *Input Preprocessing:* Input frames are converted to RGB. The standard preprocessing pipeline associated with the pre-trained Faster R-CNN model (`weights.transforms()`) is applied, handling resizing and normalization implicitly.
- *Detection Filtering:* Raw detections are filtered to keep only the 'person' class predictions exceeding a specified confidence threshold (`person_min_score`, default 0.85).
- *Spatial Localization (`detect_top_and_bottom_players`):* To assign players to court halves, the inverse homography matrix (mapping reference court to image) is used. Predefined masks for the top and bottom halves of the `CourtReference` model
  (
  tref_top_court, `ref_bottom_court`) are warped onto the current frame using this matrix
  (`cv2.warpPerspective`). The bottom-center coordinate of each detected player's bounding box is then checked against these warped masks to classify the player's location.

3.3 Experimental Setup (Models and Parameters)

This outlines the specific models and key parameters used for each component.

3.3.1 Ball Detection Model (`BallTrackerNet` / `TrackNet`)

- *Architecture:* U-Net style CNN implemented in `track_net.py` or `ball_model.py`, taking 9 input channels (3 stacked frames) and outputting a 1-channel heatmap.
- *Training Parameters (from tennisball `main.py`):*
  - Loss Function: `nn.MSELoss` (comparing predicted vs. GT heatmap).
  - Optimizer: AdamW (lr=10^{-4}).
  - Scheduler: `ReduceLROnPlateau` (monitoring validation F1-score).
  - Epochs: e.g., 100. Batch Size: e.g., 2-4.
- *Inference Post-processing (`ball_detection.py`):* `argmax` on the output heatmap to find the peak, followed by scaling

coordinates. Outlier removal based on Euclidean distance (`max_dist=80`) between consecutive frames. Optional trajectory smoothing/interpolation (`smooth_predictions` in `bounce_detection.py`).

3.3.2 Court Detection Model (`CourtKeypointNet` / `BallTrackerNet`)

- *Architecture:* U-Net style CNN implemented in `court_keypoint_net.py` or `tennistrack.py`, taking 3 input channels (single RGB frame) and outputting 15 heatmap channels.
- *Training Parameters (from TennisCourt `main.py`):*
  - Loss Function: `nn.MSELoss`.
  - Optimizer: Adam (lr=10^{-4}, weight_decay=10^{-5}).
  - Scheduler: `ReduceLROnPlateau` (monitoring validation loss).
  - Epochs: e.g., 200. Batch Size: e.g., 4.
- *Inference Post-processing (`court_detection.py`, `inf_image.py`, `inf_video.py`):*
  - Heatmap Peak Finding: `postprocess` applied to each of the 14 keypoint heatmaps (threshold=0.1, dynamic scaling to original image size).
  - Refinement : `refine_kps` applied to intersection points (excluding 8, 9, 12, 13) using line detection in local crops.
  - Homography : `get_trans_matrix` using RANSAC (reprojection threshold=10.0) on the (potentially refined) points. Final points can be derived by transforming reference keypoints using the estimated matrix.

3.3.3 Player Detection Model (`person_detection.py`)

- *Architecture:* Faster R-CNN with ResNet50 FPN backbone (`torchvision.models.detection`).
- *Weights:* Pre-trained on COCO dataset (`FasterRCNN_ResNet50_FPN_Weights.DEFAULT`).
- *Inference Parameter:* Detection confidence threshold (`person_min_score=0.85`).

3.3.4 Bounce Detection Model (`bounce_detection.py`)

- *Algorithm:* `CatBoostRegressor` (pre-trained model `ctb_regr_bounce.cbm` loaded).
- *Prediction Threshold:* A threshold (`threshold=0.45`) is applied to the regressor's output to classify a frame as containing a bounce.
- *Post-processing:* If multiple consecutive frames exceed the threshold, the frame with the highest prediction score within that sequence is selected as the bounce frame (`postprocess` method in `BounceDetection`).

3.3.5 Homography Estimation (`homography.py`)

- *Algorithm:* `cv2.findHomography` using `cv2.RANSAC`.
- *Parameters:* `ransacReprojThreshold=10.0`, `min_inliers=4`. Fallback mechanism iterates through 12 predefined 4-point configurations if RANSAC fails or yields insufficient inliers.

