FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

3D Representation from a Racket Sport Replay

Luís Filipe Carvalhais dos Santos de Matos

Mestrado em Engenharia Informática e Computação

Supervisor: Prof. António Augusto de Sousa

Co-Supervisor: Prof. Rui Rodrigues

July 25, 2024

3D Representation from a Racket Sport Replay

Luís Filipe Carvalhais dos Santos de Matos

Mestrado em Engenharia Informática e Computação

Approved in oral examination by the committee:

President: Prof. Alexandre Miguel Barbosa Valle de Carvalho
External Examiner: Prof. Telmo Miguel Oliveira Adão
Supervisor: Prof. António Augusto de Sousa

July 25, 2024

Resumo

As funcionalidades de repetição e de realce a partir de vídeo desportivo são essenciais para as equipas e os atletas, proporcionando-lhes uma vantagem competitiva ao permitir-lhes analisar o seu desempenho. Em desportos de raquete de ritmo elevado, como o ténis e o padel, a capacidade de fornecer uma visão abrangente do jogo é de extrema importância, uma vez que serve não só para identificar áreas a melhorar, mas também para verificar a legalidade das jogadas. Apesar da sua utilidade, estas tecnologias não estão acessíveis aos jogadores amadores, criando uma valiosa oportunidade de mercado.

O foco principal desta investigação é a criação de uma framework acessível de reconstrução 3D acessível e robusta que possa capturar e visualizar com precisão a trajetória da bola, bem como a posição dos jogadores, tendo em conta uma repetição de desportos de raquete. Este estudo irá explorar técnicas de visão por computador e algoritmos de aprendizagem profunda para melhorar a qualidade e a dimensionalidade das repetições desportivas, oferecendo uma experiência mais imersiva para os espectadores e informações valiosas para treinadores e jogadores.

O processo inicia-se com uma calibração cuidadosa da câmara disponível. Em seguida, o seguimento da bola e a estimativa da pose e posição dos jogadores são efectuados em simultâneo para extrair os dados únicos de cada jogada, ultrapassando obstáculos como oclusões e desfocagens. Em seguida, um processo de segmentação da jogada detecta alterações na direção da bola, como toques pelo o jogador ou ressaltos no chão e nas paredes. Esse processo também identifica a entidade responsável pela mudança de direção, além de anotar a instância específica em que ocorre. Depois disso, a etapa de reconstrução 3D estima as trajectórias balísticas entre estes eventos utilizando um modelo físico que tem em conta a resistência do ar. Esta etapa considera informações previamente recolhidas, como a posição da bola na imagem e as posições dos jogadores, aproveitando o erro de reprojecção para refinar a precisão das trajectórias.

Finalmente, como prova de conceito, o culminar destas tarefas resulta na criação de um modelo 3D que integra toda a informação recolhida. Isto dá uma perspetiva do potencial do protótipo como ferramenta de repetição e realce de desportos de raquete, com o objetivo de enriquecer tanto a análise desportiva como o envolvimento dos espetadores.

Palavras-chave: Análise Desportiva, Desportos de Raquete, Rastreio, Estimação de Pose, Reconstrução 3D

Abstract

Sports Video Replay and Highlight functionalities are game-changers for teams and athletes, providing a competitive edge by allowing them to analyze their performance. In fast-paced racket sports like Tennis and Padel, the ability to provide a comprehensive view of the game is of paramount importance since it serves not only to identify areas for improvement but also to ascertain the legality of plays. Despite their usefulness, these technologies are not readily available to amateur players, creating a valuable market opportunity.

The primary focus of this research is the development of an accessible and robust 3D reconstruction framework that can accurately capture and visualize the ball trajectory and the players' position given a Racket Sports Replay. This study explores computer vision techniques and deep learning algorithms to enhance the quality and depth of sports replays, offering a more immersive experience for viewers and valuable insights for coaches and players.

The process is initiated with careful camera calibration for the available single view. Following that, ball tracking and 2D multi-person human pose and position estimation are performed concurrently to extract each play's unique data, addressing hurdles like occlusions and blurs. Then, a play segmentation process detects changes in the ball's direction, such as player hits or ground and walls bounces. This process also identifies the entity responsible for changing the direction, in addition to noting the specific instance when it occurs. After this, the 3D reconstruction step estimates the ballistic trajectories between these events using a physics model that accounts for air resistance. It integrates previously gathered information, such as the ball's image position and the players' positions, leveraging reprojection error to refine the accuracy of the trajectories.

Finally, as a proof of concept, the culmination of these tasks results in the creation of a 3D model that integrates all collected information. This provides insight into the potential of the prototype as a Racket Sports Replay tool aimed at enriching both sports analysis and viewer engagement.

Keywords: Sports Analysis, Racket Sports, Tracking, Pose Estimation, 3D Reconstruction

Acknowledgements

I would like to leave a brief note of appreciation to all of those who made the realization of this dissertation possible. I want to start by thanking Prof. António Augusto de Sousa and Prof. Rui Rodrigues for supervising all the work present in this document and for guiding me throughout it all, instilling in me good academic and scientific practices. Also, I must mention the effort and professionalism of FEUP and its professors in this important academic chapter.

I am grateful to MOG Technologies, especially Hugo Neves, not only for allowing me to work professionally in this field but also for trusting me with this challenge.

I also could not forget to thank my family and friends for all the morale support provided during the toughest parts of this journey.

Luís Carvalhais de Matos

"If you want something you've never had,
you must be willing to do something you've never done."

Thomas Jefferson

Contents

List of Figures

List of Tables

Abbreviations and Symbols

HPE	Human Pose Estimation
AI	Artificial Intelligence
FPS	Frames Per Second
DNN	Deep Neural Network
DLN	Deep Learning Network
SMPL	Skinned Multi-Person Linear
CCMA	Curvature Corrected Moving Average
PCP	Percentage of Correct Parts
PCK	Percentage Correct Keypoints
AUC	Area Under the Curve
PDJ	Percentage of Detected Joints
OKS	Object Keypoint Similarity
IoU	Intersection over Union
AP	Average Precision
mAP	mean Average Precision
AR	Average Recall
MPJPE	Mean Per Joint Position Error
MPJVE	Mean Per Joint Velocity Error
HCT	Hough Circle Transformation
MAE	Mean Absolute Error
MedAE	Median Absolute Error
MRE	Mean Reprojection Error
MedRE	Median Reprojection Error
MISO	Multiple-Input Single-Output
MIMO	Multiple-Input Multiple-output
MOT	Multi-Object Tracking
DLT	DLT

Chapter 1

Introduction

The possibility of rewatching selected parts of a sports game enhances the overall experience for both athletes and fans. This is made possible by sports video replay and highlights tools that allow for, besides reliving key moments, detailed analysis, strategic insights, and even verifying if plays are in conformity with the rules. These benefits result in an added depth and enjoyment of the game. This dissertation's motivation and goals revolve around being able to provide this kind of experience for the games of Tennis and Padel.

In this chapter, an overview of the dissertation's problem is provided. The chapter begins with a contextualization of the project, followed by a description of the motivation and objectives for addressing it. Lastly, an outline of the document's overall structure is presented.

1.1 Context

For racket sports like Tennis, sport analysis tools have been developed since the 1970's [76] and have traditionally relied on expensive cameras and/or sensors. Reducing these equipment costs is useful to expand the reach of such tools, and one way to do it is to use computer vision techniques to overcome eventual setup limitations. But, even though there has been a strong evolution of the computer vision field in the last decades, the application of such advanced techniques for these purposes is still quite limited.

This dissertation will be elaborated at *MOG Technologies, S.A.* with the intent to provide a prototype for their *PADEL4PRO* project. This prototype aims to be the basis of a potential commercial software specialized in providing an automatic highlight system for amateur games of Tennis and Padel. These amateur games are usually played indoors, in court-filled environments, as seen in Figure 1.1. The project's equipment should be minimized to the essential components necessary to meet its objectives, ensuring the solution remains as accessible and cost-effective as possible. So, this dissertation aims to analyze the feasibility of accomplishing such a task in a monocular setup with the minimum equipment possible.

Figure 1.1: Indoors Padel complex[1].

1.2 Motivation and Objectives

Current Sports video replay and highlights tools are mainly reserved for professional players due to their price tag, leaving a key market opportunity to explore. To expand this market gap, and to allow more people to access these tools, deep research and evaluation of state-of-the-art computer vision techniques are needed to make up for possible setup limitations.

With this work, a 3D reconstruction framework for Tennis and Padel is expected as the outcome. This framework aims to serve as a sports video replay and highlight tool that, given a video from one of the targeted sports, retrieves a 3D representation of the play, including the players' positions and the ball's location. The expected result should also be a relatively low-budget alternative to current market solutions that would not only be able to provide a 3D representation of a replay but also assess if plays are in conformity with the rules.

1.3 Document Structure

Apart from the Introduction chapter, this document is composed of five additional chapters. The next two chapters of this document, Background Knowledge and State-of-the-Art provide a solid foundation for the multiple ways to tackle the problem at hand, that is, the 3D reconstruction of Tennis and Padel games from video replays. In them, both state-of-the-art ball trajectory and pose estimation methods are described, with the purpose of evaluating the feasibility of the work's goals on multi-view and, especially, monocular setups. Then, the Methodology and Development chapter defines the problem, the proposed main approach to tackling it, and an in-depth description of each of its steps. The Tests and Results chapter evaluates the efficacy of the proposed solution through comprehensive testing and analysis. It examines the outcomes achieved at each step of the solution, offering insights into both its performance and the challenges encountered. Lastly, the Conclusion chapter provides a comprehensive summary of the completed work. It highlights the achievements and the shortcomings, discusses the significance of the results, and proposes directions for future research and development.

[1] https://www.lowenergydesigns.com/sport-case-studies/we-are-padel-derby

Chapter 2

Background Knowledge

The Background Knowledge and State-of-the-Art chapters of this document provide a solid foundation for the multiple ways to tackle the problem at hand, that is, the 3D reconstruction of Padel and Tennis games from video replays.

In this literature review, both multi-view systems approaches and, especially, monocular approaches are analyzed to assess their feasibility in retrieving the desired outcome.

In this chapter, the fundamentals of 3D reconstruction, path filters, human pose estimation, and targeted racket sports are introduced for a better assessment of the current state-of-the-art methodologies related to the current problem.

2.1 Fundamentals of 3D Scene Reconstruction from Images

In 3D localization problems, one of the most common approaches is the use of multi-camera systems. These systems detect an object in a frame, for a given instant, in parallel for multiple cameras, and then fuse the detections in order to obtain the 3D location of the object by a process called triangulation. The following subsections cover the fundamentals of 3D reconstruction, beginning with an introduction to the pinhole camera model and camera calibration, followed by triangulation and the evaluation metrics typically used for problems of this nature.

2.1.1 Pinhole Camera Model

Triangulation becomes feasible because of the known camera model for each camera within the system. The use of the pinhole camera model is a key assumption in establishing these camera models since it derives relationships between 3D world points and their 2D projections, and, consequently, the relationship between a 2D image point and the 3D world line of possible points it could have been projected from. Such derivations are only possible due to the pinhole camera model being regarded as a fundamental imaging device capable of flawlessly capturing the geometry of perspective projection, see Figure 2.1.

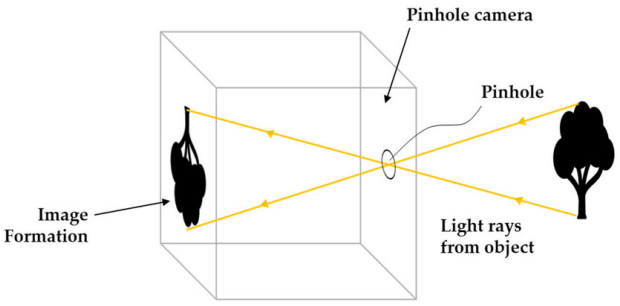


Figure 2.1: A diagram of a pinhole camera [87].

The camera model is defined by its extrinsic and intrinsic camera parameters. The former describe the camera's position and orientation in the 3D world; they may change with respect to the world frame, and are used to transform a point in the object coordinate space into the 3D camera coordinate system. The latter, intrinsic parameters, are the internal characteristics of the camera, including the focal length, principal point, and lens distortion, which are fixed to a particular camera setup and are used to project points in the 3D camera coordinate system into the 2D image coordinates. Both of these conversions can be seen in Figure 2.2.

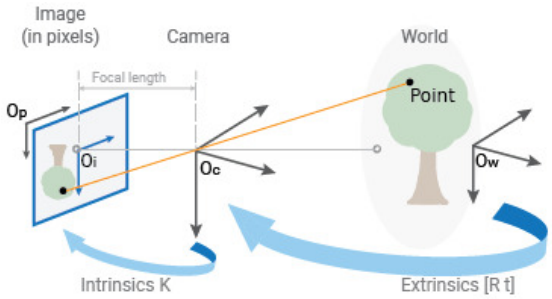


Figure 2.2: Application of extrinsic and intrinsic camera parameters[1].

These parameters can be represented in a 3-by-4 camera matrix that includes both the extrinsic and intrinsic parameters. The 3-by-4 camera matrix is a fundamental concept in computer vision and camera calibration. The camera matrix, often denoted as P, is a combination of the intrinsic matrix (K) and the extrinsic matrix ($[R|t]$), and is denoted as [28]:

$$P = K[R|t] \tag{2.1}$$

Here K is the 3-by-3 intrinsic matrix, which includes parameters such as focal length (f_x, f_y), optical center, also known as principal point, (c_x, c_y), and the skew coefficient. The extrinsic matrix $[R|t]$ is a 3-by-4 matrix composed of the concatenation of R, a 3-by-3 rotation matrix representing the camera's orientation, and t, a 3-by-1 translation vector describing the camera's position in the world coordinates. By knowing these extrinsic and intrinsic camera parameters, it is possible to

[1] https://www.mathworks.com/help/vision/ug/camera-calibration.html

model the mapping from any point in the 3D world coordinates system to the 2D image coordinates through the following equation[1]:

$$w \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = P \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.2}$$

Where (u,v) are the image coordinates of a 3D point in the image plane, (X,Y,Z) are the coordinates of the same point in the world coordinate system, and w is a scaling factor.

2.1.2 Camera Calibration

The process of finding the extrinsic and intrinsic camera parameters is called geometric camera calibration or simply camera calibration. For the task at hand, this process needs to be simple but not necessarily automatic. Also, the speed of the camera calibration process is not a priority if the cameras are static, meaning that this process would only need to be made once. Tough, it would be a priority if pan–tilt–zoom (PTZ) cameras were used because the cameras would change both their intrinsic and extrinsic parameters on the fly.

As denoted in equation 2.1, the 3-by-4 camera matrix P represents both intrinsic and extrinsic parameters, and as P is scaling invariant, it requires eleven free parameters [27]. These parameters can be computed using six points with known positions in both the 3D world coordinate system and the 2D image coordinates. It is worth noting that these six points are not fixed and can be selected on a case-by-case basis, considering that certain points may be occluded in some views. It is also important to note that the choice of the points may impact the usability of the calculated camera matrix P, like if all points chosen are co-planar. In this situation, P is not more than a transform from a 2D plane to another 2D plane, similar to a homography, unable to provide any 3D information outside of the plane. To avoid this issue, [27], uses the 4 corner points of the Tennis court and the 2 posts' top points (Figure 2.3).

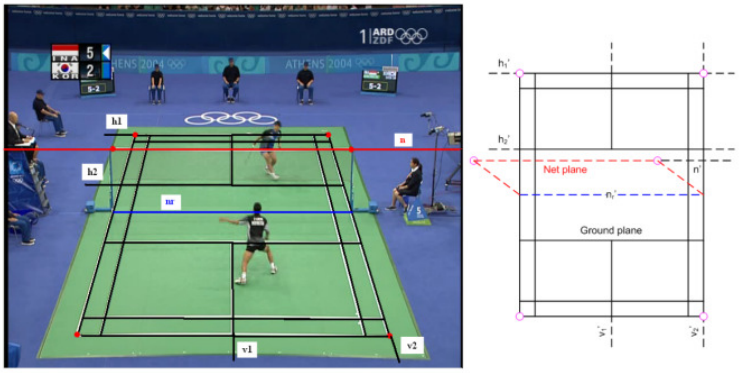
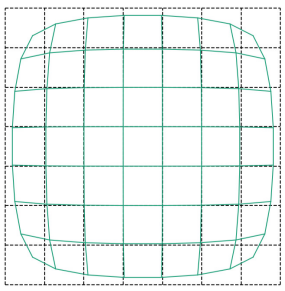


Figure 2.3: The planes, lines, and points selected in the image and the correspondences in the standard model [27, 50].

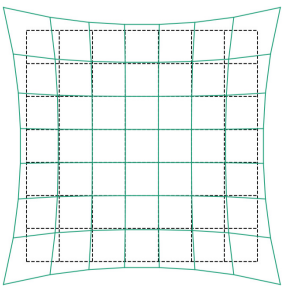
Instead of directly using point features, [27]'s camera calibration algorithm is based on lines. This choice is made because accurately detecting the position of a specific point on a court is more challenging than estimating the position of line segments. Additionally, the detection of lines is more robust, as they are less prone to complete occlusion. Another requirement of [27] is that these lines and points should be selected from two perpendicular planes to avoid a situation like the one mentioned above.

Due to the more detailed specification of the court of Padel, it is expected, as the worst case scenario, a choice/detection of points similar to Tennis.

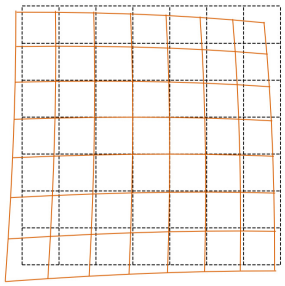
Even if the above steps are performed excellently, the results may be suboptimal due to uncorrected camera lens distortion. The camera matrix alone does not account for lens distortion, as it is based on the idealized pinhole camera model, which assumes a lensless system. To address this issue, lens distortion must be corrected separately, as distortions such as radial and tangential aberrations can significantly impact the accuracy of 3D reconstructions and measurements. For instance, radial distortion, which occurs when light rays bend more near the edges of a lens than at its optical center, causes straight lines to appear curved (see Figures 2.4a and 2.4b), while tangential distortion results from the lens being improperly aligned with the image plane (see Figure 2.4c). To mitigate these issues, careful camera selection and positioning, along with traditional camera calibration techniques such as using calibrated patterns like checkerboards or grids, are typically recommended.



(a) Negative (barrel) radial distortion.



(b) Positive (pincushion) radial distortion.



(c) Tangential (de-centering) distortion.

Figure 2.4: Examples of radial and tangential distortions[2].

2.1.3 Triangulation

With the known camera's intrinsic and extrinsic parameters, triangulation is feasible. Triangulation, as mentioned earlier, is the process of determining a point in the 3D world coordinate system from at least two images from different views.

As shown in Figure 2.5a, a given point y_1 in the 2D image coordinates corresponds to a ray O_1 in the 3D space. Therefore, with only one view of a point, a single 2D image, we can determine that the object lies along a specific 3D ray in the 3D space. However, to precisely determine its

[2] https://www.tangramvision.com/blog/camera-modeling-exploring-distortion-and-distortion-models-

exact location, at least 2 views are needed. So, by having 2 or more 2D projections of the same point in the 3D space, meaning projections from different views, multiple similar rays can be easily reconstructed, with the intersection of these rays corresponding to the previously unknown 3D point's position.

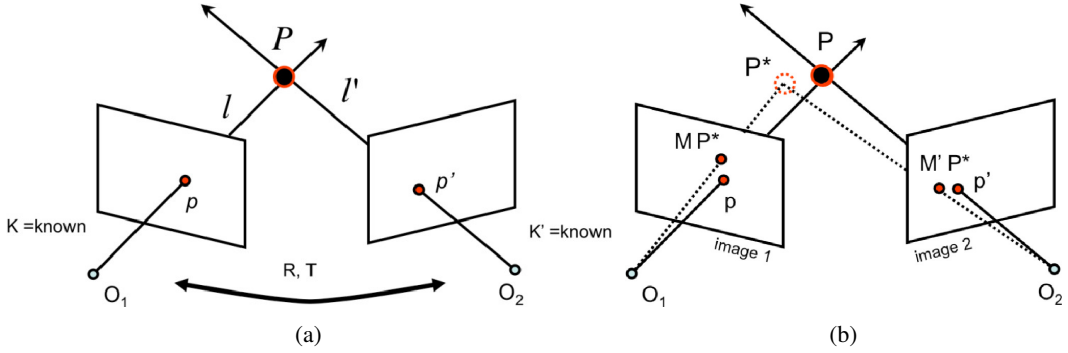


Figure 2.5: Two triangulation cases: a perfect scenario and a more realistic flawed one [29].

Unfortunately, the scenario from Figure 2.5a is not realistic as it is unlikely that 2 rays will intersect. This might be due to noise in the camera calibration process or in detecting the 2D coordinates of a point, as shown in Figure 2.5b. To address this issue, multiple triangulation methods like the intuitive Midpoint Method, the Linear Least-Square (LS) Method, and the Iterative Linear-LS Method have been developed [50]. Their aim is to find the 3D point in space that optimally fits the detected 2D image points, and all can be defined as a triangulation method τ such as [50, 28]:

$$x = \tau(y'_1, y'_2, P_1, P_2) \tag{2.3}$$

Here, y'_1 and y'_2 correspond to the coordinates of the faulty 2D detections, as in Figure 2.5b, P_1 and P_2 to their respective camera matrices and x to the target 3D point.

2.1.4 Evaluation Metrics

The evaluation metrics for 3D scene reconstruction are quite simple due to the approximation of the ball and joints to a single point, meaning that it has no surface. So, the only information available is the location of both the estimated point and the ground truth point.

Mean Absolute Error (MAE) [75], in the context of 3D scene reconstruction, is a metric introduced to measure the accuracy of the reconstructed 3D points or surfaces compared to the ground truth. As the name indicates, this metric is equal to the average of all the absolute errors present, and the lower the value better. Since there are multiple MAE metrics depending on the context, in this specific scenario MAE is also known as **Mean Absolute 3D Error (MA3DE)** [8]. A similar metric called **Median Absolute 3D Error(MdnA3DE)** exists, and instead of the average, it assesses the median error of the reconstructed 3D point. **Mean-Square Error (MSE)**, is also a similar metric to MAE, difference is that in MSE the error is squared.

Precision Plot [82, 81] shows the percentage of frames whose estimated 3D location, i.e. center of the object, is within the defined threshold Euclidean distance to the ground truth.

Reprojection Error [53, 62], sometimes called back-project error, is a metric measured in pixels that represents the distance between the projections of each triangulated point and its detection.

2.2 Path Filters

Tracking systems are susceptible to noise and outliers from sensors, environmental factors, or other sources that might lead to a jittery trajectory. To reduce these unwanted jitters, the system's output needs to be refined and enhanced, and Path Filters are necessary for that effect. These filters perform a smoothing task by diminishing the curvature change of a sequence of noisy 2D/3D points, thus reducing and/or eliminating unwanted elements such as noise, outliers, or irregularities. The main objective is to produce a smoother and more accurate representation of the object's movement over time, but they may also be used or adapted to predict the system's future state.

