



Institute for Information
and Communication Technologies,
Electronics and Applied Mathematics

3D Objects Localization from a Single Calibrated Viewpoint: A Study of Basketballs

Gabriel Van Zandycke

Thesis submitted in partial fulfillment
of the requirements for the degree of
Ph.D. in Engineering Sciences

Dissertation committee:

Prof. Christophe De Vleeschouwer, supervisor (UCLouvain, Belgium)

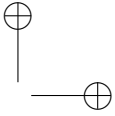
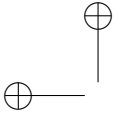
Prof. Benoît Macq (UCLouvain, Belgium)

Dr. Damien Delannay (Sportradar AG)

Prof. Rainer Lienhart (University of Augsburg, Germany)

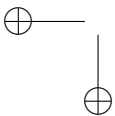
Prof. Laurent Francis, president (UCLouvain, Belgium)

Version of Tuesday 30th April, 2024 (13:37).

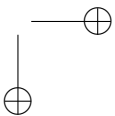


—

—

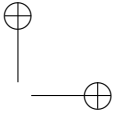
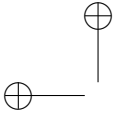


|



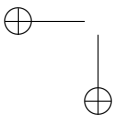
Préambule

Il y a des aventures qui commencent avec incertitude, un mélange d'audace et d'ignorance, et c'est dans cet esprit que j'ai commencé la formation doctorale dont le présent tapuscrit est l'aboutissement. Entamée dans le but de sortir de ma zone de confort et d'affronter de nouveaux défis, cette aventure académique m'a permis d'explorer de nouveaux horizons. Et au final, même si les contributions académiques restent modestes, j'ai acquis une multitude de connaissances et de compétences dans des domaines divers et variés. Ainsi, la valeur de ce travail ne réside pas seulement dans cet ouvrage, mais d'avantage dans la démarche intellectuelle qui l'a nourri. À l'image de ce préambule, ce tapuscrit se veut succinct et concis. Bonne lecture.

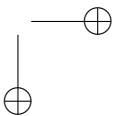


—

—



|



Remerciements

L'aboutissement de cette thèse de doctorat n'aurait jamais été possible sans l'aide et le soutien de nombreuses personnes que je tiens à remercier ici.

Tout d'abord, je souhaite à exprimer ma gratitude à mon promoteur de thèse, Christophe, pour sa supervision éclairée, son optimisme à toute épreuve, sa disponibilité et sa confiance tout au long de ce doctorat. Ses encouragements et ses conseils ont entretenu ma motivation jusqu'à la fin. Je remercie aussi les membres du jury de cette thèse, et en particulier les membres de mon comité d'accompagnement.

Je voudrais également remercier mes collègues de l'UCLouvain pour l'environnement intellectuel particulièrement riche dans lequel j'ai baigné durant ce doctorat. Tout d'abord Maxime avec qui j'ai partagé le bureau, pour les nombreux échanges, son expertise en génie logiciel qui m'a permis de progresser considérablement, et sans oublier sa camaraderie tout au long de ce parcours. Mais aussi les *Dauphins du Stévin*, et en particulier Pierre-Yves, Stéphanie, Bénible, Kévin, Simon, Victor, Antoine, Anne-Sophie et Niels. Les moments conviviaux et multiples discussions (politiques, scientifiques, climatiques, pédagogiques, et bien autres) partagés dans la kfet pirate ont rendu cette expérience doctorale particulièrement enrichissante. Et enfin le *Concatenate Group* au sein duquel j'ai eu la chance de terminer cette thèse, avec Abolfazl, Antoine, Baptiste, Alexandre, Vladimir, Benoît, Gilles, et Danny. Je n'oublierai pas les échanges techniques, les CCB partagés ensemble, ainsi que leur soutien indéfectible sur la toute toute fin de ma thèse, et en particulier les soixante-cinq dernières semaines. Sans oublier le personnel administratif et technique, Brigitte, François et Souley pour leur support et services techniques; Patricia, Nathalie et Isabelle pour leur aide administrative; ainsi que Beni pour le soutien psychologique. J'inclus également les administrateurs du CISM pour l'infrastructure de calcul qui a servi cette thèse, et en particulier Damien pour ses précieux conseils.

Je tiens à exprimer ma gratitude à mes collègues de chez Keemotion et Sportradar, et en particulier à Damien, Alex, Milton, Nick et Luka pour leur soutien indispensable à l'accomplissement de cette thèse ainsi que leur support au milieu académique de manière général. Je tiens aussi à mentionner Damien, Pascaline, Cédric, Jacek, Julien, Davide, Marcello et Carlo qui, d'une manière ou d'une autre, m'ont inspiré ou aidé durant cette thèse. Et je suis également reconnaissant du support administratif apporté par Pitou et Anne-Lise.

Je tiens également à remercier chaleureusement divers collaborateurs, comme le Professeur Lienhart, le groupe de recherche de Marc Van Droogenbroek et l'équipe de SoccerNet, mais aussi les annotateurs, et en particulier Lua, pour la constitution des datasets utilisés dans cette thèse, et dont une partie a pu être rendue publique pour la communauté scientifique.