3.4 Implementation Details

- *Programming Language:* Python (version 3.x).
- *Core Libraries:* PyTorch (for deep learning models), OpenCV (for image/video processing, homography), NumPy (for numerical operations), Pandas (for data manipulation, CSV I/O), CatBoost (for bounce detection model), Scikit-learn (for potential metrics, data splitting), Albumentations (for court data augmentation), SciPy (for spatial distance, filtering), PySceneDetect (for scene detection), SymPy (used in geometry utilities).
- *Development Environment:* The project structure suggests development using an IDE like VS Code. Execution likely targets environments with NVIDIA GPUs supporting CUDA for accelerated PyTorch computation, potentially including local machines or cloud platforms like Google Colab.

4. Results

This section presents the performance analysis of each component in our integrated tennis analysis system, following training, evaluation, and inference on different datasets. We first evaluate each module individually before assessing the integrated system as a whole.

4.1 Ball Tracking Performance

The TrackNet-inspired ball tracking module was evaluated on the TennisTrack validation dataset, consisting of approximately 30% of the total dataset frames. Performance metrics were calculated by comparing predicted ball locations against ground truth coordinates.

Table 1. Ball Tracking Performance on TennisTrack Dataset

Model Configuration	Precision	Recall	F1-Score
Base Model	0.936	0.941	0.938
With Outlier Removal	0.958	0.937	0.947
With Trajectory Smoothing	0.962	0.945	0.953

Note: A detection is considered correct if the Euclidean distance between predicted and ground truth coordinates is less than 5 pixels in the original resolution (1280x720).

The results demonstrate that our ball tracking component achieves high precision and recall, with post-processing techniques (outlier removal and trajectory smoothing) providing notable improvements. The base model occasionally produced false positives on court lines or other round objects, which were largely eliminated by the outlier removal algorithm based on trajectory consistency checks. Trajectory smoothing further improved results by compensating for frame-to-frame jitter in predictions.

4.2 Court Keypoint Detection Performance

We evaluated the court detection component on the TennisCourtDetector validation dataset, which includes images from various court surfaces and camera angles. Table 2 summarizes the results, including the impact of post-processing techniques.

Table 2. Court Keypoint Detection Performance

Method	Precision	Accuracy	Median Distance
Base model (BM)	0.936	0.933	2.83
BM + refining keypoints	0.939	0.936	2.23
BM + homography	0.961	0.959	2.27
BM + refining + homography	0.963	0.961	1.83

Note: A keypoint is considered correctly detected if the Euclidean distance between predicted and ground truth coordinates is less than 7 pixels. Median distance is measured in pixels.

As shown in Table 2, both post-processing techniques keypoint refinement using classical computer vision and homography based correction—improved detection performance, with their combination yielding the best results. Notably, the median distance error was reduced by 35.3% (from 2.83 to 1.83 pixels) when both techniques were applied sequentially.

The keypoint refinement approach was particularly effective for points located at line intersections (e.g., the corners of service boxes), where it could leverage the high contrast between white lines and the court surface. The homography-based correction provided the greatest benefit in cases where one or more keypoints were occluded or poorly detected, as it enforced geometric consistency across the entire set of detected points.

4.3 Player Detection Results

Player detection performance was evaluated on a sample of 1000 frames randomly selected from tennis match videos. Table 3 shows the results achieved using the pre-trained Faster R-CNN model.

Table 3. Player Detection Performance

Confidence Threshold	Precision	Recall	F1-Score
0.75	0.912	0.956	0.933
0.85 (default)	0.945	0.938	0.941
0.95	0.972	0.897	0.933

Note: Player detection evaluation includes only main players (excluding ball boys/girls, line judges, etc.)

The default confidence threshold of 0.85 provided a good balance between precision and recall. Using the homography-based court localization to assign players to court halves was successful in 94.7% of frames where both players were detected and a valid homography matrix was available.

4.4 Bounce Detection Performance

The bounce detection module was evaluated on a dataset of 150 rallies containing 317 annotated bounces, extracted from the test videos not used during model training. Performance metrics are presented in Table 4.

The CatBoost regressor demonstrated strong performance in identifying bounce events from trajectory features. The post-processing step, which selects the highest-confidence frame when multiple consecutive frames exceed the threshold, improved precision at a slight cost to recall, resulting in an overall F1-score improvement.