Path Filters can be categorized [66] based on their output, which can either be a collection of points (discrete) or an analytical function (continuous) and based on their approaches, model-based or model-free. Model-based approaches employ optimization algorithms that excel at finding optimal solutions for well-defined problems, accommodating constraints like kinematic limitations. Nevertheless, their effectiveness hinges on the choice of models and cost functions. Both necessitate careful selection, and even slight tweaks to parameters or the environment can dramatically alter performance or introduce instability. Determining parameters for cost functions is often non-intuitive due to their lack of clear physical significance, compelling a trial-and-error approach until satisfactory outcomes are achieved. These approaches are frequently utilized in domains with clearly understood mechanisms for path generation. On the other hand, model-free approaches are purely data-driven, thus versatile, and often serve as an initial exploration. In this section, both model-based and model-free path filters are described.

2.2.1 Kalman Filter

The Kalman filter [38] is a digital model-based filter that, since the 1970s, has been widely used in various applications such as guidance, navigation, and control of vehicles [88]. It is a recursive algorithm that, via a continuous refinement process, generates estimates of unknown variables from a set of observations, with the underlying assumption that these observations are influenced by Gaussian noise.

This filtering method performs two key steps at each iteration: prediction and update (see Figure 2.6). In the prediction step, the algorithm estimates the next system state based on the previous one, using the state transition model. Then, the update (or correction) step adjusts the state estimate through the weighted average of the predicted state and observed measurements.

[3] https://www.kalmanfilter.net/kalman1d.html

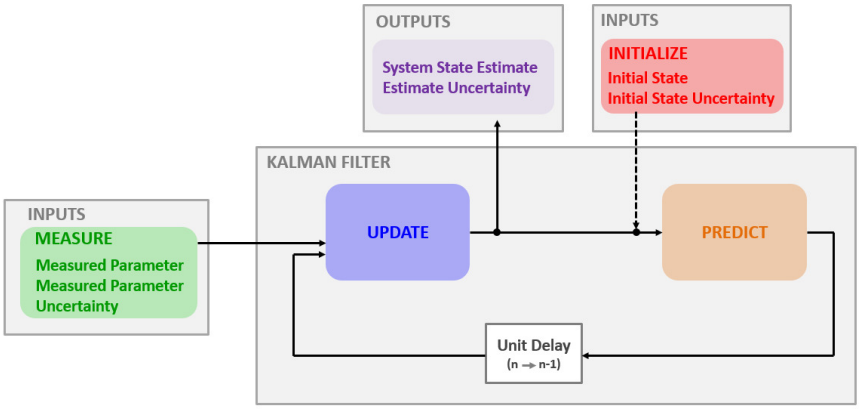


Figure 2.6: General flow of Kalman filter process[3].

The Kalman gain is the weight of these two values and is also updated at each iteration, providing a dynamic mechanism to balance the influence of the prediction and measurement information. In cases of absence of measurements, the system state estimate is simply the predicted state.

The Rauch–Tung–Striebel (RTS) smoother [59], also known as Kalman smoother, is an extension of the Kalman filter that is designed for offline state estimation. By performing a backward pass after a normal Kalman filter algorithm (forward pass), it aims to refine and improve the estimates of the past states of a dynamic system based on the complete set of measurements available.

2.2.2 Curvature Corrected Moving Average (CCMA)

The Curvature Corrected Moving Average (CCMA) [66] is a model-free, general-purpose filtering method for noisy 2D and 3D paths. As the name suggests, CCMA is based on one of the most popular and notable model-free and discrete approaches, the moving average filter [15]. This older filter, although used commonly in digital signal processing, is not suitable for path smoothing since its smoothed points tend to bend inwards in curves. To overcome this phenomenon of inwards bending curves, CCMA extends the moving average approach, aiming to correct the curvature, hence the name. This filtering method can be summarized into four steps, as described in Figure 2.7:

- **Apply moving average** — This step applies the moving average method to ensure that the points are smoothed;
- **Calculate curvature relation** — With the calculated curvature from the smoothed path and the number of points used in its calculation, the original curvature at any point can be estimated;
- **Calculate shift** — Having both the curvatures of the previous steps, the projection length along the orthogonal of the smoothed path is calculated. The shift value is signed, taking into account which one of the sides the curve bends to;

[4] https://medium.com/@steineckertommy/an-accurate-model-free-path-smoothing-algorithm-890fe383d

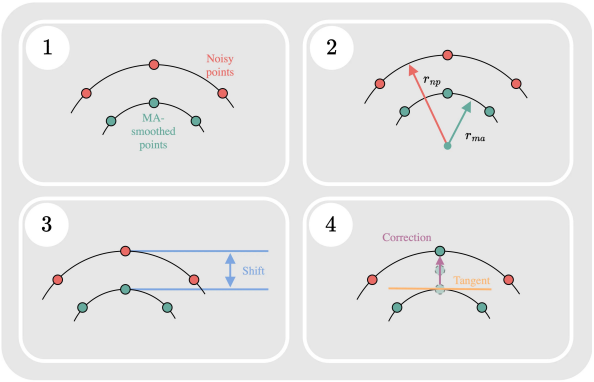


Figure 2.7: Illustration of the four steps of the CCMA[4].

- **Reconstruct** — Given both the smoothed point and the shift calculated obtained on previous steps, an approximation of the original point can be reconstructed by translating the smoothed point along the orthogonal of the smoothed path by the shift calculated.

CCMA's main advantage is its simple approach without optimization techniques, needing only the definition of two parameters W_{ma} and W_{cc}. W_{ma} is a weight that regulates the noise reduction, meaning that while lower values reduce the noise (high variance), larger values will reduce the information of true geometry (large bias). The W_{cc} weight has a similar role, but instead of being applied during the first two CCMA steps, it is applied on the third step, during the calculus of the projection shift length.

2.3 Human Pose Estimation

Human pose estimation (HPE) is a computer vision task that aims to extract the body's configuration from images and videos. Usually, HPE provides a representation of the spatial locations of key body joints that typically include those of the head, shoulders, elbows, wrists, hips, knees, and ankles. HPE finds application in numerous domains [14], including action/activity recognition, action detection, human tracking, human-computer interaction, video surveillance, self-driving, and, also, sports motion analysis. In this section, the fundamentals of HPE, along with the main approaches, are explained.

2.3.1 Top-down and Bottom-up approaches

In a multi-person pose estimation task, human pose estimation methods generally follow a top-down or a bottom-up approach depending on how they start [14].

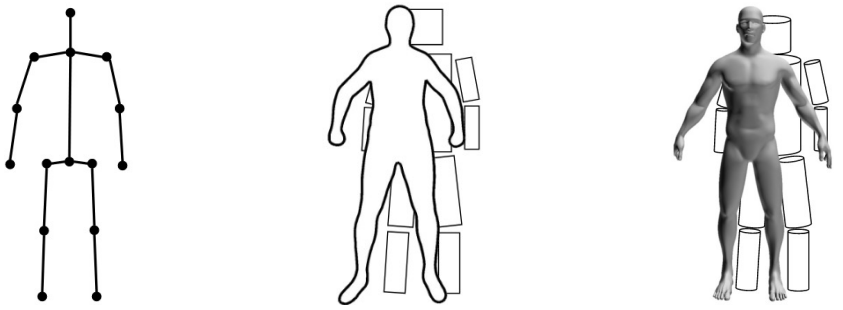
Top-down methods, first identify the location of each person present in the image, in bounding boxes, and then, single-person pose estimation is conducted for each previously identified person.

Bottom-up methods, first detect points of interest, i.e., body parts like joints, limbs, small template patches, etc., of every person present in the input image. Then, these coordinates are grouped by human body model fitting or other algorithms.

When dealing with an elevated number of individuals in an image, the computational cost of top-down methods experiences a notable increase, whereas it remains relatively stable for bottom-up methods. However, in scenarios where individuals may exhibit substantial overlap, bottom-up methods encounter challenges in accurately grouping corresponding body parts. It also may struggle with scale variation throughout the image. Both the considered racket sports may be played in doubles, increasing both the number of players and the possibility of frames in which the players appear overlapped, so, for this, an approach choice isn't clear.

2.3.2 Human Body Models

Human body modeling is crucial for human pose estimation, as the human body is a flexible and intricate non-rigid object with distinct features such as kinematic structure, body shape, surface texture, and the position of body parts or joints. The process involves capturing and representing these characteristics to accurately estimate the human body's pose. In this section, the three main human body models are described.



(a) Skeleton-based model. (b) Contour-based models. (c) Volume-based models.

Figure 2.8: Commonly used human body models [14].

Skeleton-based models [14], also known as kinematic models or stick-figures, depicts a collection of joint locations (usually ranging from 10 to 30) and their corresponding limb orientations based on the human body's skeletal structure (Figure 2.8a). The kinematic model functions as a graph with vertices representing joints and edges encoding constraints or prior connections within the skeleton structure. Despite its advantages in simplicity and flexibility, it has drawbacks, such as a lack of texture information, resulting in the absence of width and contour details of the human body. This simple and flexible human body topology is widely used in both 2D and 3D human pose estimation and human pose datasets.

Planar Models [14], also called contour-based models, contain the rough width and contour information of body limbs and torso (Figure 2.8b). Human body parts are approximately represented with rectangles or boundaries of a person's silhouette. Cardboard models [37] and Active Shape Models (ASMs) [21] are widely used as contour-based models.

Volumetric Models [23], also known as mesh models, approximate the complete surface of the human body (Figure 2.8c). These models provide more detailed information compared to kinematic models, enabling the inference of a subject's spatial representation in a virtual scene or the rendering of captured poses into fully rigged meshes for animation. The most commonly utilized human mesh model for pose estimation is the Skinned Multi-Person Linear model (SMPL) [46].

2.3.3 Pose Estimation Evaluation Metrics

To evaluate the performance of both 2D and 3D pose estimation methods, several metrics have been developed. In this section, the most well-known metrics for this task are described.

2.3.3.1 Threshold metrics

Threshold metrics for human pose estimation are the criteria used to evaluate the accuracy of a pose estimation algorithm by assessing the proximity of predicted poses to the ground truth poses. Common threshold metrics used for pose estimation evaluation include:

The Percentage of Correct Parts (PCP) [14, 25] metric assesses the accuracy of limb definition by examining whether the distance between predicted and true key points is less than a specified threshold (usually 0.5) of a limb's length. While the metric is intuitive in its evaluation, a significant drawback lies in its sensitivity to variations in limb lengths across individuals. This results in shorter limbs being disproportionately penalized compared to longer ones, introducing an element of unreliability to the metric and leading to potentially inconsistent results when applied across diverse datasets.

Percentage Correct Keypoints (PCK) [14, 85] was introduced to address the issue of disproportionately penalizing shorter limbs in PCP. In contrast to PCP, PCK determines the correctness of a key point based on whether the distance between the predicted and true points falls within a specified threshold. For example, setting a threshold of 0.2 of the head bone link means that the distance between true and predicted points should be less than 0.2 times the person's head bone link. This threshold accounts for individual variations, making it a more reliable method for identifying correct points. It offers flexibility by adjusting to the proportions of each person, ensuring fair evaluation irrespective of differences in limb lengths or body sizes.

Area Under the Curve (AUC) [6] metric evaluates various PCK thresholds across the entire range (e.g., from 0 to 0.5 of head bone link), assessing the model's capability to differentiate individual body joints.

Percentage of Detected Joints (PDJ) [6, 71] is another metric that mitigates the problem introduced by PCP. Although sometimes looked at as a variation of PCK [14], PDJ considers a detected joint as accurately positioned if it lies within a specific fraction of the torso length in comparison to the ground truth joint location.

Average Precision (AP) [14] was proposed to address the detection problem that arises in multi-person human pose estimation when only joint locations are annotated, missing both bounding boxes for human bodies/heads and the number of people in the image, as ground truth. In the

AP measure, if a predicted joint falls within a threshold of the ground-truth joint location, it is considered a true positive. AP and its variants are reported based on the Object Keypoint Similarity (OKS) [6, 43] metric. Akin to the Intersection over Union (IoU) metric in object detection, OKS calculates the distance, normalized by the person's scale, between the predicted and ground truth joint location.

2.3.3.2 Mean Per Joint Position Error (MPJPE)

Mean Per Joint Position Error (MPJPE) [23], also known as mean reconstruction error or 3D error, is computed as the average of Euclidean distances between the estimated coordinates and ground truth coordinates for each joint:

$$MPJPE(x,\hat{x}) = \frac{1}{N}\sum_{i=1}^{N} \|m_{\hat{x}}(i) - m_x(i)\| \tag{2.4}$$

In summary, N is the number of joints being processed, $m_{\hat{x}}(i)$ is the function estimating the coordinates of the ith joint, and $m_x(i)$ is the actual (ground truth) position of the joint. Equation 2.4 is the measurement of MPJPE for a skeleton of one frame. When evaluating a video, sequence of frames, it is generalized to the average of each MPJPE's frame.

2.3.3.3 Mean Per Joint Velocity Error (MPJVE)

Mean Per Joint Velocity Error (MPJVE) [23, 58] is a metric introduced for evaluating pose estimation models on video sequences. Unlike traditional metrics that focus on joint positions, like MPJPE, MPJVE considers the smoothness of predictions over time by analyzing the first derivative of 3D pose sequences. This means it assesses how well the model predicts the velocity or transitions of joint positions between frames. This approach is helpful in comparing models using temporal data, providing insights into their ability to capture smooth and realistic motion in pose sequences.

2.3.3.4 Angular Metrics

Angular metrics[23, 32], like the Mean Per Joint Angle Error (MPJAE), offer an alternative way to assess pose estimation:

$$MPJAE(x,\hat{x}) = \frac{1}{3N}\sum_{i=1}^{3N} |(m_{\hat{x}}(i) - m_x(i) mod \pm 180)| \tag{2.5}$$

Instead of focusing on the absolute position of joints like MPJPE, this approach measures the errors in angles between joint segments. The Equation 2.5 calculates each joint's average absolute angular difference between predicted and ground truth angles. The functions $m_x(i)$ and $m_{\hat{x}}(i)$, instead of referring to the positions, like MPJPE, refer to 3D angles for ground truth and prediction, respectively. These metrics are suitable for analyzing the angles between specific limbs rather than

the entire body, like in rehabilitation or sports motion analysis, where the focus is on evaluating and improving the movement of individual limbs. This metric is not suitable for full-body analysis because interpreting these metrics can be relatively unintuitive since they calculate angles locally, and inaccuracies in one joint might not lead to errors in related joints, even if the overall body alignment is incorrect.

2.3.3.5 Volume & Surface Based Metrics

Since some human pose estimation methods aim to estimate the human with volumetric models, there is a clear need for measurement over surfaces. For these cases, Geodesic distance-based metrics are often used [23]. Geodesic Point Similarity (GPS) [26] and Mean Per-vertex Error [57] are some examples of such metrics since they report the distance and error between predicted and ground truth meshes.

2.4 The Targeted Sports

A short introduction to both Tennis and Padel is done in this section to contextualize the distinctive characteristics that define these racket sports.

2.4.1 Basic Rules

In both Tennis and Padel, a single point is initiated with the server positioned behind the baseline. In Tennis, the server delivers the ball diagonally to the receiver, who must let it bounce once before engaging in volleys and rallies. Points are accumulated in a sequence of 15, 30, and 40, culminating in the game. If both players or teams are tied at 40-40, a deuce is declared, requiring a two-point margin for game victory.

Similarly, in Padel, the server stands outside the service box, executing an underhand serve to the diagonally opposite service box. After the initial bounce, the receiver returns the ball, and points are also tallied in a progression of 15, 30, 40, and game. Unlike Tennis, Padel adopts a more straightforward approach after a 40-40 tie, with the subsequent point securing the game.

Although it is common for Tennis to be played in both singles and doubles, doubles is the official game format for Padel. So, in official Padel tournaments, only doubles games are played, but single games can still be played in casual settings, typically on a smaller court.

2.4.2 Courts

In traditional Tennis, players compete on either clay, grass, or hard courts, each surface offering distinct advantages and challenges. Clay courts are known for slower ball speeds and higher bounces, grass courts for quick pace, and hard courts for a balance between the two. The dimensions of a standard tennis court[5] are regulated by international standards, measuring 23.77 meters

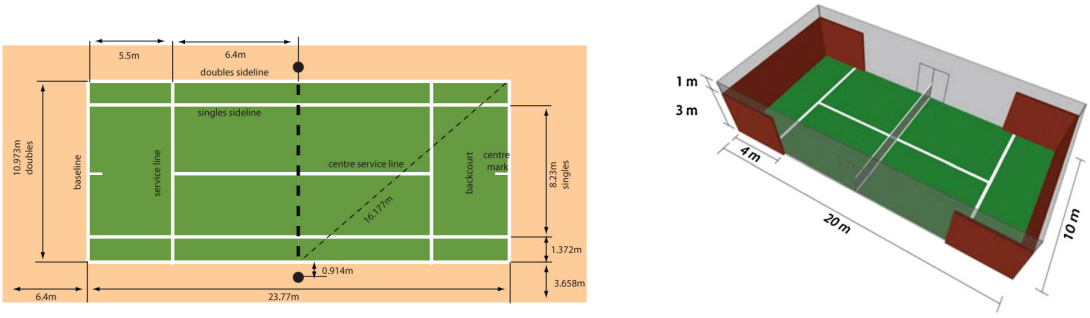


Figure 2.9: Tennis[6]and Padel [10] doubles court dimensions.

long, 8.23 and 10.97 meters wide for singles and doubles, respectively, with a net positioned at a height of 107 centimeters at the posts and 91 centimeters in the center.

Conversely, Padel is played on a smaller court [10], typically measuring 20 meters long and 6 or 10 meters wide, depending on whether it is a singles or doubles game. The net is 10 meters long and 88 centimeters high at the center and can reach a maximum of 92 centimeters at the lateral anchor points. The major difference between these two sports is that the Padel court is entirely enclosed by a 3 to 4-meter-tall surface (transparent or opaque), which gives the ball a regular rebound, and wire netting at the top where the rebound is irregular.

2.4.3 Equipment

While Tennis rackets are similar to the more traditional string rackets, Padel rackets are made of a composite material without strings, and even though they are smaller and perforated, they are still easier to identify on replays than their Tennis counterparts, see Figure 2.10.

Figure 2.10: Example of a Tennis and Padel racket[7].

Both Racket Sports balls are similar[8]. Besides their color, yellow, they share the same weight ranges and have similar diameters, 6.54 to 6.86 centimeters and 6.35 to 6.77 centimeters, for

[5] https://www.harrodsport.com/advice-and-guides/tennis-court-dimensions
[6] https://grand-slam.com.au/tennis-court-dimensions/
[7] https://www.wilson.com
[8] https://worldpadelinsider.com/what-is-the-difference-between-a-padel-ball-vs-tennis-ball

Tennis and Padel, respectively. The only major difference is that Tennis balls tend to bounce higher due to higher internal pressure.

These differences in internal pressure and rackets turn Padel into a lower-paced game than Tennis, making it highly unlikely to reach the same top speeds as Tennis, in which the average speed of a ball hit by a professional is around 120 km/h[9].

[9] https://tennisbolt.com/how-fast-do-pro-tennis-players-hit-the-ball/

Chapter 3

State-of-the-Art

This chapter explores the existing literature that is relevant to the research conducted in this dissertation. It starts with a presentation of the equipment, purpose, and methodology used on current market solutions for Racket Sports Analysis. Then, the chapter follows with a thorough description of the 3D ball localization methods developed in the last few years, as well as a less detailed section for the players' positions and poses estimation methods.

While numerous studies have investigated Computer Vision and Deep Learning applications in the realm of Sports Analysis, as far as the research goes, none of them have integrated both 3D ball tracking and the 3D positions and pose estimation of players. This gap in the existing research has sparked interest in exploring this particular topic.

3.1 Current Market Solutions

Several Sports Analysis tools have already been employed in professional Tennis broadcasting. From the 1980's Cyclops system [76], which used infra-red beams to detect illegal serves to the most recent IBM Slamtracker [76]. Introduced in 2012, this AI tool provides real-time scores and statistics (comprising between 15 and 25 parameters for each game point), such as the number of aces, winners, receiving points won, break points conversions, etc., for each player. This section describes the most popular and mature sports analysis tool, the Hawk-Eye system.

The Hawk-Eye system [4] is the state-of-the-art technology involved in officiating and collecting data in sports like Tennis, Cricket, and, more recently, Football. This system was first used in a professional Tennis Tournament in 2006 with the objective of displaying a 3D reconstruction of a play to clarify controversial calls (Figure 3.1). For that effect, tracking the ball accurately and obtaining its real-world position is necessary.

Since 2013, the Hawk-Eye system has also started to track the player's position, and in 2020, a real-time version, Hawk-Eye Live [7], was first introduced. This technology uses 12 cameras to call shots in real-time located at different locations and angles around the area of play and, when an

[1] https://www.abc.net.au/listen/programs/melbourne-breakfast/tennis-hawkeye-steve-gelagotis/13134764

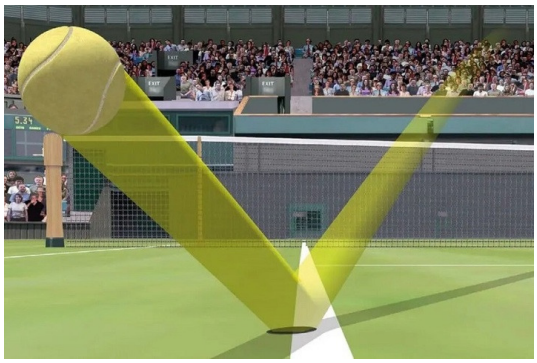


Figure 3.1: Shot Spot - Hawk-Eye's 3D reconstruction of tennis shot[1].

infringement of the rules happens, emits a sound to emulate a human line judge. Information about this novel system is scarce, but from the original ones' patent [63], published in 2001, it is possible to assume that it is based on the principles of triangulation, mentioned in section 2.1. Nonetheless, this system relies on high-end cameras at selected locations and angles with an extensive calibration process. As a result, it becomes too costly for non-professional teams/tournaments.

In 2021, the reliability of this system was so high that both the *Australian Open* and *US Open* started using it to replace all line judges, thus reducing personnel during the COVID-19 pandemic.

3.2 Ball Trajectory

Multiple viewpoint setups are state-of-the-art for 3D ball localization since they tend to offer higher accuracy and higher reliability in case of occlusions. Still, due to the exploitation of multiple synchronized cameras, these systems are evidently more costly and challenging to install compared to single viewpoint setups. In this section, both monocular and multi-camera approaches for 3D ball localization are described to assess better the feasibility of both of these approaches for the task at hand. Still, before that, regular 2D ball detection methods are characterized since they often serve as intermediate steps.

3.2.1 Datasets

2D ball detection and 3D ball localization datasets are essential for the problem at hand since they allow the training and evaluation of the pretended models. Unfortunately, datasets with 3D ball annotations for the targeted sports were not found, so to address that issue, other sports were considered. So, in this section, datasets for Tennis, Badminton, and Basketball are described.

TrackNet [31] introduces two datasets. One is a 2D ball detection dataset for Tennis that consists of 75 minutes of HD video at 30 frames per second (FPS) of segmented Tennis rallies. There are $20,844$ frames in total, each one labeled on its 2D ball position and Visibility Class. This class classifies each frame based on the visibility of the ball, having a different value for frames in which the ball is not within the frame, can be easily identified, and is in the frame but cannot be easily identified.

The other dataset introduced by TrackNet is a Badminton 2D ball detection dataset. It was later improved by both TrackNetV2 [67] and MonoTrack [44], leading to 77 thousand annotated frames from 26 unique professional singles matches. To prevent overfitting, the matches were chosen from different backgrounds and viewing angles. Besides the annotated 2D ball position and Visibility Class, as mentioned before, it also contains timestamp information for each time a player performs a hit on the shuttlecock.

The **APIDIS Basketball** [78, 75, 81] Dataset was acquired by seven un-synchronized cameras, five ground cameras around the court and two fish-eye cameras overhead of the court, at about 22 FPS. This dataset poses challenges for ball tracking due to unfavorable camera locations and difficult lighting conditions. Despite these difficulties, it stands out as one of the limited publicly available datasets for team sports, even if it only provides 2D ground truth.

DeepSport [75] is a basketball dataset that comprises 314 high-resolution panoramic images captured in 15 distinct basketball arenas during professional matches. Detailed 3D annotations for the basketball are provided, specifying the ball center and its vertical projection on the court in the image space.

3.2.2 2D Ball Detection

For this task of 3D ball localization, both possible approaches, monocular and multi-view systems, depend on a good and reliable ball positioning on the image. The hurdles mentioned previously, such as occlusion and blur, which the intended setup is particularly susceptible to, make conventional computer vision techniques produce inadequate results for the task at hand [31]. To address this issue, convolutional neural networks (CNN) 2D ball detectors have been studied, and this section delves into their evolution over the last few years.

3.2.2.1 TrackNet

TrackNet [31] is a heatmap-based deep learning network that aims to precisely retrieve the 2D position of the ball and shuttlecock from amateur broadcast videos of Tennis and Badminton. What set it apart in 2019 was its option to take multiple frames as input to learn, aside from the ball's features and trajectory's characteristics, thus enhancing its capability of object recognition and positioning. This approach addresses challenges associated with low image quality issues, such as blurs, afterimages, and even short-term occlusions.

For that effect, TrackNet is composed of a convolutional neural network (CNN) based on VGG-16 [65], followed by a deconvolutional neural network based on DeconvNet [54]). It takes a number of consecutive 640×360 frames, defined as a network parameter, to generate a probability-like detection heatmap with the exact resolution as the input frames. The initial 13 layers correspond to the design of the first 13 layers of VGG-16 designed for object classification, and the following 14-24 layers are modeled after DeconvNet for semantic segmentation. Upsampling is employed to recover information lost during maximum pooling layers to achieve pixel-wise

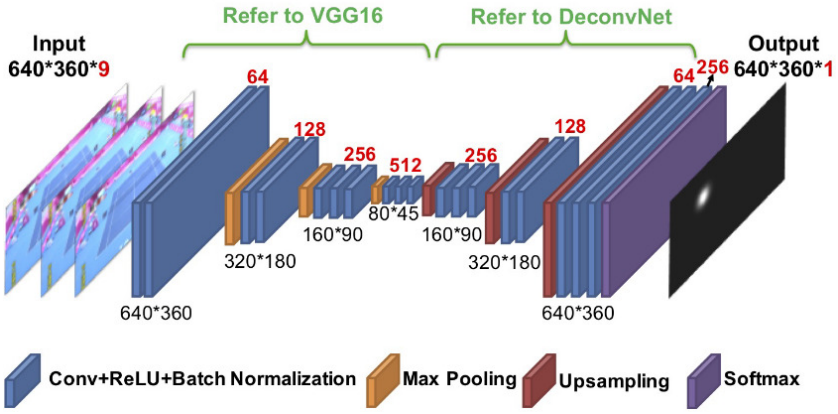


Figure 3.2: The architecture of TrackNet [31].

prediction. Lastly, a softmax layer is activated to transform the previous layer's result, a 256-channel boolean array in the pixel-wise one-hot encoding convention, into the continuous detection heatmap output, ranging from 0 to 255 for each pixel.

To locate the ball candidates in the image, the heatmap is first converted to black and white using a threshold of 128. If a pixel value is greater than or equal to 128, it is set to 255; otherwise, it is set to 0. Then, the Hough Gradient Method[2] is applied to identify a circle on the binary heatmap. If one circle is found, its centroid is returned as the ball's coordinate; otherwise, no ball is considered detected. Thanks to the multiple-frame approach, the object's location can be inferred not only from the current frame but also from previous and future information. This capability allows for predicting the ball's location even in cases of invisibility caused by confusing backgrounds or foreground occlusion.

The evaluations were made with TrackNet's own dataset [31] 3.2.1 and are compared between a conventional image processing technique called Archana's algorithm [49], and three versions of TrackNet: TrackNet Model I, which takes only one input frame; Model II, which takes three consecutive frames as input; and Model II', which differs from Model II only by its enriched training set. To consider that the ball was rightly detected, the Euclidean distance from the model prediction to the ground truth needs to be no more than 5 pixels, the mean diameter of the ball.

Table 3.1: Accuracy metrics of Archanas's models and TrackNet variations [31].

Model	Precision	Recall	F1-measure
Archana's [49]	92.5%	74.5%	82.5%
TrackNet Model I	95.7%	89.6%	92.5%
TrackNet Model II	99.8%	96.6%	98.2%
TrackNet Model II'	99.7%	97.3%	98.5%

[2] https://docs.opencv.org/4.x/da/d53/tutorial_py_houghcircles.html

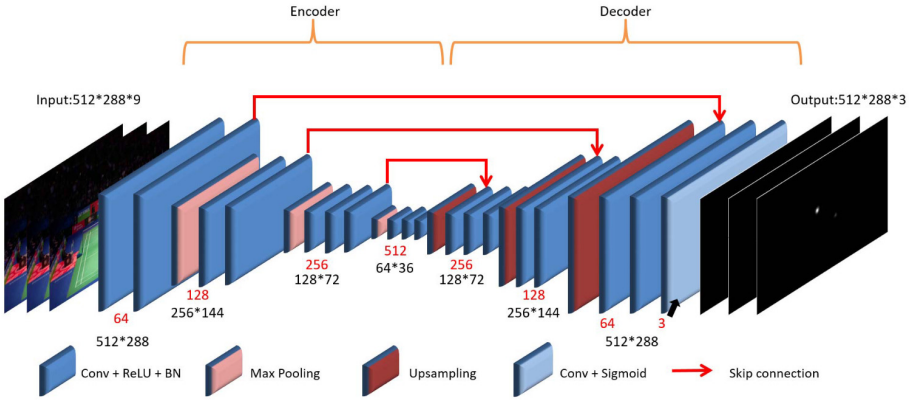


Figure 3.3: The architecture of TrackNetV2 [67].

As it is possible to see in Table 3.1, Model II and II' clearly outperform Model I and Archana's algorithm, making use of its extraction of the trajectory information from neighboring frames. [31] goes deeper in result analysis, stating that taking consecutive input frames reduces the positioning error and also enables it to detect 2 out of 7 occluded test cases correctly.

3.2.2.2 TrackNetV2

One year later, after the introduction of TrackNet [31], the same department proposed TrackNetV2 [67], a deep learning network that aimed to enhance the performance of TrackNet in several aspects, notably in processing speed, prediction accuracy, and GPU memory usage. This proposed deep learning network (DLN), tuned for Badminton, follows a similar encoder-decoder structure as its predecessor, see Figure 3.3.

VGG-16 [65] is still adopted as the encoder to generate the feature map, which in this version is smaller in size due to the reduction of the input image size from 640×360 to 512×288. And, to act as the decoder that generates the prediction heatmaps, with the same size as the input images, an upsampling structure corresponding to the downsampling structure is also adopted. The adjustment in the size of input images aims to enhance processing speed and decrease memory usage while maintaining similar accuracy and precision.

However, three major improvements in architecture were made: the adoption of the skip connection idea of U-Net [61], the renovation of the heatmap representation and loss function, and the re-engineering of the Multiple-Input Single-Output (MISO) design to a Multiple-Input Multiple-Output (MIMO) design.

In the conventional waterfall-like design, the features of small objects might diminish gradually along the lengthy pipeline of convolution layers and max-pooling layers. The skip connections, which transmit feature arrays from the encoder network directly to the corresponding layers in the decoder network, establish shortcuts to retain information about small objects. This feature proves beneficial in tracking the movement of small and swiftly moving shuttlecocks.

The original TrackNet goes through an inefficient process of generating the final prediction heatmap, which, through an *argmax* operation, transforms the 256-channel boolean array in the

Table 3.2: Speed comparisons between TrackNet [31] and TrackNetV2 [67].

Model	Input size	Heatmap	Skip connection	Speed
TrackNet 3-1	640×360			2.64 FPS
TrackNetV2 3-1	640×360	$\surd$		10.42 FPS
TrackNetV2 3-1	512×288	$\surd$		14.15 FPS
TrackNetV2 3-1	512×288	$\surd$	$\surd$	12.88 FPS
TrackNetV2 3-3	512×288	$\surd$	$\surd$	31.84 FPS

pixel-wise on-hot encoding convention into the final prediction heatmap. To avoid this, TrackNetV2 directly generates a 1-channel real-values array in the last sigmoid layer. This alteration not only increases run-time speed but also decreases memory usage. Due to this modification, a new loss function was also altered.

The original TrackNet goes through an inefficient process of generating the final prediction heatmap, which, through an *argmax* operation, transforms the 256-channel boolean array in the pixel-wise on-hot encoding convention into the final prediction heatmap. To avoid this, TrackNetV2 directly generates a 1-channel real-values array in the last sigmoid layer.

Due to the streamlined heatmap representation, generating multiple heatmaps for all input images is feasible, rather than producing only one for each input image. Although the MIMO design may marginally decrease processing speed, as the overhead is primarily in the last layer, the actual throughput of the model is significantly increased. For instance, a 3-in-3-out design almost triples the processing speed compared to a 3-in-1-out design, thereby substantially enhancing performance.

Similarly to [31], [67] considers that detection was done correctly if the Euclidean distance between the predicted coordinate and the ground truth is smaller than 4 pixels. It is possible to assume that this reduction in the threshold is due to the smaller input frame size.

Although [67] does not present any results comparing the detection quality between itself and TrackNet, the values in Table 3.2 clearly demonstrate that a significant boost in speed performance was achieved, from 2.64 to 31.84 FPS in the MIMO, 3-input 3-output, version, enabling its use in real-time applications.

3.2.2.3 MonoTrack

Monotrack [44] is a system developed for use in Badminton. It is an end-to-end system that, without human intervention, can provide shot segmentation and 3D shot trajectory. Its shuttle detection step proposes a U-Net [61] style architecture inspired by TrackNetV2 [67], as shown in Figure 3.4 where $\bigoplus$ and $\bigotimes$ represent additive and concatenation skip connections, respectively.

In line with TrackNetV2, [44] employs a 3-in-3-out architecture to prompt the network to learn temporal context, simultaneously predicting shuttle masks for three consecutive frames (each resized to 288-by-512). Notably, this model has a significantly smaller footprint compared to the

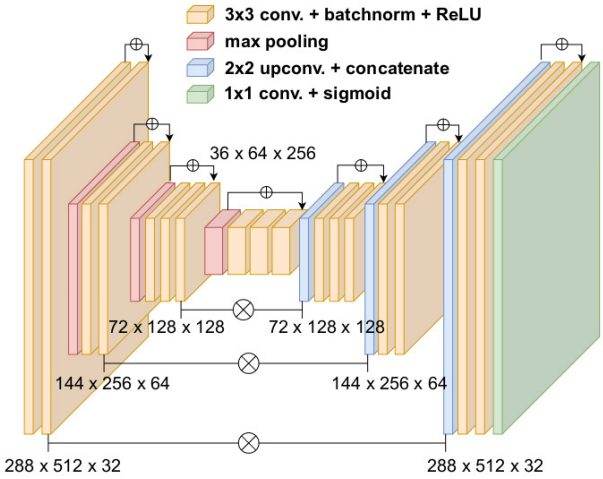


Figure 3.4: The architecture of MonoTrack's shuttle detection model [44].

original TrackNetV2, with 2.9 million parameters versus 11.3 million parameters. This reduction in parameters makes the model faster to train and execute.

Accordingly to their own, [44] also increases accuracy, from 84.0% to 88.6%. This enhancement is credited to two key modifications it implemented. Firstly, it incorporates residual connections to each convolutional layer. Secondly, rather than employing a binary focal loss, it utilizes a weighted combination of the dice loss and the binary cross-entropy to address the input imbalance issue related to the tiny shuttlecock.

3.2.2.4 Widely Applicable Strong Baseline (WASB)

The Widely Applicable Strong Baseline (WASB) [69] argues that the encoder-decoder architecture present in the previous detectors is a drawback for SBDT(Sports Ball Detection and Tracking). WASB aims, as the name implies, to present an SBDT method that performs well for various sports categories despite their differences in ball size, speed, and court size.

WASB incorporates a customized high-resolution feature extraction method inspired by a series of HRNet works [80, 89]. This decision was made to address the requirement for a CNN module capable of generating semantically rich representations without compromising spatial resolution.

The MIMO design present in [67, 44] is also present in this work: N consecutive frames are concatenated along the channel dimension, and the resulting $H \times W \times 3N$ tensor is used as WASB's model input. The model then generates N heatmaps of the same spatial resolution as the input, i.e., $H \times W \times N$.

Initially, WASB's ground truth (GT) was composed of position-aware maps, different from the usual binary GT maps used by [31, 67, 44], and as seen in Figure 3.5a. This decision was made because, according to [69], the resulting prediction tends to be less sensitive to the exact ball position on the existing methods, since the exact ball position is made obscure through the GT map generation process. Later on, [69] found empirically that applying this alteration to all the

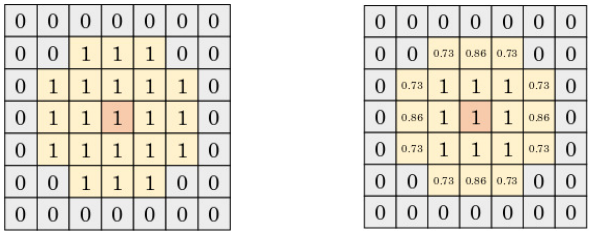


Figure 3.5: An exemplar (a) binary ground-truth (GT) map and (b) real-valued GT map [69].

training data does not statistically improve performance. So, instead, it is only applied to frames where there is a clear difficulty in localizing the object. By managing the loss functions when the background is noisy, focal loss [42] in the earlier epochs with binary GT maps to quality focal loss [40] with real-valued GT maps, the blurry heatmaps get clearer which leads to a more precise localization.

Table 3.3: Benchmark results of SBDT methods on 2 SBDT datasets [69].

		Tennis				Badminton			
Model	# param	F1	Acc.	AP	FPS	F1	Acc.	AP	FPS
TrackNetV2 [67]	11.3*M*	89.4	81.4	80.6	55.3	90.5	85.6	83.6	77.0
MonoTrack [44]	2.9*M*	92.1	85.9	87.3	64.1	90.9	85.9	84.9	75.5
WASB	1.5*M*	94.0	89.0	91.0	58.2	91.6	87.0	88.5	70.4

As it is shown in Table 5.1, for a fixed distance threshold $\tau = 4$[pixel], WASB outperforms both TrackNetV2 and MonoTrack in all metrics but frames per second (FPS), nonetheless is still real-time capable.

3.2.3 Multi-View approach

Multi-camera setups are the go-to solution for 3D reconstruction tasks. Although more expensive and with a longer installation process than their monocular counterpart, this approach naturally tackles several problems like occlusion and 3D object positioning due to its multiple viewpoints and triangulation feasibility.

Generally, the way to tackle this task is to simply triangulate the location of the ball from multiple calibrated 2D ball detection images. Both the 2D ball detection (3.2.2) and 3D ball tracking (2.2) steps are independent of the multiple viewpoints and, so, are not delved in this section. The addressed steps, 2D ball tracking, and 3D ball fusion are usually the main differences between multi-view methods since they make use of the multiple variables and scenarios introduced by this approach.

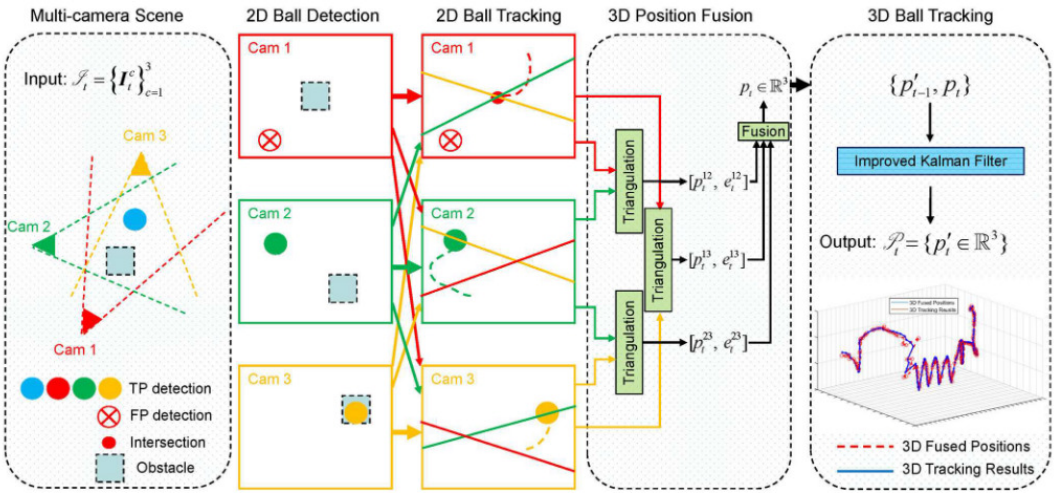


Figure 3.6: Multi-camera 3d ball tracking framework from [81]'s proposed pipeline.

The multi-camera 3D ball tracking framework presented in [81] introduces a 2D ball tracking methodology that aims to detect and remove false detections and even compensate them by considering the cross-view information from other viewpoints based on their epipolar constraints. First, it labels each detection as a false positive (FP) or a true positive (TP) by comparing the current detection with the previous ones. If an outlier, it is labeled as a FP detection, TP otherwise. If no previous detections are available, it could also be considered as a TP detection if it is close to the intersection of all the epipolar lines that correspond to other TP detection from other viewpoints. Second, in case it was labeled as a FP, the detection can still be compensated by the intersection point of epipolar lines from two or more TP detections. [81] also makes use of a novel 2D ball tracker, based on ECO tracker [22], that aims to handle missed detections when the ball is fully occluded from most views.

Before fusing all the 3D positions triangulated from each camera pair, [81] evaluates the accuracy for every obtained position. This accuracy is measured by the reprojection error of the triangulated position by comparing the reprojection of the calculated point with the original detected point in the 2D image. The smaller the distance, the better. With this error, an evaluation of each of the triangulated points is done by excluding the ones that are associated with a bigger reprojection error than the threshold defined. This is because a larger reprojection error typically implies a smaller probability that the two image observation points are projected from the same point in 3D space. Figure 3.6 shows a full example of [81]'s framework.

Instead of using the reprojection error as a criterion for rejecting or not each one of the 3D position triangulated, [3] uses it to attribute weights to each one of those points. Those weights are then used on a weighted average to fuse them and find a robust and accurate 3D ball position.

Unfortunately, [81] does not isolate either of its improved steps, so it is not possible to evaluate their performance fully.

3.2.4 Monocular approach

Monocular camera setups are simpler and more cost-effective compared to multi-camera setups. They reduce the hardware requirements and make the system more accessible. However, recovering the 3D trajectories from monocular videos is difficult since triangulation is not feasible for objects positioned off the ground.

3.2.4.1 Ballistic Trajectories Estimation

In both considered racket sports, shots' trajectories can be approximated to a ballistic one. However, they are not ideal due to factors like air resistance and the Magnus force from the ball's spin, which can be significant depending on the conditions. Vid2Player [92] is a system that uses this assumption to estimate a Tennis ball trajectory so it can later generate novel points between professional Tennis players. For that effect, [92] uses the knowledge of the ball contact times and the player's position and pose at that same time to then estimate the trajectory between two consecutive ball contacts. Since a Tennis rally may contain multiple shots, segmenting these shots is an activity recognition problem. Instead of addressing this problem, [92]'s dataset already has the ball contact times human-annotated.

Vid2Player starts by estimating the 3D location of the ball at the time of contact by making use of the swinging player's court position and pose. Then, the ball's trajectory between two consecutive ball contacts is estimated by fitting one (or two, depending on if there is a bounce) ballistic trajectory to the start and end contact points determined previously. The physical model adopted for the ball flight takes into account cases of topspin (ball rotating forward) and backspin(ball rotating backward) shots that were already human-annotated on the dataset. It is worth noting that any ball sliding along the court that occurs during a bounce is not modeled.

Badminton is an even more challenging racket sport to extract the 3D trajectory of the object of interest. In Badminton, the shuttlecock is not allowed to touch the ground, is heavily affected by air drag, the damage taken may lead to unpredictable paths, and the frequent jumps of the players turn methodologies like [92] unfeasible. Monotrack [44] tackles these problems, resulting in the reconstruction of the trajectory of shots from unlabeled videos.

[44] first identifies the court, players' poses, and shuttlecock 2D position (method described previously on 3.2.2.3) to use them as input of a recurrent network, called HitNet (Hit detection architecture), and retrieve the segmentation of shots. This GRU-based [16] recurrent network, as shown in Figure 3.7, classifies each frame as one of these labels: *no hit*, *near player hit*, and *far player hit*. MonoTrack's authors also realized that optimization based on statistical observation and the sport's own rules could be made. The three main constraints that were imposed on this optimization were: implying that a rally lasting S has approximately S ball contacts since the average duration of a shot is around 1 second; two consecutive hits can not be too close in time; and the hits should alternate between the players.