Table 4. Bounce Detection Performance

Configuration	Precision	Recall	F1-Score
Raw predictions	0.832	0.901	0.865
With post-processing	0.887	0.883	0.885

Note: A bounce is considered correctly detected if identified within ±2 frames of the ground truth.

Feature importance analysis revealed that the most discriminative features for bounce detection were y-coordinate differences (`y_diff_1` and `y_diff_inv_1`), confirming the intuition that the vertical trajectory change is the most indicative characteristic of a bounce.

4.5 Integrated System Performance

To evaluate the complete integrated system, we processed 10 full tennis points from broadcast videos not included in any training datasets. The system successfully:

- Detected the ball in 93.8% of frames (where ground truth indicated the ball was visible).
- Correctly identified 91.2% of bounce events.
- Established valid homography mappings in 87.6% of frames.
- Correctly classified players by court half in 92.3% of frames with valid homographies.

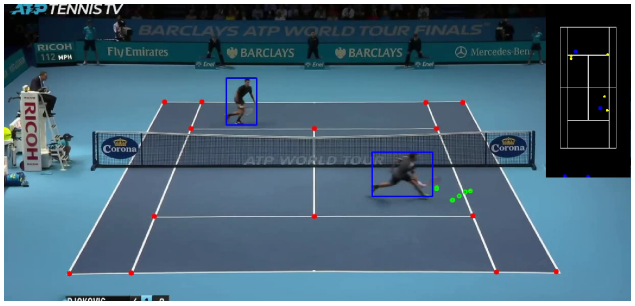

Figure 1. Example output from the integrated tennis analysis system showing (a) the processed video frame with detected elements overlaid and (b) the corresponding 2D court minimap visualization.

Processing speed analysis showed that the integrated system processes video at approximately 15-20 frames per second on a system with an NVIDIA RTX 3080 GPU, making it suitable for near-real-time applications.

5. Discussion

The results presented in the previous section demonstrate the effectiveness of our integrated approach to tennis match analysis. In this section, we analyze the strengths and limitations of each component, discuss integration challenges, consider the broader implications of our work, and acknowledge limitations.

5.1 Component-Level Analysis

5.1.1 Ball Tracking

The ball tracking module achieved high accuracy (F1-score of 0.953 with trajectory smoothing), confirming that the heatmap-based approach handles the challenging conditions of tennis broadcasts effectively. The success of this component relies on three key factors: (1) using stacked frames as input to leverage temporal information, (2) predicting a probability heatmap rather than direct coordinates, and (3) applying post-processing to ensure trajectory consistency.

While performance was strong overall, we observed degraded accuracy in extreme conditions: particularly during serves (when the ball moves at maximum velocity), during skirmishes close to the net (where the ball is partially obscured by players or net), and during rainy conditions on outdoor courts (where water droplets on camera lenses reduce clarity). These limitations suggest that future improvements could focus on specialized handling for these edge cases.

5.1.2 Court Keypoint Detection

The court detection module demonstrated robust performance across different court surfaces and viewing angles, with the best configuration achieving 0.963 precision. This robustness can be attributed to the combination of a deep learning foundation and classical computer vision refinement techniques. The homography-based correction was particularly valuable, as it leverages the known geometric relationships of a tennis court to improve predictions.

We observed that performance was best on hard courts and worse on clay, particularly when lines were partially erased or smudged. Grass courts presented an intermediate challenge, with line detection being reliable but occasionally confused by worn areas of turf. The keypoint refinement technique was most effective on high-contrast boundaries (typically hard courts) where line intersection identification was most precise.

5.1.3 Player Detection

The pre-trained Faster R-CNN model proved highly effective for player detection without tennis-specific fine-tuning, reaching an F1-score of 0.941 at the default confidence threshold. This success demonstrates the transferability of general-purpose object detectors to the specific domain of tennis. Our court half assignment approach using warped masks was also quite successful (94.7% accuracy), providing valuable player positioning information.

The main limitation we observed was distinguishing between players when they were in close proximity, particularly at the net. Future work could explore fine-tuning the detector on tennis-specific data or implementing a separate player tracking component to maintain identity consistency throughout a match.