To evaluate HitNet, [44] performed ablation studies to evaluate the significance of each feature (court, pose, shuttlecock) and also compare them to some baselines, a derivative-based method and

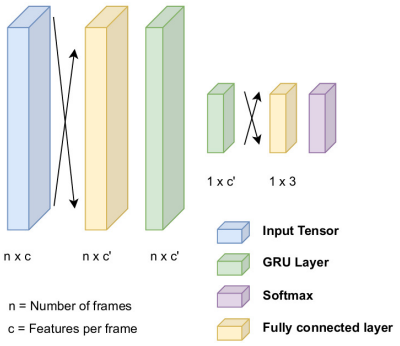


Figure 3.7: Architecture of Monotrack's hit detection model [44].

a Random Forest (RF) based classifier. As it is shown in Table 3.4, HitNet undoubtedly outperforms the baselines. Not only that, but it also demonstrates that both the features and especially the optimization were beneficial to detecting the hits.

Table 3.4: Comparison of HitNet over baseline and ablation study [44].

Model	Recall	Acc.	Pre.	F1
Derivative-based	65.7%	53.8%	74.7%	0.699
RF	65.1%	57.4%	83.0%	0.730
HitNet (Shuttle)	70.2%	68.8%	97.4%	0.815
HitNet (Shuttle+Pose)	73.9%	75.6%	97.1%	0.850
HitNet (All)	78.1%	76.4%	97.2%	0.8866
HitNet+optimization	94.3%	89.7%	94.9%	0.946

As for 3D reconstruction, MonoTrack's algorithm uses the reprojection error of the 3D trajectory estimated by a physics model, to then refine it. An optimization based on the sport's rules and physics is also made, taking into account the maximum velocity, the maximum height, the initial velocity direction, and the initial position of the shuttle. The results for this approach, which are shown in Figure 3.8, show that longer flight times result in a significant reduction in error, reaching saturation at approximately 5 centimeters.

3.2.4.2 Ball's Diameter Estimation

Even though it is not based on a racket sport like Tennis or Padel but instead Basketball, a recent work from 2022 [75] aims to localize the ball in the 3D space from a single calibrated image without relying on temporal analysis and ballistic trajectory assumptions. Instead, it divides its task into two distinct analysis problems: ball detection and ball diameter estimation.

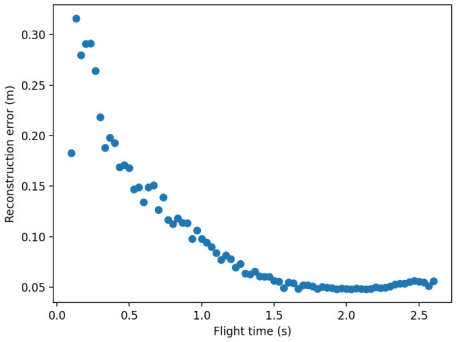


Figure 3.8: 3D Reconstruction error by shuttlecock flight time [44].

To detect the ball in the image space, it adopts a solution already developed by the same authors, BallSeg [74], and to estimate the ball diameter in pixels, it compares two methods: the Hough circle transformation and a proposed CNN-based method.

As a baseline, the work applies the Hough circle transformation (HCT) [24] to the top ball candidate on the heatmap generated by the BallSeg detector to estimate the ball diameter. HCT is a familiar and established method for estimating the size of a circular and highly contrasted object. In practice, the method is only applied after the application of a morphological opening with a circular filter operation followed by an edge detector operation on the heatmap generated.

The ICNet-based [94] CNN method proposed is designed to regress the ball diameter, in pixels, from an image patch centered on the candidates proposed by any conventional ball detector. Since the number of candidates, in the form of bounding boxes, can be more than one, the CNN proposed is also trained to predict whether the input patch contains a ball or not. This is due to a situation like that leading to most candidates being false positives. This design decision is valuable in situations where a simple top-1 detection strategy would yield a false positive, making it possible to recover the correct detection from the other candidates.

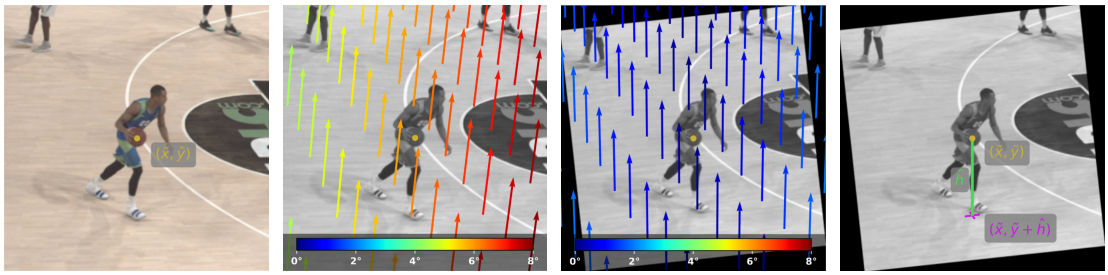
Given that the real ball size and the camera parameters are known, after detecting and estimating the size of the ball, in pixels, the work 3D localizes it. It uses the 3D rays of the ball center $\mathbf{b}^c$ and two diametrically opposed ball edges $\mathbf{e}^c_-$ and $\mathbf{e}^c_+$ to calculate the 3D location of the ball $\mathbf{b}^o$, for a camera placed at $\mathbf{c}^o \in \mathbb{R}^3$ in world coordinates system:

$$\mathbf{b}^o = R^T \cdot \frac{\phi \cdot \mathbf{b}^c}{\mathbf{e}^c_{+\mathrm{y}} - \mathbf{e}^c_{-\mathrm{y}}} + \mathbf{c}^o \tag{3.1}$$

The results concluded that the proposed CNN was indeed more accurate and precise than its baseline, HCT: 2.3 $\pm$ .2 versus 6.3 $\pm$ 1.0 $\mathrm{MAE}_{[m]}$, respectively, on the DeepSort[3] test set.

Even though Basketball is prone to more occlusions than the targeted racket sports, the 2.3 meters Mean Absolute Error (MAE) makes it unsuitable for the task at hand when considering that the blurrier and smaller balls from these racket sports would most likely yield worse results.

[3] https://kaggle.com/code/sakshaymahna/deepsort



(a) (b) (c) (d)

Figure 3.9: The predicted height and the given 2D position are converted to 3D coordinates using the camera calibration data [8].

3.2.4.3 Context-Aware 3D Ball Localization

A year later, in 2023, the same team decided to solve the same problem but by predicting the ball's height, in pixels, in image space instead. This way, several limitations, like sensitivity to variations in the ball's size within the image, issues with motion blur and image resolution, and the requirement for prior knowledge of the ball's real-world size, could be addressed. The work predicts the ball's height by estimating its projection onto the ground plane within the image, leveraging the image itself and the ball's position as inputs. The 3D coordinates of the ball are later reconstructed by exploiting the already-known camera matrix.

The model used to estimate the ball's height, in pixels, has a CNN VGG[65] backbone to extract image features. These features are then processed by a regression head, supervised to predict the distance in pixels between the ball and its ground projection in the image space. The model operates on square image crops centered on the ball to ensure ample contextual information is considered.

After estimating the ball's height, in pixels, this distance is then used to retrieve the 3D ball coordinates, see Figure 3.9.

Initially, the input image undergoes undistortion to ensure that straight lines in the real world correspond to straight lines in image space, see Figure 3.9b. Subsequently, the image is rotated, aligning the Z-axis in the real world with the vertical direction y in the image space, see Figure 3.9c. Following rotation, the predicted height of the ball in pixels, represented by $\hat{h}$, effectively signifies the vertical displacement between the ball's position in the image $(\tilde{x},\tilde{y})$ at the center of the crop and its projection onto the ground in image space $(\tilde{x},\tilde{y}+\hat{h})$, see Figure 3.9d. Given the camera calibration, $(\tilde{x},\tilde{y}+\hat{h})$ is projected into 3D space on the $Z=0$ ground plane, determining the position of the ball's projection on the ground $(\tilde{X},\tilde{Y},0)$. Finally, to obtain the 3D world coordinates of the ball, a reprojection process is executed. The path of the light ray passing through the ball in the image space is traced, determining the two 3D points where it intersects the $Y=\tilde{Y}$ and $X=\tilde{X}$ vertical planes. By averaging these two points, the 3D coordinates of the ball are retrieved.

The baseline used for this work's approach is the method described previously in 3.2.4.2, and the results retrieved clearly demonstrate the superiority of the proposed method, see Table 3.5.

Table 3.5: Comparison of performance metrics between the proposed method and baseline model [8].

Metrics	**Proposed**	**Baseline**
MA3DE [m]	1.29 $\pm$ 0.10	2.97 $\pm$ 0.23
MdnA3DE [m]	0.95 $\pm$ 0.16	2.18 $\pm$ 0.40

[8] also analyzes the influence of the crop size chosen and image scale on performance. Even though it concludes the expected behavior in crop size variance, that a small crop size is prejudicial to performance but increasing it eventually reaches a plateau, it also concludes that this approach is robust to image scale, while in [75], high-resolution images were a prerequisite for achieving accurate ball localization.

3.3 Players' Positions and Poses Estimation

In sports motion analysis, the importance of precise position and pose estimation is of paramount importance, since they form the foundation for understanding and improving athletic performance. For this purpose, and similarly to the ball's localization process 3.2, both monocular and multi-view camera systems were considered.

3.3.1 Datasets

Given the necessity to train the 3D HPE models and also the need to benchmark them, proven datasets are crucial. These datasets are useful for their established quality, quantity, and variability, and given their widespread usage in evaluating and comparing various approaches, they also serve as key benchmarks for comparison. Unfortunately, 3D HPE datasets are not as varied as 2D ones since they are usually captured in a lab environment using Motion Capture systems [6]. This environment may limit variations in background, viewpoints, and lighting, bringing down the generalization potential of the trained models. The following list details the most popular public 3D HPE datasets:

Human3.6M [32, 6, 14] is a large-scale 3D single-person HPE dataset that contains about 3.6 million 3D human poses and corresponding video frame, from 4 different RGB views. These 3D human poses, captured by a high-speed motion capture system, correspond to 11 professional actors during 17 different activities, such as sitting, walking, eating, smoking, talking on the phone, etc. Main capturing devices comprise 4 digital video cameras, 1 time-of-flight sensor, and 10 motion cameras operating synchronously. The capture area spans approximately 4 by 3 meters. The provided annotations include essential details such as 3D joint positions, joint angles, person bounding boxes, and 3D laser scans for each actor. The Human3.6M dataset is a widely used benchmark for 3D HPE models, usually measured by the Mean per Joint Position Error (MPJPE) in millimeters.

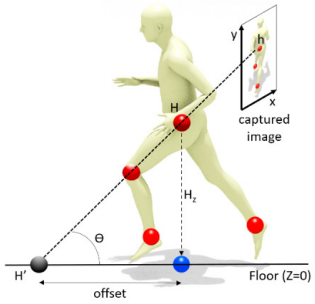


Figure 3.10: [81]'s definition of 2D position of the player and offset.

HumanEva-I [64, 14] is a 3D single-person HPE dataset that contains synchronized video sequences from multiple camera views, seven of them grayscale and the other three RGB. The poses are performed indoors by four subjects performing six common actions such as walking, jogging, throwing, catching a ball, etc. An extension of this dataset, HumanEva-II [64, 90] exists and is used for testing.

CMU Panoptic [36, 44] is a large-scale multi-view and multi-person 3D HPE dataset. It contains 65 videos (5.5 hours) captured from a dome containing 480 VGA cameras (25 FPS), 31 HD cameras (30FPS), 10 Kinetic2 Sensors (30FPS), and 5 DLP Projectors, resulting in 1.5 million 3D skeletons. 17 of these 65 video sequences are in multi-person scenarios with socially interacting groups.

3DPW [79, 44, 14] is a single-view multi-person 3D HPE dataset, captured mostly outdoors by a single hand-held camera, using IMU sensors attached to the subjects' limbs to compute pose and shape ground truth. It consists of 60 video sequences with day-to-day actions like walking, sitting, going up the stairs, dressing up, etc.

3.3.2 Position Estimation

Retrieving the player's position from a multi-view camera system can be quite similar to the 3D ball localization step since each joint can be treated as an independent point, to which triangulation can then be applied. Another simple approach is to position the entire pose relatively to the 3D location of a single point, e.g., the right ankle. For monocular approaches, though, the task becomes more complex, and, therefore, more creative approaches are needed.

Except for the *split-steps* and *smash* jumps, a player always has a contact point with the ground, something that would turn the ball localization problem straightforward. Such work that aims to estimate the 2D position of players within the Padel court, based on the assumption that the player never jumps, is [34]. This work evaluates and compares the estimations of players' 2D positions from either their 2D bounding boxes, their segmentation masks, or their poses.

[34] considers a player's 2D position as the vertical projection of their hip, a common output for pose estimation methods, onto the floor, as it is seen in Figure 3.10. In the same Figure, it is also possible to observe the offset, which is the intersection of the epipolar line that passes on the hip H with the floor.

The pose estimation approach calculated the 2D position p as:

$$p = (H_x, \frac{1}{2}(L_y + R_y)) \tag{3.2}$$

Here, H_x is the X coordinate (width) of the hip joint, and L_y and R_y are the Y coordinates (length) of the left and right ankle joint, respectively.

Of the three approaches analyzed by [34], the pose estimation approach was the one that yielded the best results, with random errors below 11 centimeters, compared to the 18 centimeters of the 2D bounding boxes approach.

3.3.3 2D Multi-Person Pose Estimation

2D multi-person pose estimation focuses on detecting and mapping the 2D coordinates of body joints for multiple individuals within a single image or video frame. Unlike single-person pose estimation, this task must handle the complexities of distinguishing multiple individuals' joints in crowded scenes. It also involves addressing challenges such as overlapping bodies, diverse poses, and different appearances and scales among the people in the scene.

One of last year's most recent and successful research on this field was RTMPose [35], a 2D real-time multi-person human pose estimator. RTMPose employs a top-down approach, utilizing an off-the-shelf detector to acquire bounding boxes and subsequently estimate the pose of each individual. While traditionally considered accurate yet slow, [35] state that leveraging highly efficient real-time detectors like [47] and [48] that the detection process is no longer a bottleneck for inference speed in top-down methods.

[35] uses CSPNeXt [47] as the backbone, which was originally designed for object detection purposes. This research indicates that, while effective, backbone architectures originally designed for image classification may not be optimal for dense prediction tasks such as object detection, pose estimation, and semantic segmentation. While some backbones achieve high accuracy on pose estimation benchmarks, they often come with drawbacks such as high computational cost, inference latency, or deployment challenges. CSPNeXt demonstrates a favorable balance between speed, accuracy, and ease of deployment, as highlighted by the RTMPose research.

RTMPose uses a SimCC-based [41] algorithm to locate keypoints, treating the task as a classification problem. This approach achieves competitive accuracy with lower computational effort compared to heatmap-based algorithms [30, 83, 84, 91]. Its inference pipeline is also optimized to reduce latency and enhance robustness, using techniques such as skip-frame detection and pose Non-Maximum Suppression (NMS).

Multiple RTMPose models have been made available, with the aim of catering to a wide range of application scenarios, ensuring an optimal balance between performance and speed. One of the main advantages of this work is that it is based on MMPose, allowing for an ease of use that other state-of-the-art models lack.

MMPose

MMPose [19] is the most popular, according to *Github*[4], open-source pose estimation toolbox available, and it is part of the *OpenMMLab* [12] project.

OpenMMLab is a comprehensive open-source toolkit for computer vision tasks primarily developed by researchers at the Chinese University of Hong Kong. In addition to MMPose, it includes several other toolboxes, such as MMDetection [18] for detection, MMTracking [20] for tracking, and MMSegmentation [20] for semantic segmentation. These toolboxes offer a wide range of state-of-the-art algorithms and tools for various computer vision applications.

MMPose, along with the other *OpenMMLab* toolboxes, is implemented in Python and utilizes the PyTorch library as the primary artificial intelligence framework. Additionally, MMPose provides both the implementation and pre-trained weights of numerous state-of-the-art models for both 2D and 3D pose estimation, offering researchers and developers a comprehensive toolkit for pose estimation tasks.

3.3.4 3D Multi-Person Pose Estimation

3D multi-person pose estimation involves predicting the 3D locations of body joints for multiple individuals in input images or other sources. This task is inherently more challenging than its 2D counterpart, as it requires predicting depth information of body joints. It is also more laborious than the equivalent single-person pose estimation since it needs to solve two additional sub-tasks [86]: identifying joint-to-person associations across different views and managing mutual occlusions within a crowded environment.

Human pose estimation is a widely concerned research topic [90], but only a relatively small part of it is dedicated to real-time 3D pose estimation, which is necessary for the sports video and highlights tools like the one this document is aiming for. Given both demands, multi-person and real-time, state-of-the-art like MotionBert [95] and TesseTrack [60] cannot be considered appropriate to solve the problem at hand. In this section, state-of-the-art real-time 3d pose estimators are described for both monocular and multi-view systems.

ROMP [68, 44] is a real-time monocular one-stage method for multi-person human mesh recovery. Within ROMP, and as seen in Figure 3.11, individuals' 2D body center positions and 3D mesh structures are expressed through a 2D Body Center Heatmap and a Mesh Parameter map, respectively. This clear center-based representation ensures precise pixel-level feature encoding. By employing a simple parameter sampling process, [68] retrieves the 3D body mesh parameter vectors for all individuals from the Mesh Parameter map at the body center positions delineated by the heatmap. Subsequently, these sampled mesh parameter vectors are merged into the SMPL [46] body model to generate multi-person 3D meshes. To address crowded scenarios with significant overlap, the representation's disambiguation is improved by separating the collided body centers.

[4] https://github.com/

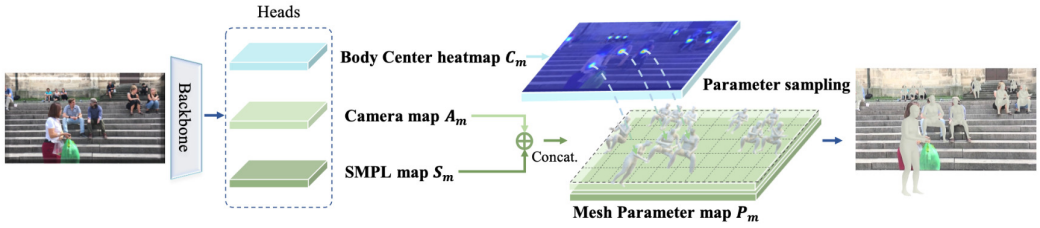


Figure 3.11: Overview of ROMP approach [68].

VoxelPose [72, 90, 6] is a top-down 3D multi-person pose estimation model that works directly in 3D space and takes as input multiple camera views. The initial stage involves the estimation of 2D heatmaps for each viewpoint, capturing the per-pixel likelihood of all joints. Then [72] takes a unique approach by projecting the heatmaps from all views into a shared 3D space. This yields a feature set that eases accurate estimations of the 3D positions of all joints. The process includes utilizing a Cuboid Proposal Network (CPN) to identify individuals within the scene. The CPN predicts multiple 3D cuboid proposals derived from the 3D feature volume. Subsequently, for each proposal, a Pose Regression Network (PRN) is employed to estimate the 3D pose based on a volume of characteristics. This approach can be seen in Figure 3.12.

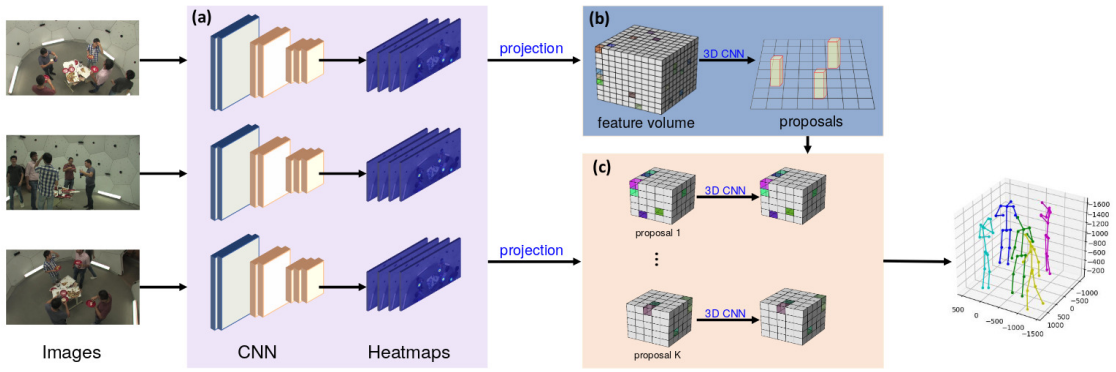


Figure 3.12: Overview of VoxelPose approach [72].

To address the heavy computation burdens of VoxelPose, Faster VoxelPose [86] re-projects the feature volume onto the three 2D coordinate planes and separately estimates X, Y, Z coordinates from them. For this purpose, [86] initiates the process by determining the 3D bounding box for each person. This involves estimating a 2D box and its height by utilizing volume features projected to the xy-plane and z-axis, respectively. Subsequently, for each individual, [86] independently estimates partial joint coordinates from the three coordinate planes. These partial estimates are then fused to derive the ultimate 3D pose. This results in a ten times faster [86] model than the original VoxelPose on common benchmarks, with the particularity that it scales well to larger scenes.

3.4 Summary

In this chapter, state-of-the-art approaches to both problems, retrieving the ball trajectory and players' positions and poses, were described for each of the possible camera setups. Naturally, more attention to detail was given to the ball localization through a single viewpoint as it was deemed the highest hurdle to achieve this work's objectives. Even though 2D ball detection has seen great results throughout the last few years, with each development based on the last, from TrackNet [31] to WASB [69], monocular 3D ball localization hasn't. The research has been scarce and with sport-dependent methodologies. Having said this, MonoTrack's [44] shot segmentation is an interesting approach that, instead of making weak estimations for the shuttle 3D position during mid-air, when the error is at its highest, it estimates using only the start and end 3D contact points of each shot, as well as the shuttle flying time and 2D trajectory. Also, due to the closed court of Padel, this shot segmentation could prove useful in identifying wall hits.

Unexpectedly, increasing the number of viewpoints significantly increases the feasibility of the work's objectives due to the decrease in occlusions, the possibility of triangulation, and error reduction. Still, the 2D ball detection and 3D ball tracking steps would remain unchanged as they don't have to deal with the extra information received.

Although 3D human pose estimation is a highly researched field, the demand for a reliable monocular real-time multi-person pose estimation is currently unrealistic due to its significant challenges. At this time, such an approach is deemed unfeasible for this work, leaving 2D pose estimation as the more viable option. Furthermore, it is hard to judge the impact the high-performance requirement for pose estimation would have on the final results, so when the time arrives, a thorough analysis of the results will be done to reevaluate the method chosen and, if necessary, the requirement.

Chapter 4

3D Representation Acquisition

The methodology and development sections are pivotal in a dissertation as they outline the path taken from the initial research proposal to the proposed solution, and from there to the final state of development that led to the results obtained. This is crucial as it provides sufficient detail for replication by others, ensuring transparency and credibility in the research process.

The dissertation elaboration started with understanding the research proposal thoroughly; to that effect, various reunions with *MOG Technologies*, the company where the dissertation was elaborated, occurred. This ensured not only to provide the most relevant findings for the industry, but also to narrow the relevant research topics of the state-of-the-art as much as possible.

The initial research primarily aimed to assess the feasibility of accomplishing the task using a single camera, given its potential advantage for future market applicability. For that purpose, the main research focus was the state-of-the-art in 3D reconstruction for ball tracking and pose estimation. While the literature on pose estimation was extensive and varied in methodologies, the literature on 3D reconstruction was more limited. This was due to the unique challenges posed by factors such as ball size, blur, and occlusions, which differentiate this area from the more established fields of detection and tracking. These challenges were addressed in a few article selections detailed in the State-of-the-Art chapter. Also, fundamental concepts shared across these topics were identified and documented in the Background Knowledge chapter throughout the research process.

Then, after thorough research of the state-of-the-art, a preliminary proposed solution and future work plan were formulated and discussed with the team to gather their insights and feedback. This process led to adjustments in the scope to mitigate potential time constraints.

At last, the dissertation's proposed solution development and validation proceeded as scheduled and as described throughout this chapter and the following Tests and Results chapter.

The following sections of this chapter give a deeper definition of the problem, detail the proposed solution for the dissertation's development phase, and explain the multiple components of the final result on a lower level.

4.1 Problem Definition

In the domain of sports, specifically in the envisaged racket sports, traditional methods of replay and analysis are often confined to two-dimensional perspectives, offering a limited view of the intricate dynamics involved. This limitation poses challenges for both players and spectators in fully understanding the nuances of a match. Operating as a sports video replay and highlight tool prototype for the game of Tennis and Padel, this work aims to improve the players' and spectators' overall amusement and analysis capabilities by providing a 3D reconstruction of plays in a relatively short time. This reconstruction should include the play's static elements, such as the court and net, and dynamic ones, such as the players and ball. These dynamic agents' representations must be constantly updated according to the information captured, using ball localization and human pose and position estimation methods.

Information collected during a game play could come from various sensors like accelerometers, gyroscopes, microphones, and, most notably, cameras. For the problem at hand, only a calibrated RGB camera is used due to its accessibility, which is particularly beneficial for amateur players. However, significant challenges such as motion blur and occlusions arise, which are critical hurdles to address. These issues, along with the difficulty of reliably estimating the ball's location in the air, require robust and out-of-the-box methodologies to ensure sufficient accuracy to not undermine the fidelity of the original play.

4.2 Proposed Solution

After the literature review realized during the dissertation's preparation phase and described in the two previous chapters, a better understanding of the multiple ways to tackle the problem was obtained.

The problem is clearly divided into two sub-problems: multi-person human pose estimation and 3D ball localization. In the case of the first sub-problem, the prototype relies heavily on the performance of state-of-the-art multi-person real-time pose estimation methods currently available. The real performance of these methods applied in the work's final prototype is difficult to predict from the available results in the literature. This is due to the clear difference between the datasets used to evaluate them and the highly specific environment in which this work intends to apply them. As there aren't any public datasets that would allow for optimal evaluation of these scenarios, and creating one is impractical with the resources available, due to the intensive annotation work required, a state-of-the-art model will be employed initially and will be retained if its results suit the project's needs.

From what was researched in the literature, works that handled the second sub-problem, 3D ball localization, for the game of Tennis or Padel were only found for multi-view camera setups. Having said this, a monocular approach was still deemed possible due to the research done on other sports domains, as long as the camera is calibrated. Taking advantage of the novel Badminton's MonoTrack's [44] play segmentation approach, the proposed pipeline is present in Figure 4.1.

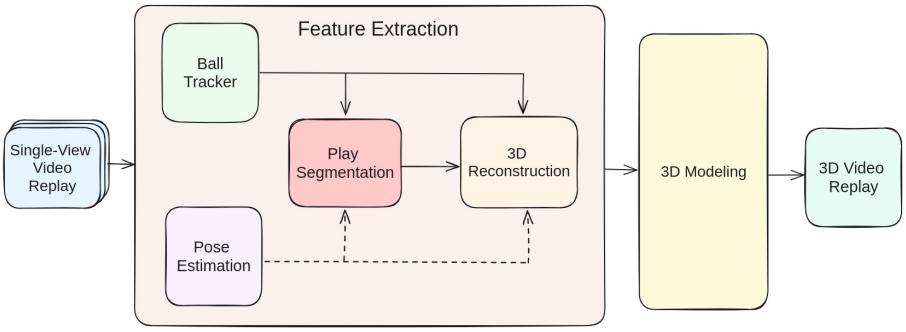


Figure 4.1: Proposed solution pipeline.

The proposed solution is composed of 5 tasks/modules, 4 of which are dedicated to the play's feature extraction and can be described as:

Ball Tracker: It is the cornerstone of the solution as the rest of the tasks depend on it, and so do their results. This task is responsible for detecting and tracking the ball in the image space throughout the entire play while avoiding obstacles such as motion blur and occlusion.

Pose Estimation: This task involves retrieving the 2D poses and positions of players from the video. To accomplish this, a pose estimation model and a multi-object tracker (MOT) are used to group the multiple poses detected in all the video frames according to the players. If the poses identified were from the players, and no bystanders, and if they were accurate without losing tracking. Additionally, it involves ensuring the accuracy and quality of player tracking and poses despite challenges such as occlusions, motion blur, and varying scales of players in the clip frames.

Play Segmentation: By using multiple frames' 2D coordinates of the ball, court corners, and players' poses as input, a recurrent network is trained to output the status of the ball. Throughout a play, the ball can be in various states: in the air, bouncing on the ground, bouncing on the walls, or being hit by a player. This approach aims to tie the ball's 2D image position in specific frames to known 3D entities of the court, providing sufficient information to accurately place the ball in the 3D real space at those frames.

A play segmentation model like MonoTrack's HitNet [44] for Tennis and Padel is one of the novelties of this solution, as there isn't any existing literature of this genre according to the research conducted. Yet, the increase in the dimensionality of the models' output classes poses a significant challenge for this task, potentially limiting its generalization and subsequent performance.

3D Reconstruction: This task aggregates all the information extracted from the previous steps, including the ball's detected coordinates, the players' positions, and the ball's status to retrieve the ball's real location at any time during the play. This is achieved by employing a physics model that combines this data with the calibrated camera parameters to extract the ball's trajectory between hits, utilizing line-of-sight calculations and the reprojection error.

Challenges include ensuring the physics model faithfully represents the ball's movement behavior and accurately estimating trajectories between player contacts.

As far as the literature review indicates, this method is the first of its kind for racket sports.

3D Modeling: At last, the 3D Modeling task combines the ball's location and players' positions and creates a proof of concept of a 3D representation of a racket sports replay.

This approach encompasses three evaluations: the first for the 2D ball detector, the second for the *hits* segmentation, and a third for the 3D reconstruction of the trajectory. Similar metrics such as precision, accuracy, recall, and F1 will be used for the first two evaluations since they can be calculated from the resulting confusion matrix. In the case of the third one, the 3D reconstruction of the ballistic trajectory, the reprojection error, also known as mean absolute error (MAE), and the mean absolute 3D error (MA3DE) will be employed. Also, due to the high-performance requirement, metrics such as execution time and frames-per-second (FPS) will be used to quantify the performance of each of the solution's modules, as well as the entire resulting prototype.

All of this will only be possible with a Tennis and Padel dataset tailored for the same purpose. This dataset needs to consist of multiple views, each annotated with the status of the ball (e.g., no hit, ground hit) and its position for each frame. It should also be diverse enough, with varying backgrounds, players' equipment, and courts, to ensure that any model trained on it can possibly generalize.

4.3 Development

Although the previous section provided an introduction and high-level overview of each task required for the final prototype, much more remains to be discussed. This includes detailing the steps taken, decisions made, and approaches adopted that were not thoroughly addressed earlier. Therefore, this section is dedicated to providing a comprehensive understanding of the work undertaken, ensuring that it is sufficiently detailed and documented to be reproducible by peers.

4.3.1 Dataset

The different AI modules proposed in this dissertation require an adequate dataset to ensure their effectiveness and generalizability across diverse settings and situations. Given the specific needs of this research, it was necessary to gather a tailored dataset, as no publicly available dataset fully met the requirements. This section details the dataset's organization, sources, and the preprocessing steps undertaken to prepare it for the models' evaluation and training.

4.3.1.1 Annotated Data

Annotated data refers to relevant information extracted from the available videos. This data can be used to train and evaluate components throughout the pipeline and is archived into CSV files. For the various proposed modules, three categories of annotated data are essential for each frame:

- **Ball Position:** Precise pixel coordinates of the ball's position within each frame were annotated, identifying whether the ball was in-frame and visible, in-frame but occluded, or out-of-frame. Additionally, the tool annotated inactive or stationary balls, which are present in a frame but not in play—a common occurrence in Padel. Segments examples of these annotations can be seen in Table 4.1.

  All this information is critical for tracking the ball's movement throughout the game and is essential for both training and evaluating the ball detection models.

Table 4.1: Segments of ball positions CSV files.

(a) Active ball positions.

Frame	**Ball**	**x**	**y**
0	0	0.700	0.709
1	0	0.703	0.707
2	0	0.704	0.707
3	1	0.703	0.705
4	1	0.704	0.703
5	1	0.705	0.701

(b) Inactive ball positions.

inactive_ball_no	**x**	**y**
1	0.315	0.479
2	0.734	0.485

- **Hit Label:** Detailed labeling of the ball's state in each frame. This includes:
  - **No Hit:** The ball is in motion or stationary without being in the process of being hit.
  - **Ground or Walls Hit:** The ball is currently bouncing on a stationary surface, like the ground or the walls, having each of these surfaces an according label.
  - **Player Hit:** A player has just struck the ball. The side from which the player hits is also identified.
  - **Net Hit:** The ball makes contact with the net.

Table 4.2: Segments of hits labels CSV Files.

Frame	**Hit**
0	1
1	1
.	
13	1
14	4
15	1

This label is essential for the training and evaluation of the model responsible for segmenting a given play into segments of predictable trajectory, and an example of it can be observed in Table 4.2.

- **Court Points:** Accurate pixel coordinates of specific reference points on the court, such as corners, center lines, and net positions, to allow the projection matrix calculation for the 3D reconstruction module. This label is shared throughout the frames that share the same camera, and its values are ultimately compared to the real-world coordinates of the points identified in the frame. In Table 4.3, segments of each annotation file are represented.

Table 4.3: Segments of court points CSV files.

(a) Real court points world positions.

BLLC_X	BLLC_Y	BLLC_Z	BLRC_X	BLRC_Y	...
0	0	0	10	0	...

(b) Image court points positions

BLLC_X	BLLC_Y	BLRC_X	...
0.0723	0.9796	0.9244	...

While useful for the project, human keypoints annotation was discarded due to the vast amount of effort required. Labeling 17 (COCO's human model [43]) for each player in thousands of frames was simply not feasible for this work's designated duration. Still, an alternative was found, as is seen in the next section.

4.3.1.2 Sources

The creation of the final dataset began by combining multiple existing datasets, followed by manual labeling when missing. The use of a completely new, self-collected dataset was avoided not only due to the significant time and effort required but also to ensure diversity. Existing datasets encompass various court surfaces, equipment colors, camera positions, and resolutions, which would be challenging to replicate independently. This diversity is crucial for the models' ability to generalize effectively, as it exposes them to a wider range of conditions and scenarios.

PadelVic PadelVic [33] is a recent multi-camera Padel dataset based on a 1-hour long amateur match. It comprises 4 videos from different camera angles, as seen in Table 4.4, estimated court positions for the players, precise motion capture data for one player (obtained using a professional MoCap system), and synthetic videos with ground-truth positions and poses.

Table 4.4: PadelVic's [33] captured datasets description.

Dataset	Device	Height	Resolution	FPS
Main Camera	Panasonic AG-UX180 4K	6 m	3626x1960	50
Second camera	GoPro Hero9 Black	7.5 m	2806x1870	60
Court level 1	Samsung Galaxy S22+	1.45 m	1920x1080	60
Court level 2	iPhone 13 Pro Max	1.45 m	1920x1080	60
Motion Capture	Xsens MVN Awinda	—	—	60

As it can be seen in Table 4.4, the cameras, although positioned on the same side of the court, were positioned at different heights and were also inherently different, varying in both resolution and frame rate. In Figure 4.2, it is possible to observe the distinct positions and resolutions of the cameras, as well as variations in color calibration.

(a) Main camera.

(b) Second camera.

(c) Court level 1.

(d) Court level 2.

Figure 4.2: PadelVic's [33] cameras' frame samples.

One of its main limitations is that the video streams show only an approximate synchronization, not a per-frame sync, due to the cameras' lack of support for external sync sources. It is unavailable, but it would be interesting to see if a camera placed on the other side of the court would yield better results.

For each camera, court points were annotated to ensure accurate spatial referencing. The first sixty plays of the match were segmented across all the video streams. The hit status was labeled for every play, and these labels were shared across the different camera angles of the same play.

Figure 4.3: Public Padel plays' frame samples.

This consistency was maintained because events such as ball bounces occur simultaneously across all camera views. Then, the ball position at each frame was annotated in six plays of three cameras, resulting in a total of 10,872 frames labeled.

Padel Public Plays Since PadelVic's dataset consists of a single amateur match, it lacks diversity in court floors, backgrounds, and equipment colors that will not help generalize the proposed models. To address this, ten additional 23.98 FPS Padel clips found in a public compilation video[1] were fully annotated (court points, ball position, and hit label annotations), resulting in an extra 4,374 1080p frames. The desired court and background diversity can be seen in Figure 4.3.

TrackNet's Dataset During the research of Tracknet [31], the largest development in ball detection in the last years, as shown in the State-of-the-Art chapter, a Tennis dataset composed of 10 matches at 23.98 FPS, each with 4 to 15 plays, totaling 19,740 720p frames. This public dataset includes annotations for the ball position and visibility in a format similar to the desired one, as well as annotations for bounces and player *hits*, though it does not specify which side the hits are from as specified in 4.3.1.1. This was quickly solved by identifying who made each play's first *hit* and then alternating it for the next player *hits*.

In Figure 4.4, it is possible to observe a sample of 4 of 10 ten matches available.

4.3.1.3 Annotation Tools

Manual annotations for each data categorization were made using two simple annotation tools. Initially adapted from a labeling tool for ball positioning [11], one tool was enhanced to support hit annotations as well. This tool was then further modified into a separate tool specifically for

[1] https://www.youtube.com/watch?v=geTTr4ACMWY

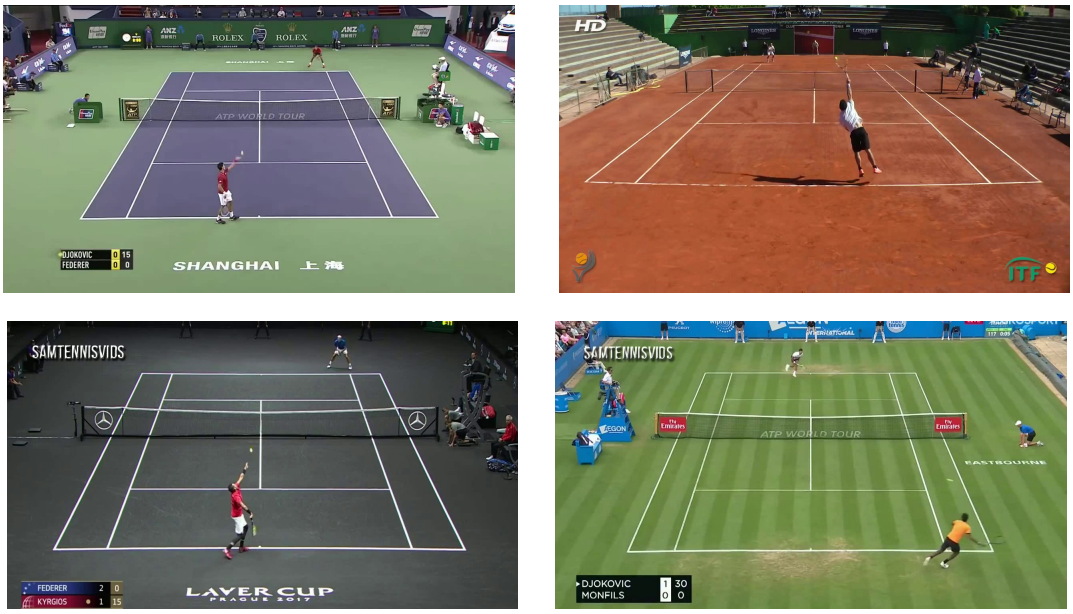
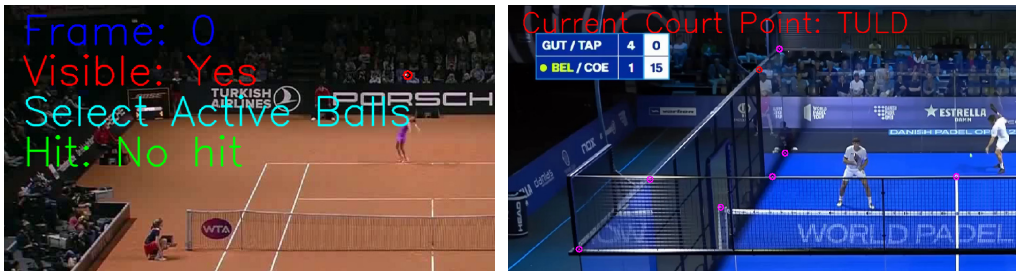


Figure 4.4: Tracknet's dataset's [31] frame samples.

annotating the court points positions for each camera. A normal usage of these tools can be seen in Figures 4.5a and 4.5b.



(a) Ball position and hit labeling.

(b) Court points registration.

Figure 4.5: Annotation tools usage example.

4.3.1.4 Training and Validation Data Split

To prevent overfitting during the training phase of the proposed models, the dataset was divided into training and validation sets following the methodology outlined in the original paper of TrackNet. This division was performed carefully to ensure that various scenarios were evenly represented across both subsets. Specifically, for the PadelVic dataset, a 4:2 split was implemented, while an 8:2 split was applied for the public Padel plays dataset. As for the TrackNet dataset, the last three matches were reserved exclusively for testing purposes, aligning with the approach described in the original work [31]. The final dataset composition is summarized in Table 4.5.

Table 4.5: Summary of all the training and evaluation data collected.

	Train			Test		
	Matches	Plays	Frames	Matches	Plays	Frames
PadelVic [33]	1	4	9,006	1	2	1,866
Public Padel clips	8	8	3,186	2	2	1,188
TrackNet [31]	7	65	14,100	3	30	5,640
Total	16	77	26292	6	34	8694

4.3.2 Ball Tracker

As illustrated in the proposed pipeline Figure 4.1, the ball tracker module is the cornerstone of the proposed methodology. The ball tracker is crucial because it provides the necessary data to segment plays and perform 3D reconstruction, the ball's image coordinates in each frame. Various conditions, such as occlusion and motion blur, can affect performance. It is essential to address these issues early in the pipeline to prevent error accumulation in subsequent steps.

This section discusses the ball tracking models and their associated loss functions and optimizers, as well as the tools and methodologies used to train and test them.

4.3.2.1 Models

This study aims to analyze and determine the best possible approach to detect a ball's position for both Tennis and Padel. This does not mean that these sports must share the same solution, so a study is needed to determine the most suitable one.

Three models were considered: TrackNetV2 [67], MonoTrack [44], and WASB [69]. These models are extensively discussed in the State-of-the-Art chapter, and their respective implementations, which followed the original papers meticulously, have demonstrated the best results for both Badminton and Tennis in [69]'s research.

For this study, an evaluation was conducted to determine the performance of the current models and their pre-trained weights on the gathered Tennis and Padel datasets. The focus was on enhancing the models for Padel, as the optimizations for Tennis were already addressed in [69].

Since ball detection is similar for both considered sports and training these three models from scratch would take considerable time with the available hardware, it was deemed not worth it. Instead, the already trained models' Tennis weights were tuned for the Padel dataset to tackle the unique challenges the sport poses. Due to hardware limitations, the batch size used for tuning was reduced from 8 to 2. The tuning process took 15 epochs and used the optimizers and loss functions described in the original papers [67, 44, 69]:

Table 4.6: Loss functions and optimizers used for training for each model.

Model	Loss Function	Optimizer
TrackNetV2 [67]	Focal	Adadelta
MonoTrack [44]	Binary cross-entropy	Adadelta
WASB [69]	WBCE	Adam

4.3.2.2 Training and Testing Tool

WASB [69], Widely Applicable Strong Baseline, is not only significant for this study as one of the selected models but also highly relevant to this dissertation because the datasets, model weights, and codebase used to achieve its results have been made publicly available[70]. This codebase, written entirely on Python, utilizes the PyTorch [56] library as its primary artificial intelligence library. It includes implementations of various models, including those relevant to this work, as well as multiple loss functions and optimizers. Therefore, it was employed in this work to build upon its robust framework.

The metrics collected for evaluation purposes are based on the confusion matrix, using a pixel threshold to determine correct ball detections. In the original work, [69] used an input frame resolution of 288x512 with a τ threshold of 4 pixels. In this work, an input resolution of 1280x720 was used instead, so the τ threshold chosen was 10 pixels, maintaining approximately the same proportion:

$$\frac{4^2\pi}{288 \times 512} = \frac{10^2\pi}{1280 \times 720} \tag{4.1}$$

It is worth noting that the model's input resolution is fixed, and inference cannot be done at any other resolution. This resolution change is only for dataset loading purposes, with the frames being resized at runtime.

4.3.3 Pose Estimation

For this dissertation's objective of retrieving the 3D ball coordinates during a racket's sports play, the pose and position of the players represent valuable information in evaluating whether a player interacted with the ball and, if so, where it was done. For these purposes, a pose estimation study was conducted to identify an approach to gather this information.

Several 2D pose estimation methods are available, but implementing them and developing all the tools to support them is not feasible for the duration of this work. So, MMPose [19], an open-source toolbox for pose estimation described in Section 3.3.3 was utilized.

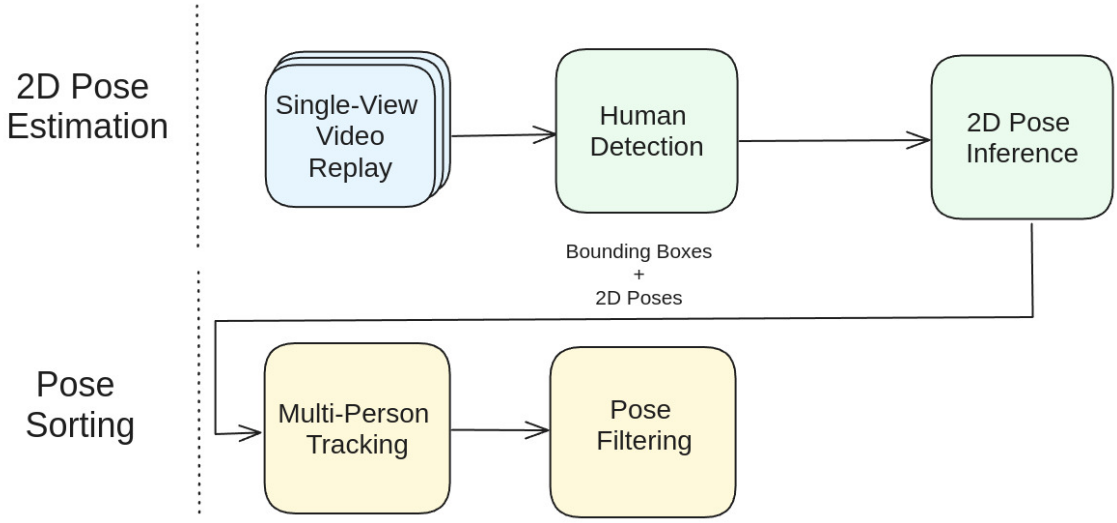


Figure 4.6: High-level block diagram of the players' pose estimation task.