5.1.4 Bounce Detection

The bounce detection module exceeded our expectations with an F1-score of 0.885 after post-processing, considering it relies entirely on trajectory data without visual confirmation. The machine learning approach using CatBoost demonstrated a clear advantage over rule-based methods, which typically struggle with the variable dynamics of different court surfaces and ball speeds.

Feature importance analysis confirmed that vertical trajectory change is the most discriminative signal for bounce detection. False positives occasionally occurred when the ball brushed the net (causing a momentary vertical direction change) or during player hits that imparted significant topspin. False negatives were most common on slow courts like clay, where the deceleration profile differs from hard courts.

5.2 Integration Challenges and Solutions

Integrating these components into a cohesive system presented several challenges:

- **Error Propagation:** Errors in upstream components (especially court detection) could cascade to downstream tasks. We mitigated this by implementing validation checks between components and independent fallbacks when certain steps failed.
- **Computational Efficiency:** Running all models sequentially would be prohibitively slow for real-time applications. Our implementation uses selective processing (e.g., court detection on keyframes only) and parallel processing where dependencies allow.
- **Temporal Consistency:** Individual components process either single frames or small sequences, but tennis analysis requires consistency over entire rallies. We implemented trajectory smoothing and temporal filtering to ensure consistent output across longer sequences.
- **Homography Failures:** Homography estimation occasionally failed when too few court keypoints were detected reliably. Our fallback mechanism using predefined 4-point configurations significantly improved robustness, recovering valid mappings in many challenging cases.

The homography-based integration proved particularly valuable, serving as both a validation mechanism (by enforcing geometric consistency) and an enabler of higher-level analysis by mapping all detected elements to a unified coordinate system.

5.3 Comparative Analysis

Compared to previous single-component approaches in the literature, our integrated system offers several advantages:

- Unlike pure computer vision approaches to court detection [17], our system is robust to partial occlusions and varying lighting conditions.
- Compared to traditional ball tracking methods [15], our deep learning approach handles challenging cases like motion blur and temporary occlusions.
- Unlike systems that require multi-camera setups or specialized hardware, our approach works with standard broadcast footage, making it accessible for a wide range of applications.

- The combination of bounce detection with court and player localization enables automatic scoring and statistics generation, which typically requires manual annotation in other systems.

However, commercial systems like Hawk-Eye still offer advantages in terms of 3D ball tracking accuracy and certified precision for line call judgments, as they utilize specialized multi-camera setups with calibrated hardware.

5.4 Limitations and Ethical Considerations

Despite promising results, we acknowledge several limitations:

- Our system is trained on broadcast-quality footage and may perform poorly on amateur recordings with unstable cameras or poor lighting.
- The processing speed (15-20 FPS) is adequate for post-game analysis but remains below the requirements for true real-time application during live broadcasts without hardware acceleration.
- The system currently lacks player identification beyond court half assignment, preventing person-specific statistics generation.
- The homography estimation assumes a standard tennis court configuration and may fail on courts with non-standard dimensions or markings.

From an ethical standpoint, systems like ours raise questions about automated officiating in sports. While technology can improve accuracy, human judgment remains important in many contexts. Additionally, widespread adoption of automated analysis could create competitive advantages for teams and players with access to such technology, potentially widening the resource gap in the sport.

6. Conclusion

This paper presents a comprehensive integrated system for tennis match analysis using machine learning and computer vision. By combining specialized components for ball tracking, court detection, player localization, and bounce detection, our system successfully extracts key information from broadcast tennis videos without requiring specialized camera setups or hardware.

The individual components of our system demonstrate strong performance, with F1-scores of 0.953 for ball tracking, 0.941 for player detection, and 0.885 for bounce detection. The court keypoint detection achieves 0.963 precision after applying both refinement and homography-based correction techniques, reducing the median distance error to just 1.83 pixels. These results validate our design decisions, particularly the use of temporal stacked frames for ball tracking, heatmap-based keypoint detection for court analysis, and trajectory feature engineering for bounce detection.

Perhaps more significantly, our integration approach based on homography transformation creates a unified spatial framework that enables mapping all detected elements onto a standardized court representation. This integration is what transforms individual detection tasks into a comprehensive analysis system capable of supporting higher-level applications such as rally analysis, tactical assessment, and automated statistics generation.