Figure 4.6 shows a high-level block diagram of the process of retrieving the players' poses. This section provides a description of the methodology for both tasks in the diagram: pose estimation and pose sorting.

Unfortunately, the dataset gathered lacks pose annotations for all the players involved, so a quantifiable evaluation was not possible. Still, since the following steps use this module's outcome, their results may indicate the success of this effort. Nevertheless, it is worth discussing the challenges faced and how they were overcome.

4.3.3.1 2D Pose Estimation

Given the variability in camera distances in racket sports play videos, with players sometimes differing by more than 20 meters, there's a notable difference in perceived size. A top-down approach model was chosen to address these scale variation issues as the bottom-up ones are unfavorable to these conditions. RTMPose [35], detailed in the State-of-the-Art chapter, was selected as it is supported by MMPose and offers state-of-the-art performance for real-time multi-person pose estimation.

In order to utilize the RTMPose top-down model to infer poses, it is necessary to first detect the bounding boxes of each player. For this task, RTMDet [47] is used as it is a real-time detection model available in MMDetection recommended by RTMPose authors. Then, the pose inference step occurs and produces the keypoints associated with each bounding box in every frame. The keypoints are defined as COCO [43], where 17 keypoints, identified by their pixel coordinates, are detected for each human, as seen in Figure 4.7.

As mentioned in the State-of-the-Art chapter in Section 3.3, [34], a study on estimating player positions from high-angle Padel videos, considers that the best way to estimate a player's position is through pose estimation. The player position is estimated as the projection of the center of the player's hip on the ground, calculated as shown in Equation 3.2. Since COCO's pose definition

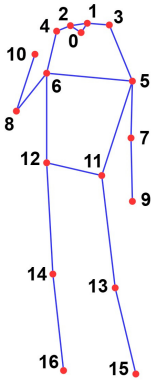


Figure 4.7: COCO's [43] definition of human pose.

does not include a single point for the hip but instead has two points at its limits, the position is calculated as:

$$p = (\frac{1}{2}(Hl_x + Hr_x), \frac{1}{2}(Fl_y + Fr_y)) \tag{4.2}$$

Here, Hl_x and Hr_x are the X coordinate (width) of each of the hip joints, and L_y and R_y are the Y coordinates (length) of the left and right ankle joint, respectively.

Also, since the camera calibration is available and COCO's pose definition identifies the ankle rather than the heel, an optimization can be applied. Instead of assuming that the point defined by Equation 4.2 is in contact with the ground, as in [34], it is assumed to be at a height λ, representing a small offset above the ground. This adjustment results in a more realistic position, helping to mitigate any inaccuracies due to the height offset, as demonstrated in Figure 3.10. Figure 4.8 illustrates an example of both pose and position estimation on a frame from the gathered dataset. In this figure, the blue dots and green lines represent the keypoints and keypoint connections of the pose estimation step, while the red dot represents the calculated position.

4.3.3.2 Pose Sorting

The previous 2D pose estimation step only evaluates frames individually without preserving continuity or identifying specific players, which is needed. To address this issue, a pose sorting step is implemented.

The outcome of the 2D pose estimation step includes the detection of players, with each frame containing the bounding box associated with each detected human and their respective poses. The challenge is that these detected humans are not tracked across frames and are treated as independent detections. To address this, a multi-object tracker (MOT) is used to associate the bounding boxes across frames to individual humans.

A study for Multi-Object Trackers was not within the scope of this dissertation, and therefore, they were not researched. However, based on the available dataset, OC-SORT [9] outperformed ByteTrack [93] as it held onto each human's identity better. It's important to mention that neither

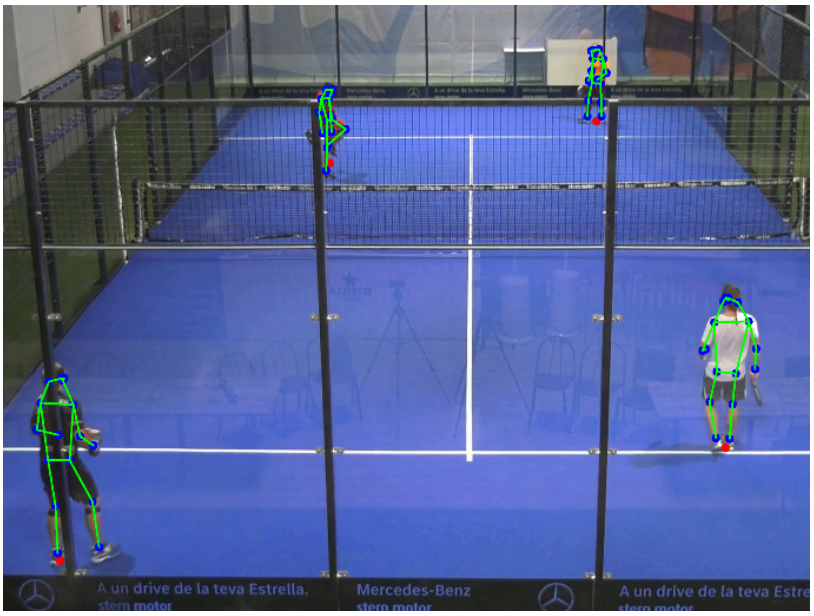


Figure 4.8: Example of pose and position estimation on a frame from the dataset gathered.

of these MOT implementations was carried out by the authors. The implementations are available at [51] for OC-SORT and [52] for ByteTrack.

Once the bounding boxes are associated, the position of each player in each frame is calculated as explained in the previous section. Subsequently, it's essential to identify the actual players and assign them to their respective teams. Two metrics are used to distinguish the players: total distance traveled and their level of presence on the court.

The first metric assumes that players will travel the most during the clip, while the second metric is based on the assumption that players will be on top of the court more consistently, especially in Padel, where players are typically inside the court. These metrics are normalized and summed, and the humans with the highest scores are identified as players.

To determine team affiliation, a simple sorting based on the average position of each player along the court's depth axis is performed, as the cameras in the dataset are all positioned on one side of the court. This sorting divides the players into two teams: those on the top and those on the bottom of the net.

Finally, the results are interpolated to avoid abrupt transitions and gaps, ensuring smooth and accurate tracking throughout the video.

4.3.3.3 3D Pose Estimation

Although it was planned to work only on 2D pose estimation in racket sports, an attempt was made to explore 3D pose estimation using a pose lifter model. However, due to time constraints and the complexity of the task, this aspect of the project was not completed. Implementing and evaluating the 3D pose estimation approach remains a promising area for further development to enhance

the accuracy and robustness of player movement analysis and improve the final 3D representation with human models.

4.3.4 Play Segmentation

The trajectory of a ball in the air can be deduced, or at least approximated (when considering air resistance), by knowing the ball's location at the start, the end, and the duration of its travel. Considering that the camera is calibrated, by having the 2D ball position on the image and the real coordinates of the plane the ball contacted, it is possible to calculate the exact 3D world coordinate of the ball's contact point. This contact point is important as it marks the end of the previous trajectory and the start of the next.

Therefore, two things are needed: the instance of a ball hit and the plane or object (such as a racket) it contacted. For this purpose, a study on play segmentation was conducted to explore how this information can be extracted from a single Padel or Tennis play video.

As stated in the State-of-the-Art chapter, MonoTrack's HitNet [44] was developed with the same objective. This module revolves around it. HitNet, as mentioned before, uses the ball position and, optionally, the court points or the players' poses from a set of 12 consecutive frames to evaluate whether there was a hit in the last 6 frames and, if so, where it occurred. It is worth noting that the dataset used [67] is based on clips of 30 FPS, while the ones from the available dataset range from 23.98 to 60 FPS, Therefore, the original HitNet model will first be evaluated on this new dataset, and if demanded, adjustments will be made.

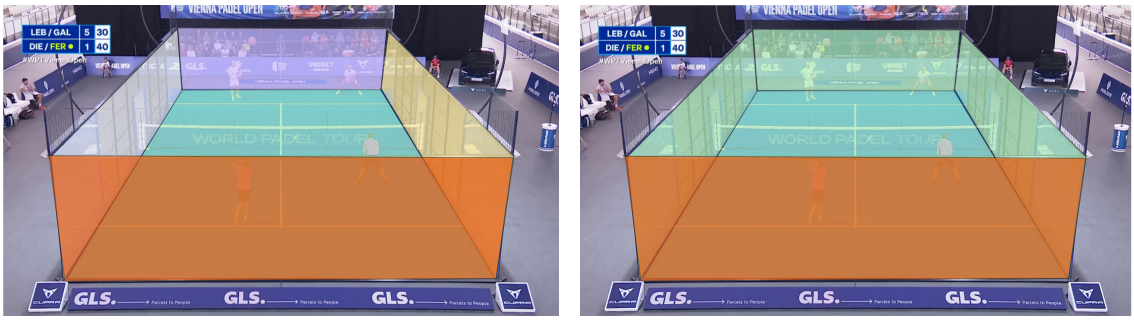
Fortunately, although not stated in the original work, MonoTrack's codebase is available publicly [45], so their HitNet development was used as the basis for the research undertaken.

This section offers a comprehensive explanation of the data preparation process, including insights into the model's architecture and its training procedure.

4.3.4.1 Data Preparation

As mentioned before, it is possible to calculate, or at least approximate, the trajectory of the ball between two points, as long as their positions and travel time are known. To adopt this strategy to solve the problem at hand, it is necessary to first retrieve the points at which the trajectory will then be extrapolated. These points are when the ball changes direction, or in other words, suffers a *hit-event*. These *hit-events* happen naturally during the course of a game and can be either a bounce on the ground or walls, a hit on the net, or contact with one of the players' rackets.

As explained in the Dataset section 4.3.1, each frame of every clip in the dataset is annotated to identify when the ball made contact and with what (e.g., player, ground, net). For Padel, the annotation changes slightly due to the presence of walls. Specifically, the ground and all the walls besides the one right in front of the camera are grouped as a single class.



(a) (b)

Figure 4.9: Aggregation of the initial five plane classes in (a) to the final two plane classes in (b).

In this process, and as demonstrated by Figure 4.9 planes that don't overlap in the camera's view can be grouped together under the same class. This is because, for any given line-of-sight, only one of these grouped planes is valid, as only one can be intersected. This decision was made to simplify classification and potentially improve the model performance.

In the Tennis case, the *net-hit* event was discarded as the original dataset [31] did not annotate it.

So, the classification classes for Tennis and Padel ended as follows:

- **Tennis**: No hit, ground hit, bottom player hit, and top player hit.
- **Padel**: No hit, close wall hit, ground/remaining walls hit, bottom team player hit, top team player hit, and net hit.

These annotations represent the data that the HitNet model will attempt to predict, and as such, they are used as the ground truth for any subsequent evaluation.

As for the input, it comprises the ball's image position, along with optional additional information such as the court corner points or the players' poses. For each frame, these values, along with the corresponding values for the subsequent frames (set as 12 frames in [44]), are stored in a feature vector. This feature vector consists of normalized 2D coordinates, where each coordinate of the pair (x, y) is scaled to the interval $[1, 2]$, and in case it is missing, it is set as $(0, 0)$. Additionally, to address the class imbalance between hit and no-hit events, the training and validation datasets are balanced to include an equal number of each type. This approach prevents the model from becoming biased towards predicting no hit in all frames, which would otherwise result in misleadingly high accuracy.

Following the original work, standard data augmentation techniques were used to improve model performance. These included data dropping, corruption, and rotation adjustments, with image reflection being the most frequent, applied to about 50% of the inputs.

4.3.4.2 Model

HitNet architecture was shown in the Figure 3.7 of the State-of-the-Art chapter. It is a simple model composed of:

- **Input layer**: Receives the feature vector from the preprocessed dataset, initializing the input shape for the model.
- **Reshape layer**: Adjusts the input shape to prepare it for sequential processing.
- **GRU layers**: Processes the sequential reshaped input data using a one-way Gated Recurrent Unit (GRU) [16] layer to capture temporal dependencies and patterns in the data.
- **Global max pooling layer**: Compresses the output from the GRU layers by selecting the maximum value from each feature map.
- **Fully connected layer**: Maps the reduced feature set to a four-dimensional output space for Tennis and six for Padel, corresponding to the model's target classes.
- **Softmax output layer**: Converts the output into a probability distribution across the target classes, facilitating multi-class classification.

The most suitable number of GRU layers for our problem may vary depending on the sport or different inputs (considering pose or not, for example). To address this variability, hyperparameter tuning with Keras Tuner [55] was employed with TensorFlow [1] to search for the most capable one for each configuration. In this process, multiple training sessions occur, changing the number of available GRU layers and their output dimensionality to check which configuration performs best, with the number of layers ranging between 1, 2, and 4, and the dimensionality varying from 16, 64, to 128. This training process can go up to 500 epochs, adopts cross-entropy as its loss metric, and leverages the Adam optimizer while keeping the learning rate at 0.01.

It is expected that after this tuning process, the observed performance will approach the full potential of this architecture. This allows for an informed decision on whether this approach is suitable for play segmentation. The results will provide valuable insights into what possible steps to take for future work.

4.3.5 3D Reconstruction

After analyzing the ball's position in the images and identifying when it bounces and on which surfaces, the 3D reconstruction step becomes feasible. This process aims to estimate the 3D location of the ball between consecutive ball hits, tracing its path through the air and across the play.

When the only force acting on a ball in the movement is gravity, it is possible to calculate the exact trajectory if the initial and final positions and travel time are known. However, in reality, two additional significant forces come into play: air resistance and, to a lesser extent, the Magnus force. These forces not only cannot be ignored as they visibly affect the ball's path, but they also turn the problem from a straightforward calculation into a constrained nonlinear optimization problem. To solve this, one must minimize the 3D reprojection error by adjusting the initial estimates of the ball's position and velocity, ensuring that the modeled path aligns closely with the observed data.

Although a similar problem has been addressed in [44], this work is distinct as it leverages the camera projection matrix to enhance its initial position estimations.

This section provides an overview of the camera calibration process and its utility, followed by a description of the physics model and the various approaches for handling it. As one of the main objectives of this dissertation is to provide a prototype for an automatic highlight system for the targeted racket sports, a proof of concept 3D representation is also presented to demonstrate this capability.

4.3.5.1 Camera Calibration

As Section 4.3.1.1 mentioned, selected court image positions were annotated for each camera. At least six points were detected for each, four of those the boundary court corners plus the two tip points of the net poles. With them, and having the 3D real positions for each corresponding point, the camera projection matrix can be calculated using the Direct Linear Transform (DLT) [2].

The primary use of the calculated projection matrix is that it enables the projection of 3D points into their corresponding positions in the image space. Additionally, it is used to determine the line-of-sight, which contains all the 3D real points that could correspond to a given image space position. Since the calculation of the camera projection matrix by DLT does not allow for its decomposition into the camera intrinsic matrix, several steps were taken instead to achieve this:

- **Calculate the pseudo-inverse projection matrix**: Since the projection matrix is 3x4 and not square, the (Moore-Penrose) pseudo-inverse [5] is calculated. With it, for a given 2D image position, one of the infinite 3D real positions that could correspond to it is returned.
- **Get alternative points for the court points annotated**: Identify, for each annotated point, one additional 3D real position using the pseudo-inverse projection matrix.
- **Calculate the epipolar lines for each of the annotated points**: Utilize the annotated 3D real position and the newly calculated point to determine the epipolar line for each annotated court point.
- **Retrieve the camera's real position**: Cross all the epipolar lines and determine the optimal position corresponding to the camera's location.

Then, to determine the epipolar line corresponding to an arbitrary image point, it is sufficient to calculate one corresponding 3D real position for that point, using the pseudo-inverse matrix, and associate it with the camera position.

When coupled with information from the play segmentation module, such as the moment a ball bounces and the plane it contacts, this process enables the calculation of the ball's actual position. This position is determined by the intersection of the epipolar line and the identified contact plane.

4.3.5.2 Physics-based Trajectory Estimation

When the process reaches this last phase of the pipeline, the cameras are calibrated, the players' positions are known, and the instance at which the ball changes directions and which entity it enters contact with are known. To address the challenge of accurately estimating the ball's ballistic trajectory, an evaluation was made to assess the role of these factors, and thus four approaches were considered:

Only Gravitational Force Considered

To estimate a ballistic trajectory influenced solely by gravitational force, two key pieces of information are needed: the initial position and the initial velocity. However, the initial velocity is not directly measurable with the system developed. To determine it, the displacement of the ball, calculated as the difference between its final and initial positions, and the travel time are utilized.

The accuracy of the initial and final positions used for this purpose is high when representing a ball bounce on the ground or walls (in Padel's case). This is due to the intersection of the epipolar line at that image point with the identified plane. However, pinpointing the exact position where the ball changes direction due to a player's touch is challenging because there aren't enough constraints to determine a single point accurately.

Therefore, for this approach, the method to determine the initial and final positions of the trajectory assumes that the player's hits were consistently at a set height h.

Having said this, the initial horizontal and vertical velocity (v_{0xy} and v_{0z}) considering gravity (g) can be determined by:

$$v_{0xy} = \frac{d_{xy}}{\Delta t} \tag{4.3}$$

$$v_{0z} = \frac{d_z}{\Delta t} + \frac{1}{2} g \Delta t \tag{4.4}$$

In these expressions, d_{xy} and d_z represent the horizontal and vertical displacement, and Δt represents the time interval. By combining these horizontal and vertical components, the initial velocity vector ($\vec{v_0}$) can be determined:

$$\vec{v_0} = v_{0xy} \hat{d}_{xy} + v_{0z} \hat{z} \tag{4.5}$$

where $\hat{d}_{xy}$ is the unit vector in the horizontal direction and $\hat{z}$ is the unit vector in the vertical direction.

Then, combining this calculated initial velocity ($\vec{v_0}$) with the initial position ($\vec{r_0}$), the position of the ball at any time t can be estimated, allowing for the reconstruction of the desired trajectory. This is determined using the equations:

$$r_x(t) = r_{0x} + v_{0x}t$$

$$r_y(t) = r_{0y} + v_{0y}t$$

$$r_z(t) = r_{0z} + v_{0z}t - \frac{1}{2}gt^2$$

where r_{0x}, r_{0y}, and r_{0z} are the initial positions in the x, y, and z directions, respectively, and v_{0x}, v_{0y}, and v_{0z} are the components of the initial velocity vector ($\vec{v_0}$).

Included Air Resistance

As mentioned before, two other forces may act on the ball, the air resistance and the Magnus force. For this work, the effects of spin, the Magnus force, were not considered significant compared to air resistance and thus were not included in the analysis. Considering this, the ball's 3D trajectory, $r(t)$, can be approximated by a particle under drag [17, 44]:

$$\frac{\mathrm{d}^2 r}{\mathrm{d}t^2} = g - C_d ||r||_2 r, \tag{4.6}$$

$$\text{subject to} \quad r(0) = r_0, \quad \frac{\mathrm{d}r}{\mathrm{d}t}(0) = v_0 \tag{4.7}$$

where $r_0 = (x_0, y_0, z_0)^\top$ is the initial position, v_0 is the initial velocity, and C_d is the drag coefficient. The gravitational acceleration g is constant, while the drag coefficient C_d may vary slightly between shots. Both Euler's and Runge-Kutta's methods can be employed numerically to solve these equations.

In this scenario, this trajectory's initial position and velocity are also required, so the same height guess made for the player touches on the previous approach was done. However, objectively calculating the initial velocity becomes challenging when considering air resistance. This necessitates transforming the problem into a constrained nonlinear optimization task. Assuming a fixed direction angle (or azimuth angle) for the initial velocity, successive elevation angles and intensities are iteratively guessed to minimize the difference between the final position calculated using the air drag model and the predetermined target final position obtained from the rest of the pipeline.

To solve this minimization problem, the SciPy [77] library's 'minimize' function was utilized, using the Sequential Least Squares Programming (SLSQP) [39] method.

Reprojection Error as Loss Function

To better assess the quality of the calculated 3D trajectory, $r(t) \in \mathbb{R}^3$, the reprojection error can be used. By projecting $r(t)$ into the image space to obtain the 2D trajectory estimates $\hat{r}(t) \in \mathbb{R}^2$ the reprojection error can be determined as:

$$\mathscr{L}_r = \|\hat{r}(t) - \tilde{r}(t)\|_2^2 \tag{4.8}$$

where $\tilde{r}(t)$ is the previously tracked 2D coordinates of the detected ball. Using this metric to aid in the estimation process is also possible. Employing it as the scalar function to minimize helps to remove the fixed height assumption for the player's hit. The reprojection error loss function guides the initial position and velocity guesses to naturally converge toward realistic values. However, as observed in the results, these values may not always be ideal.

To simplify the problem, the initial position can be expressed as a function $f(h)$, where h represents the height at which the player touches the ball. As the initial guess, the height chosen is 1.5 meters. This height h is then constrained for subsequent guesses to be within 0 and 2.8 meters in order to avoid unrealistic scenarios.

Players' Positions Considered

To further aid the estimation process, the players' positions were considered, providing valuable information for the reconstruction. A player's touch can only occur near the player, so these positions constrain the possible initial positions of the ball. This ensures the initial position guess is more accurate by considering the player's proximity to the ball.

For this effect, instead of guessing the initial position solely based on its height through $f(h)$, the initial position is determined by $g(d)$, where d represents the longitudinal axis difference between the initial position of the trajectory and the closest player's position. Initially, d is set to 0, indicating that the ball was hit from the same longitudinal position as the player. During the optimization task, d is adjusted to minimize the error between valid guesses. In Tennis, d is constrained within the range of $(-0.6, 1.0)$ meters, while in Padel, d is constrained within $(-1.0, 1.0)$ meters, considering that players can hit the ball backward due to the presence of walls.

4.3.5.3 3D Representation

As the final task of the proposed solution, a basic 3D representation was created using the information gathered from previous tasks, including the positions of the ball and players at each instant. This proof of concept demonstrates the capabilities of the prototype developed in this dissertation for analyzing monocular racket sports. The representation includes objects for the ball and players that dynamically update at each frame based on their respective positions.