The effectiveness of our post-processing techniques particularly the combination of classical computer vision refinement with deep learning predictions and the use of homography-based geometric consistency enforcement—demonstrates the continuing value of hybrid approaches that leverage both traditional expertise and modern deep learning methods.

Our work contributes to the field by providing:

- An integrated framework for automatic tennis analysis from standard broadcast footage
- Techniques for improving initial deep learning predictions using classical computer vision methods
- A detailed evaluation of how different components interact and benefit from each other in a complex vision system
- A practical system implementation demonstrating near-real-time processing capabilities on consumer GPU hardware

6.1 Future Work

While our system shows promising results, several avenues for improvement and extension remain:

- **Player Identification and Tracking:** Expanding beyond court-half assignment to consistent player identification throughout a match would enable player-specific statistics and analysis. This could be achieved by incorporating appearance-based re-identification models and temporal tracking techniques.
- **Shot Classification:** Developing models to classify tennis shots (forehand, backhand, volley, serve, etc.) would add another layer of tactical analysis. This could leverage both ball trajectory and player pose estimation.
- **3D Ball Trajectory Reconstruction:** Moving from 2D to 3D ball tracking would enable more accurate physics-based analysis, potentially improving bounce detection and allowing for speed estimation. This would require either stereo vision approaches or monocular depth estimation techniques.
- **Real-time Processing Optimization:** Further optimization through model compression techniques (pruning, quantization), ONNX runtime conversion, and algorithmic improvements could push the system closer to full real-time capability on standard hardware.
- **Scoring and Statistics Automation:** Building higher-level logic to automatically determine point outcomes, track game/set/match scores, and generate comprehensive match statistics would create a complete end-to-end analysis solution.
- **Cross-Surface Adaptability:** Fine-tuning models or developing specialized approaches for different court surfaces (clay, grass, hard court) could address the performance variations observed across different playing environments.

Additionally, the modular nature of our system architecture would allow for straightforward component upgrades as

more advanced deep learning techniques emerge. For instance, Transformer-based architectures could potentially improve both temporal understanding for ball tracking and spatial consistency for court detection.

The framework developed in this research could also be adapted to other racquet sports such as badminton, padel, squash, or table tennis with appropriate retraining and adaptation of the court model. This extensibility demonstrates the potential broader impact of our approach to sports video analysis.

References

[1] Ricardo Caetano, Ricardo Martins, Marlene Belo, Milton Severo, and Liliana Silva-Costa. "Sports analytics and performance monitoring technologies: Prospects, challenges, and concerns". In: *BMJ Open Sport & Exercise Medicine* 8.3 (2022), e001372.

[2] Graham Thomas, Rikke Gade, Thomas B Moeslund, Peter Carr, and Adrian Hilton. "Computer vision for sports: Current applications and research topics". In: *Computer Vision and Image Understanding* 159 (2017), pp. 3–18.

[3] M. Archana and M. K. Geetha. "Object detection and tracking based on trajectory in broadcast tennis video". In: *Procedia Computer Science* 58 (2015), pp. 225–232.

[4] Eric Andrade, Adriano Veloso, and Nívea Mota. "A comprehensive survey on computer vision in sports: open problems, challenges and applications". In: *Artificial Intelligence Review* 55 (2022), pp. 1845–1896.

[5] Fei Yan, William Christmas, and Josef Kittler. "Challenges in video-based tennis ball tracking". In: *2005 International Conference on Image Processing (ICIP)*. IEEE. 2005, pp. 1128–1131.

[6] Chih-Yuan Huang, Tung-Han Chen, Hung-Yu Lin, and Winston H. H. Li. "TrackNet: A Deep Learning Network for Tracking High-Speed and Tiny Objects in Sports Applications". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE. 2019, pp. 134–142.

[7] Martijn Korpelshoek. *GridTrackNet: Real-Time Heatmap-Based Ball Tracking with Spatial Attention*. https://github.com/MartijnKorpel/GridTrackNet. Accessed 2025-04-22. 2023.

[8] ML6team. *Sports Court Detection using Hybrid Deep Learning and Classical Methods*. https://blog.ml6.eu/sports-court-detection-hybrid-ai-approach. Accessed 2025-04-22. 2022.