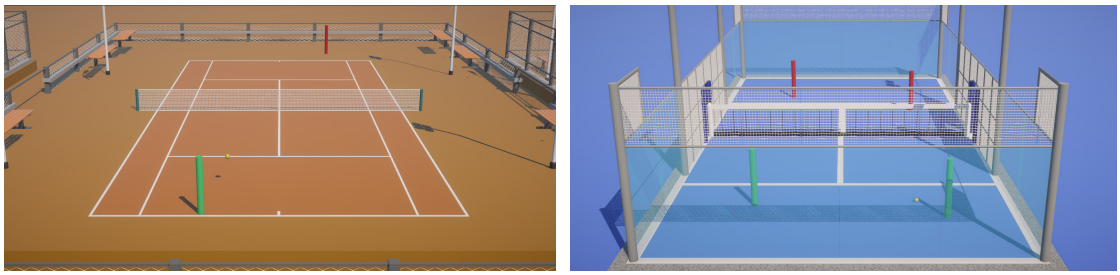


Figure 4.10: 3D representation proof of concept for Tennis and Padel plays.

Figure 4.10 shows a sample of the 3D reconstruction step created using only the CSV files containing the output of the proposed pipeline, the Unity [73] game engine, and public assets for the Tennis[2] and Padel[3] court and the ball[4].

A natural next step for this representation would be to add 3D poses for each player, which would enhance the realism and depth of the simulation. However, this capability will only be feasible once accurate 3D pose estimation for all the players involved becomes available.

4.4 Summary

The solution proposed in this chapter emerged from extensive research conducted during the dissertation's preparation phase, aiming to address the problem identified and defined effectively. The proposed work was addressed and developed successfully, providing a sound solution for the problem at hand.

Initially, publicly available datasets were concatenated and annotated to encompass the ball's image position and *hit-event* across each frame, establishing comprehensive datasets for Tennis and Padel tailored for the subsequent tasks.

Next, three ball detection models were trained, tested, and evaluated for each sport to understand which presented the best performance for each scenario. For Padel, transfer learning was utilized to adapt the already trained Tennis models to this similar sport, serving as an efficient alternative to the time-consuming task of training a model from scratch.

For 2D pose and position estimation, RTMPose [35] was employed, utilizing OCSort [9] to accurately track and organize the players' poses. Following this, HitNet [44] was implemented, trained, and evaluated to accurately detect ball *hit-events* and identify the involved entities, using the ball's detection, court corners, and players' positions. Hyperparameter tuning was employed to identify the best architecture for each scenario.

Four distinct methods were proposed to reconstruct the 3D trajectory of the ball between the *hit-events* identified in the previous step, ranging from a simple approach that only considers gravity to a more complex one that uses the reprojection error to aid in trajectory estimation.

[2] https://www.turbosquid.com/3d-models/tennis-court-a3-2136003
[3] https://sketchfab.com/3d-models/padel-court-or-paddle-court-ef977f0912af4297bd842ab7bb82dde1
[4] https://sketchfab.com/3d-models/tennis-ball-8f1cbc19dc414ad89097eabe063ada88

These efforts culminated in a dynamic 3D representation of the final scene, created using Unity and publicly available assets, which integrated the real position of the ball and the players' position retrieved from the pipeline.

Chapter 5

Tests and Results

As mentioned in the previous chapter, the proposed solution's effectiveness relies on three key components. The performance of these components was analyzed to evaluate whether the approach was correct, compare different methods, and ultimately determine if the solution meets the work's objectives.

With that intent, this section is divided into four subsections. The first three present the results from the tests undertaken for each of the pipeline's tasks, as well as an analysis of these results. The last section provides a final summary containing the conclusions from the discussion of the results and proposes future work for each block individually.

The results were collected from a machine equipped with an Intel® Core™ i5-9300H quad-core processor with a base frequency of 2.40 GHz. The system featured an NVIDIA® GeForce® GTX 1650 mobile graphics card, 16 GB of DDR4 memory clocked at 2666 MHz, and a 256 GB SSD. The operating system used was Linux Mint 21.3.

5.1 Ball Tracker

As stated in the previous chapter, three state-of-the-art ball detection models—TrackNetV2 [67], MonoTrack [44], and WASB [69]—were selected for testing and evaluation to determine the best candidate for each of the considered sports.

The performance of the ball detection models was assessed using four key metrics: F1 Score, Accuracy, Average Precision (AP), and Frames Per Second (FPS). These metrics were employed to evaluate and compare the models' ability to accurately infer the coordinates of the ball's center across a sequence of images. The distance between a predicted ball position and the ground truth for each frame is calculated to classify the prediction as true positive, true negative, false positive, or false negative. Using the previously mentioned τ threshold of 10 pixels, this classification is determined based on whether the distance falls within the acceptable range. These metrics were selected because they are commonly used in similar research, ensuring that the results are comparable with other works in the field. The F1 Score and Accuracy provide insights into the models'

overall detection capabilities, Average Precision reflects the models' precision-recall performance, and FPS measures how many frames the model can process per second.

Table 5.1: Benchmark results of pre-trained models on the sports datasets.

		Tennis				Padel			
Model	# param	F1	Acc.	AP	FPS	F1	Acc.	AP	FPS
TrackNetV2 [67]	11.3*M*	89.2	81.8	80.4	26.3	57.2	50.6	37.6	28.1
MonoTrack [44]	2.9*M*	92.3	86.6	87.7	**53.0**	68.5	60.0	51.6	**58.2**
WASB[69]	1.5*M*	**93.2**	**88.0**	**88.5**	38.4	**72.5**	**63.3**	**57.2**	42.0

Table 5.1 shows how the pre-trained models fared against both of the validation subsets of the Tennis and Padel datasets. While the Tennis results were consistent with expectations, as seen in [69], Padel's results indicate a notable decrease in performance, with an average decrease in accuracy of 32.3%.

Table 5.2: Benchmark results of tuned models on the sports datasets.

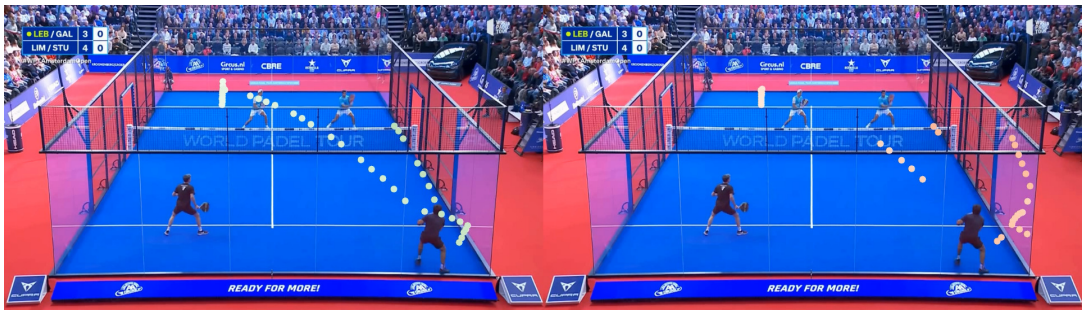
		Tennis				Padel			
Model	# param	F1	Acc.	AP	FPS	F1	Acc.	AP	FPS
TrackNetV2 [67]	11.3*M*	**91.6**	**85.6**	**87.5**	21.7	81.8	73.2	73.6	22.8
MonoTrack [44]	2.9*M*	59.6	48.2	43.1	**48.1**	81.2	72.9	71.2	**57.0**
WASB[69]	1.5*M*	75.2	63.9	60.5	37.5	**84.0**	**75.9**	**76.4**	43.0

As seen in Table 5.2, by exposing the pre-trained models to Padel-specific scenarios during the transfer learning phase, these models improved on average 21.7% and 33.8% on accuracy and average precision respectively, comparing with the previous results.

Regarding inference performance, measured in Frames Per Second (FPS), all models demonstrated real-time processing capabilities. However, a distinct performance hierarchy was observed, with WASB outperforming MonoTrack and MonoTrack outperforming TrackNetV2.

Discussion

Table 5.1 shows a clear, unexpected performance drop by the pre-trained models for Padel. It was anticipated that differences in camera angles and the increased occlusions due to more players and opaque rackets would negatively impact performance, but not to this degree. However, after analyzing the results, another circumstance was found in Padel that does not exist in Tennis: the ball detected on the wall's reflection had a confidence score that was even bigger than the real one.



(a)

(b)

Figure 5.1: Example of poor ball detection results due to walls' reflection.

This situation can be seen in Figures 5.1a and 5.1b, where the left green dots represent the ground truth ball positions, and the right red dots the model's ball detection. This was the main reason for the decision to fine-tune the current pre-trained Tennis weights.

Regarding detection performance for the fine-tuned models that are present in Table 5.2, WASB still presented the best results in all metrics, and the performance gaps of TrackNetV2 and MonoTrack shrunk into similar values.

An unanticipated outcome was that, even though both MonoTrack's and WASB's detection performance lowered for Tennis' plays, TrackNetV2 did not, having a non-negligible increase in all detection metrics. One way to interpret this is that TrackNetV2's much larger number of parameters allows for a greater generalization capacity. This is especially true compared to MonoTrack since it is a variant of TrackNetV2, but one of its main differences is the removal of some convolution layers. This difference could also explain why TrackNetV2 exhibited the largest performance improvement for Padel when comparing pre-trained and fine-tuned models.

Due to the results presented in this evaluation, the WASB model was selected as the preferred choice for both Tennis, utilizing the pre-trained weights, and Padel, using the fine-tuned weights, for the subsequent tasks in this dissertation.

5.2 Play segmentation

After the hyperparameter tuning process, which searches for the most suitable model configuration, an evaluation step occurs. Recall, accuracy, precision, and F1 score metrics are collected and

compared to assess this module's performance. Similarly to HitNet's work [44], the ground truth can be represented as the set composed of the hit-event pairs $G = \{(t_i, e_i)\}$ indicating that entity e_i hit the ball on frame t_i, and the model predicts $\hat{G} = \{(\hat{t}_i, \hat{e}_i)\}$, so the metrics were calculated as:

$$\text{acc.} = \frac{|G \cap \hat{G}|}{|G \cup \hat{G}|}, \quad \text{recall} = \frac{|G \cap \hat{G}|}{|G|}, \quad \text{prec.} = \frac{|G \cap \hat{G}|}{|\hat{G}|} \tag{5.1}$$

A random forest-based classifier (RF) acts as a baseline, and ablation studies are also done to analyze the impact of each possible input on the model's final predictions. These studies involve comparing three versions of the model: one that uses only the ball image position as input, another that includes the court corners, and a third that additionally considers the players' poses.

Table 5.3: Comparison of HitNet on Tennis over baseline and ablation study.

Model	Recall	Acc.	Pre.	F1
RF	56.6%	45.4%	49.8%	0.625
RF (Court)	45.7%	40.0%	76.0%	0.571
RF (Court+Pose)	20.5%	18.8%	69.6%	0.316
HitNet	**72.6%**	**66.2%**	88.4%	**0.797**
HitNet (Court)	70.3%	62.2%	84.3%	0.767
HitNet (Court+Pose)	70.2%	64.7%	**89.2%**	0.786

Table 5.3 shows the HitNet results collected for Tennis plays. Unlike HitNet's original work, no performance improvements were observed when incorporating court corners or players' poses; in fact, this pattern was more evident in the RF baseline model. However, as shown in the table, each respective HitNet configuration clearly surpassed the baseline in all metrics, with an improvement of 28.3% in recall and 45.8% in accuracy for each of their best configurations.

Table 5.4: Comparison of HitNet on Padel plays over baseline and ablation study.

Model	Recall	Acc.	Pre.	F1
RF	15.0%	12.0%	37.1%	0.214
RF (Court)	7.7%	6.83%	36.9%	0.128
RF (Court+Pose)	1.2%	1.18%	**57.1%**	0.023
HitNet	**40.2%**	**27.9%**	47.8%	**0.437**
HitNet (Court)	3.60%	3.45%	50.0%	0.067
HitNet (Court+Pose)	4.60%	3.53%	13.0%	0.068

In Table 5.4, HitNet's inference results for Padel plays are presented. Similar to the trend observed in Tennis plays, no performance gains were seen when adding court points or players' poses as inputs. This was even more pronounced in Padel, with recall and accuracy dropping by 91.0% and 87.3%, respectively, when incorporating these additional inputs.

Compared to the Tennis results, even the best HitNet configuration for Padel, which uses only the ball's coordinates, achieves only 57.8% of the accuracy and 45.9% of the precision of the corresponding Tennis model.

During the hyperparameter tuning phase, the most suitable configuration for each sport was determined by varying the GRU layers and the output dimensionality of each layer. For Tennis, the optimal configuration was found to be 4 GRU layers, each with an output dimensionality of 16, resulting in 16,542 trainable parameters, while for Padel, the best setup consisted of 2 GRU layers, each with an output dimensionality of 16, resulting in 6,918 trainable parameters. This demonstrates a clear adaptation of the network architecture to the specific demands of each sport.

The best models for each sport were later evaluated using the chosen ball tracker from the previous task instead of the ground-truth annotations. These results are available in the Table 5.5.

Table 5.5: Chosen models configurations.

	Model	Recall	Acc.	Pre.	F1
Tennis	HitNet	**72.6%**	**66.2%**	**88.4%**	**0.797**
	HitNet (end-to-end)	33.5%	31.7%	85.4%	0.481
Padel	HitNet	**40.2%**	**27.9%**	**47.8%**	**0.437**
	HitNet (end-to-end)	31.4%	22.9%	45.7%	0.372

Although the ball tracking task results were successful, there was a significant drop in segmentation performance for both Tennis and Padel when using the ball tracker, with Tennis experiencing a more substantial decline. Notably, while recall and accuracy decreased in both sports, precision remained stable.

Regarding processing performance, the current models comfortably exceed the real-time requirement, achieving processing speeds of several thousand frames per second.

Discussion

Despite efforts to adapt the original HitNet for both sports, the results did not meet the work's objectives. Two main reasons contributed to this outcome, offering valuable insights for future attempts to address this problem.

Firstly, performance did not improve when incorporating court corners or players' poses. This is likely due to the nature of the sports. The original HitNet was designed for Badminton, where the task is to detect the instant a player touches the shuttlecock and identify the player, making

the poses highly relevant. However, in Tennis and Padel, the ball can touch the ground, net, and in Padel, the walls. This results in less dependency on player poses and more variability, causing the additional inputs to hinder rather than help. This can be seen in the performance of the RF baseline model, as the performance degraded too, when adding the players' poses.

Secondly, the significant difference in performance between the envisaged sports could be attributed to the higher input and output dimensionality for Padel compared to Tennis, leading to worse performance. Padel, typically played by four people instead of two, involves double the number of poses. For the configuration that includes players' poses in Padel, the input dimensionality is nearly double:

$$12 \times ((1+4+17 \times 4) \times 2) = 1,752$$

compared to Tennis:

$$12 \times ((1+4+17 \times 2) \times 2) = 936$$

This results in 1.87 times the number of input values. Coupled with the fact that Padel involves classifying two additional classes, the increased dimensionality does not aid performance, especially given the smaller dataset available for Padel.

These input dimensionality values can be obtained by:

$$N_{input} = f \times ((1+4+17 \times p) \times 2) \tag{5.2}$$

Here, f represents the number of consecutive frames, and p represents the number of players. The term $(1+4+17 \times p)$ accounts for the positions provided per frame (1 for ball position, 4 for court corners, and 17 for each pose keypoint), multiplied by the number of dimensions (2) and frames f, resulting in the total number of input values N_{input}.

5.3 3D Reconstruction

The primary objective of the 3D Reconstruction task is to estimate the 3D location of the ball. This section presents and discusses the results of the tests conducted for the 3D reconstruction task, aiming to determine the most effective approach and evaluate its suitability for the problem at hand.

As stated in Section 4.3.5 of the previous chapter, four approaches were considered to estimate the trajectory between two points: *gravitational force only* (G.F.O.), *air resistance included* (A.R.I.), *reprojection error as loss function* (R.E.L.F.), and *players' positions considered* (P.P.C.). The first approach, *gravitational force only*, considers gravity as the only force acting on the ball, requiring simple and straightforward calculations. The second approach, *air resistance included*, includes air resistance, making the problem a more complex nonlinear optimization task. The

third approach, *reprojection error as loss function*, uses the reprojection error as the loss function to aid in estimating the trajectory of the ball through the air, not necessitating an estimation of the height at which player-ball contacts occur like the previous two approaches. The final approach, *players' positions considered*, leverages the players' positions, calculated by the pose estimation module of the pipeline, to constrain the possible starting points for ball trajectories that originate from player-ball touches. In the case of the first two approaches, the assumed height h for all ball-player contacts was 1.6 meters.

In order to test and compare these approaches, two key metrics were used: reprojection error and absolute error. The reprojection error includes both mean (MRE) and median (MedRE) values, representing the distance in pixels between the projected 3D position and its corresponding 2D position in the image. On the other hand, the absolute error comprises mean (MAE) and median (MedAE) values, indicating the real distance between the estimated 3D position and the ground-truth position. Given that only the PadelVic portion of the dataset provides multi-view data with ground truth, MAE and MedAE are specifically used for these clips. It is also worth noting that the ground-truth data itself may contain inaccuracies due to potential errors in data annotation and the suboptimal synchronization of the cameras, which was approximate rather than per-frame.

Additionally, to assess the prototype's practical performance, the physics-based estimation methods were evaluated using the *hit-events* identified by the preceding play segmentation task. Another evaluation included the integration of ball detection results from the ball tracker task. During these assessments, errors were computed only within time intervals feasible for estimation, specifically between the first and last *hit-events* detected by the HitNet model.

Table 5.6: The physics-estimation methods performance for Tennis plays.

Estimation Approach	MRE[px]	MedRE[px]	FPS
Gravitational Force Only	31.8	22.3	$\mathbf{1.15 \times 10^5}$
Air Resistance Included	29.0	19.2	89.0
Reprojection Error as Loss Function	**21.4**	**5.82**	44.5
Players' Positions Considered	31.2	9.16	17.8
R.E.L.F. (HitNet)	165	81.4	18.7
R.E.L.F. (Ball + HitNet)	174	96.6	18.1

The results from the physics estimation approaches for Tennis plays are presented in Table 5.6. It is evident that the approach using the reprojection error as the optimization problem's loss function yielded more favorable outcomes. Consequently, this approach was selected for ablation studies, but it showed an unacceptable median reprojection error of 96.6 pixels, representing a 1054% increase compared to the version utilizing annotations. Notably, the median reprojection error averaged 49.6% of the mean reprojection error.

Table 5.7: The physics-estimation methods performance for Padel plays.

	Padel Public Plays			PadelVic		
Est. App.	MRE[px]	MedRE[px]	FPS	MRE[px]	MedRE[px]	FPS
G.F.O.	27.7	23.5	$\mathbf{1.42 \times 10^5}$	23.0	17.4	$\mathbf{9.01 \times 10^4}$
A.R.I.	26.7	25.4	72.2	21.5	17.4	175
R.E.L.F.	**10.6**	**7.55**	35.3	**8.10**	**3.76**	43.8
P.P.C.	13.1	9.14	38.6	10.1	4.50	69.7
R.E.L.F. (HitNet)	15.3	12.5	34.5	10.6	8.96	49.2
R.E.L.F. (Ball + HitNet)	16.8	12.6	29.8	12.2	9.46	53.4

The results for Padel shown in Table 5.7 exhibit a similar trend to the Tennis results regarding the performance of the four different approaches, albeit with an overall better performance. Additionally, the difference in error between the ablation studies decreased significantly, reducing to 277%.

Table 5.8: The physics-estimation methods absolute error for Padel plays.

Estimation Approach	MAE[m]	MedAE[m]
Gravitational Force Only	1.05	1.39
Air Resistance Included	1.04	1.37
Reprojection Error as Loss Function	**0.928**	1.29
Players' Positions Considered	0.931	**0.990**
P.P.C. (HitNet)	2.58	3.40
P.P.C. (Ball + HitNet)	1.78	2.75

Given the availability of ground truth data for PadelVic's ball positions, additional evaluations were conducted to assess the proximity of the estimated ball positions to the ground truth, which is a more appropriate metric for the problem than reprojection error alone. Among the evaluated approaches, the one incorporating players' positions to aid the optimization process yielded the most favorable results, achieving mean and median absolute errors of less than a meter.

Discussion

While the estimation approaches showed satisfactory performance on their own, integrating them with the ball tracker and play segmentation HitNet to simulate real-world scenarios led to a median absolute error of 2.75 meters (Table 5.8). This indicates that the combined pipeline did not achieve optimal results compared to when the estimation approaches were used independently. It's important to highlight that this metric was specifically calculated within the play segment identified by the play segmentation task from the first to the last detected *hit-event*. This suggests that its actual performance in real-world scenarios may be poorer.

Based on the reprojection error metric used, it's evident that the approach using it as the optimization loss function yielded superior results compared to the previous G.F.O. and A.R.I. approaches. Despite this, this new approach's absolute error remains smaller, indicating its overall superiority. Additionally, incorporating players' positions led to a significant improvement in median absolute error, although this improvement wasn't reflected in MRE and MedRE metrics, emphasizing the potential limitations of relying solely on reprojection error.

Regarding an evaluation in terms of Frames per Second (FPS), all approaches demonstrated real-time capability. Notably, in Padel, the approach that considered players' positions exhibited superior processing speed compared to R.E.L.F, primarily due to its quicker convergence enabled by leveraging players' positions as a reliable indicator of the ball's contact point with the racket. However, this distinction in processing speed between P.P.C and R.E.L.F could not be observed for Tennis. Due to its reduced absolute error and superior processing speed, the P.P.C. approach was considered the most suitable estimation method for the problem.

Besides the variations in sports rules and dataset quality, the performance disparity between Tennis and Padel could be attributed to the camera calibration matrix's quality. In Tennis, the calibration process involves fewer calibration points, specifically capturing data from only two heights: ground level ($h = 0m$) and the height of the net poles ($h = 1.07m$). This limited calibration setup may contribute to more frequent inaccuracies in the 3D projections within the image space.

5.4 Summary

As for the ball tracker, its results demonstrated the success of the transfer learning step and concluded that WASB was the preferred model for both Tennis and Padel for this dissertation's objectives and, thus, was used in the subsequent modules. Despite the significant improvements observed during the fine-tuning phase, the performance still lagged behind the results obtained on the original Tennis dataset. Therefore, a full training of WASB from the ground up using solely the Padel dataset is proposed as future work. Additionally, other works like TrackNetV3 [13] have been developed during the course of this dissertation, and it would be valuable to compare the performance of these models in future studies.

Regardless of the efforts to adapt HitNet to the considered racket sports, its results still fell short of the desired performance values, emphasizing the necessity for visual information to handle this task. Future work proposes developing a computer vision AI model that diverges from the current HitNet, which relies solely on frame-extracted data. Instead, this model should leverage square image crops centered on the ball to directly analyze contextual information for detecting ball contacts. This approach is inspired by the principles of context-aware ball localization discussed in Section 3.2.4.3 of the State-of-the-Art chapter. By employing this localized image analysis approach, the AI model aims to enhance its ability to differentiate various types of *hit-events*, particularly in Padel, without overwhelming the system with excessive input values.