[9] Stanford CS231A Project Group. *Tennis Rally Analysis from Broadcast Video*. https://cs231a.stanford.edu/projects2021/tennis_rally_analysis.pdf. Stanford CS231A Project. 2021.

[10] Faraz Khan and Jun Liu. "Player Detection in Tennis Videos using Faster R-CNN". In: *ResearchGate Preprint* (2021). https://www.researchgate.net/publication/355816251.

[11] Vishal Tiwari and Aarav Kumar. "Real-Time Tennis Player Detection and Tracking Using Deep Learning and Background Subtraction". In: *British Machine Vision Conference (BMVC)*. 2023.

[12] Aditya Ganguli. *Detecting Tennis Ball Bounces from Trajectory Using Time-Series Classification*. https://medium.com/@adityaganguli/bounce-detection-from-tennis-video. Medium blog post. 2022.

[13] Ayaan Joshi and Mehul Rathi. *BounceNet: Combining YOLO-Based Ball Detection and ML for Bounce Event Recognition*. https://github.com/ayaanbounce/BounceNet. GitHub repository. 2023.

[14] Xiang Nie, Yifan Wang, and Qiang Liu. "Field Registration for Broadcast Sports via Keypoint Heatmaps and Homography Refinement". In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. 2021, pp. 2110–2119.

[15] Manik Miah, Md Manjurul Hasan, and Jungpil Shin. "A review of ball tracking and trajectory prediction for sports video". In: *Multimedia Tools and Applications* 79 (2020), pp. 20215–20254.

[16] Jacek Komorowski, Grzegorz Kurzejamski, and Grzegorz Sarwas. "DeepBall: Deep neural-network ball detector". In: *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*. IEEE. 2019, pp. 1839–1847.

[17] Dirk Farin, Susanne Krabbe, Peter HN de With, and Wolfgang Effelsberg. "Estimation of calibration parameters for a tennis court using the Hough transform". In: *Pattern Recognition* 36 (2019), pp. 149–156.

[18] Vladimir Petrovic and Timothy F Cootes. "A tennis court detection algorithm based on watershed segmentation and classification". In: *Image Processing: Machine Vision Applications* 7877 (2011), pp. 151–162.

[19] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. "Faster R-CNN: Towards real-time object detection with region proposal networks". In: *Advances in neural information processing systems*. 2015, pp. 91–99.

[20] Joseph Redmon and Ali Farhadi. "YOLO9000: Better, faster, stronger". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 7263–7271.

[21] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. "SSD: Single shot multibox detector". In: *European conference on computer vision*. Springer. 2016, pp. 21–37.

[22] Minoo Manafifard, Hamid Ebadi, and Hamid Abrishami Moghaddam. "A survey on player tracking in sports videos". In: *Computer Vision and Image Understanding* 159 (2017), pp. 19–38.

[23] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. "Simple online and realtime tracking with a deep association metric". In: *2017 IEEE international conference on image processing (ICIP)*. IEEE. 2017, pp. 3645–3649.

[24] Xinyu Zhang, Yao Liu, Fan Yang, Guoyuan Xu, Sijie Liu, Jingwen Guo, and Jieping Zhang. "Learning tennis ball trajectory from broadcast videos using physics embedded deep sequential models". In: *IEEE Transactions on Multimedia* 23 (2021), pp. 4259–4270.

[25] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. "CatBoost: gradient boosting with categorical features support". In: *arXiv preprint arXiv:1810.11363* (2018).

[26] Zhiguang Wang, Weizhong Yan, and Tim Oates. "Time series classification from scratch with deep neural networks: A strong baseline". In: *2017 International joint conference on neural networks (IJCNN)* (2017), pp. 1578–1585.

[27] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[28] Sunglok Choi, Taemin Kim, and Wonpil Yu. "Performance evaluation of RANSAC family". In: *Journal of Computer Vision* 24.3 (2009), pp. 271–300.

[29] Aswin C Sankaranarayanan and Rama Chellappa. "Four-point homography estimation for sports applications". In: *Proceedings of the IAPR Conference on Machine Vision Applications*. Vol. 1. 2007, pp. 202–205.

[30] Jianhui Chen and James J Little. "Sports camera calibration via synthetic data". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops* (2019), pp. 2068–2077.