The 3D reconstruction step, when decoupled from the rest of the pipeline, yielded satisfactory results for this dissertation's work. However, it remains an ongoing challenge to achieve perfect

reconstruction. To address this gap, future work proposes estimation approaches that incorporate the Magnus Force resulting from the spin of the ball. Additionally, if 3D human pose estimation is possible, the approach should integrate information about the hand and, if available, the racket to assist with the minimization problem. Furthermore, exploring the use of a multi-view dataset or synthetic trajectories during the testing and evaluation phases would aid in further refining and validating these approaches.

Chapter 6

Conclusions

In this dissertation, research, planning, and development were conducted to create a 3D reconstruction prototype designed to serve as a replay and highlight tool for single-view clips of Tennis and Padel. With this intent, five major tasks were implemented: ball tracking, pose estimation, play segmentation, physics-based trajectory estimation, and 3D representation.

For ball tracking, the WASB detector [69] was determined to be the most effective for both Tennis and Padel. Additionally, transfer learning proved to be a suitable technique for adapting a model to sports with similar characteristics, including ball behavior, size, and color. In the pose estimation phase, a comprehensive method was implemented to accurately capture both the 2D poses and positions of the players. For play segmentation, MonoTrack's HitNet was modified and tailored to suit the specific requirements of Tennis and Padel. The 3D reconstruction phase introduced and evaluated four different trajectory estimation strategies, utilizing ablation studies to compare and analyze their performance within the pipeline. Lastly, a proof of concept for the 3D representation step was developed, demonstrating how the accumulated data throughout the pipeline can be effectively utilized to create a functional highlight and replay tool.

After evaluating the results from each task, it became evident that the overall objective was not fully met. The sequential nature of the pipeline's tasks meant that errors from earlier stages propagated and compounded throughout the pipeline, significantly impacting the final outcome. This issue was primarily due to the underperformance of the play segmentation module. Ultimately, it was considered that the approach taken, even though approved for other sports, was not adequate for the envisaged sports. Despite these challenges, the work conducted offers valuable insights into addressing similar problems in the future, particularly for the play segmentation task and the trajectory estimation step. As far as we found, this dissertation represents the first attempt at Tennis and Padel play segmentation, further contributing to the field. Additionally, the use of line-of-sight calculations to determine the ball's location at the time of contact and to assist with the minimization problem was the first of its kind according to the literature review.

Apart from the future work proposals outlined in the summary of the Tests and Results Section, which addressed each module individually, several recommendations are made to tackle the problem as a whole. Improvements should start with the dataset. These include ensuring more

rigorous synchronization of camera views to obtain more accurate ground truth data and gathering a multi-view Tennis dataset to assess the pipeline's practical performance for that sport. Additionally, using higher-quality cameras with fast enough shutter speeds to reduce motion blur and capture different angles, such as the zenithal angle, to minimize occlusions is recommended.

Increasing the number of data points could also enable the use of other approaches, such as reinforcement learning, which were previously discarded due to their extensive data requirements. Although a reinforcement learning model designed to extract 3D features from single or consecutive frames might not provide highly reliable results due to a lack of sport-specific understanding, it could still offer a plausible 3D representation of a Tennis' or Padel's play that current methods may not achieve.

Furthermore, expanding from a monocular to a multicamera approach is suggested to enhance the solution's robustness and improve the final representation. A multicamera setup would simplify the pipeline, making it less error-prone, and deliver better quality results, particularly in 3D human pose estimation. This approach would be easily scalable for the number of cameras and involve a study to determine the optimal number of cameras needed to achieve a satisfactory result.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] Y. I Abdel-Aziz, H. M. Karara, and Michael Hauck. Direct Linear Transformation from Comparator Coordinates into Object Space Coordinates in Close-Range Photogrammetry*. *Photogrammetric Engineering & Remote Sensing*, 81(2):103–107, February 2015.

[3] A. Aksay, V. Kitanovski, Karthikeyan Vaiapury, E. Onasoglou, J. D. Agapito, P. Daras, and E. Izquierdo. Robust 3 D Tracking in Tennis Videos. 2010.

[4] Yan Baodong. Hawkeye technology using tennis match. 2014.

[5] J. C. A. Barata and M. S. Hussein. The Moore-Penrose Pseudoinverse. A Tutorial Review of the Theory. *Brazilian Journal of Physics*, 42(1-2):146–165, April 2012. arXiv:1110.6882 [math-ph].

[6] Miniar Ben Gamra and Moulay A. Akhloufi. A review of deep learning techniques for 2D and 3D human pose estimation. *Image and Vision Computing*, 114:104282, October 2021.

[7] Sydney Boyo. How Sony's Hawk-Eye electronic line-calling system transformed the U.S. Open, September 2023.

[8] Marcello Davide Caio, Gabriel Van Zandycke, and Christophe De Vleeschouwer. Context-Aware 3D Object Localization from Single Calibrated Images: A Study of Basketballs. In *Proceedings of the 6th International Workshop on Multimedia Content Analysis in Sports*, pages 49–54, October 2023. arXiv:2309.03640 [cs, eess].

[9] Jinkun Cao, Jiangmiao Pang, Xinshuo Weng, Rawal Khirodkar, and Kris Kitani. Observation-Centric SORT: Rethinking SORT for Robust Multi-Object Tracking, March 2023. arXiv:2203.14360 [cs].

[10] L. Carrasco, S. Romero, B. Sañudo, and M. de Hoyo. Game analysis and energy requirements of paddle tennis competition. *Science & Sports*, 26(6):338–344, December 2011.

[11] Max Chang. Tracknet-badminton-tracking-tensorflow2. https://github.com/Chang-Chia-Chi/TrackNet-Badminton-Tracking-tensorflow2, 2021.

[12] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open MMLab Detection Toolbox and Benchmark, June 2019. arXiv:1906.07155 [cs, eess].

[13] Yu-Jou Chen and Yu-Shuen Wang. TrackNetV3: Enhancing ShuttleCock Tracking with Augmentations and Trajectory Rectification. In *Proceedings of the 5th ACM International Conference on Multimedia in Asia*, MMAsia '23, pages 1–7, New York, NY, USA, January 2024. Association for Computing Machinery.

[14] Yucheng Chen, Yingli Tian, and Mingyi He. Monocular Human Pose Estimation: A Survey of Deep Learning-based Methods. *Computer Vision and Image Understanding*, 192:102897, March 2020. arXiv:2006.01423 [cs].

[15] Carl Chiarella, Xue-Zhong He, and Cars Hommes. A dynamic analysis of moving average rules. *Journal of Economic Dynamics and Control*, 30(9):1729–1753, September 2006.

[16] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation, September 2014. arXiv:1406.1078 [cs, stat].

[17] Caroline Cohen, Baptiste Darbois Texier, David Quéré, and Christophe Clanet. The physics of badminton. *New Journal of Physics*, 17(6):063001, June 2015. Publisher: IOP Publishing.

[18] MMDetection Contributors. Openmmlab object detection toolbox. https://github.com/open-mmlab/mmdetection, 2018.

[19] MMPose Contributors. Openmmlab pose estimation toolbox and benchmark. https://github.com/open-mmlab/mmpose, 2020.

[20] MMTracking Contributors. Openmmlab video perception toolbox. https://github.com/open-mmlab/mmtracking, 2020.

[21] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham. Active Shape Models-Their Training and Application. *Computer Vision and Image Understanding*, 61(1):38–59, January 1995.

[22] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. ECO: Efficient Convolution Operators for Tracking, April 2017. arXiv:1611.09224 [cs].

[23] Yann Desmarais, Denis Mottet, Pierre Slangen, and Philippe Montesinos. A review of 3D human pose estimation algorithms for markerless motion capture. *Computer Vision and Image Understanding*, 212:103275, November 2021. Publisher: Academic Press.

[24] Richard O. Duda and Peter E. Hart. Use of the Hough transformation to detect lines and curves in pictures. *Commun. ACM*, 15(1):11–15, January 1972.

[25] Vittorio Ferrari, Manuel Marin-Jimenez, and Andrew Zisserman. Progressive search space reduction for human pose estimation. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2008. ISSN: 1063-6919.

[26] Rıza Alp Güler, Natalia Neverova, and Iasonas Kokkinos. DensePose: Dense Human Pose Estimation In The Wild, February 2018. arXiv:1802.00434 [cs].

[27] Jungong Han, Dirk Farin, and Peter H. N. de With. Generic 3-D Modeling for Content Analysis of Court-Net Sports Sequences. In Tat-Jen Cham, Jianfei Cai, Chitra Dorai, Deepu Rajan, Tat-Seng Chua, and Liang-Tien Chia, editors, *Advances in Multimedia Modeling*, Lecture Notes in Computer Science, pages 279–288, Berlin, Heidelberg, 2006. Springer.

[28] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, 2 edition, 2003.

[29] K. Hata and S. Savarese. CS 231 A Course Notes 4 : Stereo Systems and Structure from Motion. 2017.

[30] Junjie Huang, Zheng Zhu, Feng Guo, Guan Huang, and Dalong Du. The Devil is in the Details: Delving into Unbiased Data Processing for Human Pose Estimation, December 2020. arXiv:1911.07524 [cs].

[31] Yu-Chuan Huang, I.-No Liao, Ching-Hsuan Chen, Tsì-Uí Ík, and Wen-Chih Peng. TrackNet: A Deep Learning Network for Tracking High-speed and Tiny Objects in Sports Applications, July 2019. arXiv:1907.03698 [cs, stat].

[32] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1325–1339, July 2014. Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence.

[33] Mohammadreza Javadiha, Carlos Andujar, Michele Calvanese, Enrique Lacasa, Jordi Moyés, José Luis Pontón, Antonio Susín, and Jiabo Wang. PADELVIC: Multicamera videos and motion capture data of an amateur padel match. *Padel Scientific Journal*, 2(1):89–106, January 2024. Number: 1.

[34] Mohammadreza Javadiha, Carlos Andujar, Enrique Lacasa, Angel Ric, and Antonio Susin. Estimating Player Positions from Padel High-Angle Videos: Accuracy Comparison of Recent Computer Vision Methods. *Sensors (Basel, Switzerland)*, 21(10):3368, May 2021.

[35] Tao Jiang, Peng Lu, Li Zhang, Ningsheng Ma, Rui Han, Chengqi Lyu, Yining Li, and Kai Chen. RTMPose: Real-Time Multi-Person Pose Estimation based on MMPose, July 2023. arXiv:2303.07399 [cs].

[36] Hanbyul Joo, Tomas Simon, Xulong Li, Hao Liu, Lei Tan, Lin Gui, Sean Banerjee, Timothy Godisart, Bart Nabbe, Iain Matthews, Takeo Kanade, Shohei Nobuhara, and Yaser Sheikh. Panoptic Studio: A Massively Multiview System for Social Interaction Capture, December 2016.

[37] S.X. Ju, M.J. Black, and Y. Yacoob. Cardboard people: a parameterized model of articulated image motion. In *Proceedings of the Second International Conference on Automatic Face and Gesture Recognition*, pages 38–44, 1996.

[38] R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. 2002.

[39] Dieter Kraft. *A Software Package for Sequential Quadratic Programming*. Wiss. Berichtswesen d. DFVLR, 1988. Google-Books-ID: 4rKaGwAACAAJ.

[40] Xiang Li, Wenhai Wang, Lijun Wu, Shuo Chen, Xiaolin Hu, Jun Li, Jinhui Tang, and Jian Yang. Generalized Focal Loss: Learning Qualified and Distributed Bounding Boxes for Dense Object Detection, June 2020. arXiv:2006.04388 [cs].

[41] Yanjie Li, Sen Yang, Peidong Liu, Shoukui Zhang, Yunxiao Wang, Zhicheng Wang, Wankou Yang, and Shu-Tao Xia. SimCC: a Simple Coordinate Classification Perspective for Human Pose Estimation, July 2022. arXiv:2107.03332 [cs].

[42] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal Loss for Dense Object Detection, February 2018. arXiv:1708.02002 [cs] version: 2.

[43] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft COCO: Common Objects in Context, February 2015. arXiv:1405.0312 [cs].

[44] Paul Liu and Jui-Hsien Wang. MonoTrack: Shuttle trajectory reconstruction from monocular badminton video, May 2022. arXiv:2204.01899 [cs].

[45] Paul Liu and Jui-Hsien Wang. Monotrack: Shuttle trajectory reconstruction from monocular badminton video. https://github.com/jhwang7628/monotrack, 2023.

[46] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: a skinned multi-person linear model. *ACM Trans. Graph.*, 34(6):248:1–248:16, 2015.

[47] Chengqi Lyu, Wenwei Zhang, Haian Huang, Yue Zhou, Yudong Wang, Yanyi Liu, Shilong Zhang, and Kai Chen. RTMDet: An Empirical Study of Designing Real-Time Object Detectors, December 2022. arXiv:2212.07784 [cs].

[48] Rangi Lyu. Nanodet-plus: Super fast and high accuracy lightweight anchor-free object detection model. https://github.com/RangiLyu/nanodet, 2021.

[49] Archana .M and M. Geetha. Object Detection and Tracking Based on Trajectory in Broadcast Tennis Video. *Procedia Computer Science*, 58:225–232, December 2015.

[50] Miguel Soares Moreira. Multicamera System for Automatic Positioning of Objects in Game Sports. July 2016. Accepted: 2022-09-09T07:33:38Z.

[51] Kadir Nar. Ocsort-pip: Packaged version of the ocsort repository. https://github.com/kadirnar/ocsort-pip, 2022.

[52] Kadir Nar. Bytetrack-pip: Packaged version of the bytetrack repository. https://github.com/kadirnar/bytetrack-pip, 2023.

[53] Hoang Minh Nguyen, Burkhard Wünsche, Patrice Delmas, and Christof Lutteroth. 3D models from the black box: investigating the current state of image-based modeling. *Proceedings of the 20th International Conference on Computer Graphics, Visualisation and Computer Vision (WSCG), 2012*, pages 249–258, 2012.

[54] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning Deconvolution Network for Semantic Segmentation, May 2015. arXiv:1505.04366 [cs].

[55] Tom O'Malley, Elie Bursztein, James Long, François Chollet, Haifeng Jin, Luca Invernizzi, et al. Kerastuner. https://github.com/keras-team/keras-tuner, 2019.

[56] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy,

Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library, December 2019.

[57] Georgios Pavlakos, Luyang Zhu, Xiaowei Zhou, and Kostas Daniilidis. Learning to Estimate 3D Human Pose and Shape from a Single Color Image, May 2018. arXiv:1805.04092 [cs].

[58] Dario Pavllo, Christoph Feichtenhofer, David Grangier, and Michael Auli. 3D Human Pose Estimation in Video With Temporal Convolutions and Semi-Supervised Training. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7745–7754, June 2019. ISSN: 2575-7075.

[59] H. E. RAUCH, F. TUNG, and C. T. STRIEBEL. Maximum likelihood estimates of linear dynamic systems. *AIAA Journal*, 3(8):1445–1450, 1965. Publisher: American Institute of Aeronautics and Astronautics _eprint: https://doi.org/10.2514/3.3166.

[60] N Dinesh Reddy, Laurent Guigues, Leonid Pishchulin, Jayan Eledath, and Srinivasa G. Narasimhan. TesseTrack: End-to-End Learnable Multi-Person Articulated 3D Pose Tracking. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15185–15195, June 2021. ISSN: 2575-7075.

[61] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation, May 2015. arXiv:1505.04597 [cs].

[62] Lejun Shen, Qing Liu, Lin Li, and Yawei Ren. Reconstruction of 3D Ball/Shuttle Position by Two Image Points from a Single View. In Martin Lames, Dietmar Saupe, and Josef Wiemeyer, editors, *Proceedings of the 11th International Symposium on Computer Science in Sport (IACSS 2017)*, Advances in Intelligent Systems and Computing, pages 89–96, Cham, 2018. Springer International Publishing.

[63] David Sherry and Paul Hawkins. Video Processor Systems for Ball Tracking in Ball Games, June 2001.

[64] Leonid Sigal, Alexandru O. Balan, and Michael J. Black. HumanEva: Synchronized Video and Motion Capture Dataset and Baseline Algorithm for Evaluation of Articulated Human Motion. *International Journal of Computer Vision*, 87(1):4–27, March 2010.

[65] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition, April 2015. arXiv:1409.1556 [cs].

[66] Thomas Steinecker and Hans-Joachim Wuensche. A Simple and Model-Free Path Filtering Algorithm for Smoothing and Accuracy. In *2023 IEEE Intelligent Vehicles Symposium (IV)*, pages 1–7, June 2023. ISSN: 2642-7214.

[67] Nien-En Sun, Yu-Ching Lin, Shao-Ping Chuang, Tzu-Han Hsu, Dung-Ru Yu, Ho-Yi Chung, and Tsì-Uí İk. TrackNetV2: Efficient Shuttlecock Tracking Network. In *2020 International Conference on Pervasive Artificial Intelligence (ICPAI)*, pages 86–91, December 2020.

[68] Yu Sun, Qian Bao, Wu Liu, Yili Fu, Michael J. Black, and Tao Mei. Monocular, One-stage, Regression of Multiple 3D People, September 2021. arXiv:2008.12272 [cs].

[69] Shuhei Tarashima, Muhammad Abdul Haq, Yushan Wang, and Norio Tagawa. Widely Applicable Strong Baseline for Sports Ball Detection and Tracking, November 2023. arXiv:2311.05237 [cs].

[70] Tarashima, Shuhei and Haq, Muhammad Abdul and Wang, Yushan and Tagawa, Norio. Wasb: Widely applicable strong baseline for sports ball detection and tracking. https://github.com/nttcom/WASB-SBDT, 2023.

[71] Alexander Toshev and Christian Szegedy. DeepPose: Human Pose Estimation via Deep Neural Networks. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1653–1660, June 2014. arXiv:1312.4659 [cs].

[72] Hanyue Tu, Chunyu Wang, and Wenjun Zeng. VoxelPose: Towards Multi-Camera 3D Human Pose Estimation in Wild Environment, August 2020. arXiv:2004.06239 [cs].

[73] Unity Technologies. Unity, 2024. Game development platform.

[74] Gabriel Van Zandycke and Christophe De Vleeschouwer. Real-time CNN-based Segmentation Architecture for Ball Detection in a Single View Setup. In *Proceedings Proceedings of the 2nd International Workshop on Multimedia Content Analysis in Sports*, pages 51–58, October 2019. arXiv:2007.11876 [cs, eess].

[75] Gabriel Van Zandycke and Christophe De Vleeschouwer. Ball 3D Localization From A Single Calibrated Image. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 3471–3479, June 2022. arXiv:2204.00003 [cs, eess].

[76] Silvia Vinyes Mora. Computer vision and machine learning for in-play tennis analysis: framework, algorithms and implementation. January 2018. Accepted: 2019-03-26T14:18:32Z Publisher: Imperial College London.

[77] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[78] Christophe Vleeschouwer, Fan Chen, D Delannay, Christophe Parisot, Christophe Chaudy, Eric Martrou, and Andrea Cavallaro. Distributed video acquisition and annotation for sport-event summarization. January 2008.

[79] Timo von Marcard, Roberto Henschel, Michael J. Black, Bodo Rosenhahn, and Gerard Pons-Moll. Recovering accurate 3d human pose in the wild using imus and a moving camera. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[80] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, Wenyu Liu, and Bin Xiao. Deep High-Resolution Representation Learning for Visual Recognition, August 2019.

[81] Wanneng Wu, Min Xu, Qiaokang Liang, Li Mei, and Yu Peng. Multi-camera 3D ball tracking framework for sports video. *IET Image Processing*, 14(15):3751–3761, 2020. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1049/iet-ipr.2020.0757.

[82] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Online Object Tracking: A Benchmark. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2411–2418, June 2013. ISSN: 1063-6919.

[83] Bin Xiao, Haiping Wu, and Yichen Wei. Simple Baselines for Human Pose Estimation and Tracking, August 2018. arXiv:1804.06208 [cs].

[84] Yufei Xu, Jing Zhang, Qiming Zhang, and Dacheng Tao. ViTPose: Simple Vision Transformer Baselines for Human Pose Estimation, October 2022. arXiv:2204.12484 [cs].

[85] Yi Yang and Deva Ramanan. Articulated Human Detection with Flexible Mixtures of Parts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(12):2878–2890, December 2013. Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence.

[86] Hang Ye, Wentao Zhu, Chunyu Wang, Rujie Wu, and Yizhou Wang. Faster VoxelPose: Real-time 3D Human Pose Estimation by Orthographic Projection, July 2022. arXiv:2207.10955 [cs].

[87] De Jong Yeong, Gustavo Velasco-Hernandez, John Barry, and Joseph Walsh. Sensor and Sensor Fusion Technology in Autonomous Vehicles: A Review. *Sensors*, 21:2140, March 2021.

[88] Siu Lun Yeung. *Efficient Kalman Filtering and Smoothing*. phd, University of Liverpool, August 2021.

[89] Changqian Yu, Bin Xiao, Changxin Gao, Lu Yuan, Lei Zhang, Nong Sang, and Jingdong Wang. Lite-HRNet: A Lightweight High-Resolution Network, April 2021.

[90] Dejun Zhang, Yiqi Wu, Mingyue Guo, and Yilin Chen. Deep Learning Methods for 3D Human Pose Estimation under Different Supervision Paradigms: A Survey. *Electronics*, 10(18):2267, January 2021. Number: 18 Publisher: Multidisciplinary Digital Publishing Institute.

[91] Feng Zhang, Xiatian Zhu, Hanbin Dai, Mao Ye, and Ce Zhu. Distribution-Aware Coordinate Representation for Human Pose Estimation, October 2019. arXiv:1910.06278 [cs].

[92] Haotian Zhang, Cristobal Sciutto, Maneesh Agrawala, and Kayvon Fatahalian. Vid2Player: Controllable Video Sprites that Behave and Appear like Professional Tennis Players, December 2020. arXiv:2008.04524 [cs].

[93] Yifu Zhang, Peize Sun, Yi Jiang, Dongdong Yu, Fucheng Weng, Zehuan Yuan, Ping Luo, Wenyu Liu, and Xinggang Wang. ByteTrack: Multi-Object Tracking by Associating Every Detection Box, April 2022. arXiv:2110.06864 [cs].

[94] Hengshuang Zhao, Xiaojuan Qi, Xiaoyong Shen, Jianping Shi, and Jiaya Jia. ICNet for Real-Time Semantic Segmentation on High-Resolution Images, August 2018. arXiv:1704.08545 [cs].

[95] Wentao Zhu, Xiaoxuan Ma, Zhaoyang Liu, Libin Liu, Wayne Wu, and Yizhou Wang. Motionbert: A unified perspective on learning human motion representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.