Pour terminer, je voudrais remercier tous mes amis et membres de ma famille qui m'ont entourés et soutenus dans l'accomplissement de cette aventure. Je souhaite en particulier exprimer ma profonde gratitude à mon épouse, Maëlle, qui a été un pilier essentiel dans la réalisation de ce projet. Par sa patience, son soutien et son amour inconditionnel, elle m'a permis de surmonter les défis et persévérer dans les moments difficiles. Finalement je remercie aussi mes enfants, Gaspard, Sophie et Brieuc, qui ont été une source infatigable pour me motiver à me lever chaque ~~matin~~ nuit matin, et sans qui, arriver au bout de cette thèse, aurait peut-être été trop facile.

Abstract

Localizing objects in the 3D space is a fundamental computer vision task that finds applications in various domains such as autonomous driving and augmented reality. In the context of team sports analytics, the accurate localization of the ball provides fundamental cues of the game being played. Traditionally, this localization involves triangulation from multiple viewpoints. However, there are many practical cases where 3D localization is valuable even though only a single viewpoint is available ; typically when the acquisition system is not within the stakeholder's control. Moreover, multi camera systems are complex and expensive to deploy. In this thesis, we address the task of 3D object localization from a single calibrated viewpoint, and apply it to basketballs.

We first study approaches for annotating ball 3D position from a single viewpoint and present an original solution that leverages known camera calibration to deduce ball 3D position from the annotation of its center and its ground projection. Our method is applicable to any object or point whose 3D position can be described by its height relative to a known ground floor. We use this solution to annotate high resolution images captured during 165 professional basketball games, establishing a basketballs dataset of unmet quality and diversity. Using our dataset, we leverage the capabilities of deep neural networks to detect basketballs in the image space. Based on a fast multi-scale segmentation model trained to predict the ball segmentation mask, our original method exhibits state of the art performances. We then explore the task of 3D ball localization from individual images with known calibration. Given the detections from our detector, we propose and compare two alternative neural-network-based approaches to estimate the ball 3D position. In the first approach, the neural network is trained to estimate ball diameter in the image space, and knowledge of real ball diameter is used to convert the prediction into depth. In the second approach, the neu-

ral network directly predicts ball height in the image space. It doesn't make any assumption on the object and could be extended to other domains beyond sport. Finally, we evaluate our image-based approaches on sequences and exploit a physical motion model to improve individual predictions during ballistic trajectories.

Contents

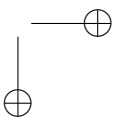
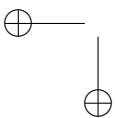
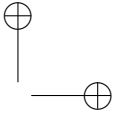
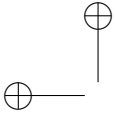
Contents	vii
List of publications	xi
List of symbols	xiii
1 Introduction	1
2 Background	5
2.1 Notations	5
2.2 Camera calibration	5
2.2.1 Homogeneous coordinates	6
2.2.2 Coordinate systems	6
2.2.3 Word to camera coordinates	6
2.2.4 Projection	7
2.2.5 Distortion model	7
2.2.6 Pixel coordinates	9
2.2.7 Full projection model	10
2.2.8 Linear projection model	10
2.2.9 Affine transformations	11
2.3 Convolutional Neural Networks	11
2.3.1 Artificial neural networks as universal approximators	11
2.3.2 Neural networks for image processing	13
2.3.3 Neural networks supervision	14
2.3.4 Challenges of training neural networks	15
2.3.5 Popular network architectures for computer vision .	17
2.4 Ballistic motion models for basketballs	18
2.4.1 Motion under gravity alone	18
2.4.2 Motion with air friction	19
2.4.3 Specific case of horizontal spin axis	20

CONTENTS

2.4.4	Application to Basketballs	21
3	Ball 3D annotations from a single calibrated viewpoint	23
3.1	Existing datasets	24
3.2	Dataset acquisition	25
3.2.1	Basketball Instants Dataset	26
3.2.2	Basketball Sequences Dataset	28
3.2.3	Practical constraints	28
3.2.4	Open science with open data	29
3.3	Ball 3D annotation from a single viewpoint	30
3.3.1	Annotation method	30
3.3.2	Evaluation procedure	31
3.3.3	Experimental results	33
3.3.4	Discussion	34
3.3.5	Conclusion	35
3.4	Ballistic trajectories 3D annotation	35
3.4.1	Annotation strategies	35
3.4.2	Evaluation procedure	37
3.4.3	Experimental study	37
3.4.4	Discussion	39
3.4.5	Conclusion	40
3.5	Summary	40
4	Ball detection in an image	43
4.1	Related works	43
4.2	Method	45
4.3	Validation Methodology	48
4.4	Experimental results	49
4.4.1	Inputs choice and detection rule validation	49
4.4.2	Comparison with SotA object detection method	50
4.4.3	Test-time data augmentation	51
4.5	Conclusion	55
5	Ball 3D localization from a single calibrated image	57
5.1	Related works	57
5.2	Diameter approach	58
5.2.1	Method	58
5.2.2	Datasets	61
5.2.3	Evaluation procedure	62

CONTENTS

5.2.4	Experiments	63
5.2.5	Discussion	71
5.3	Height approach	72
5.3.1	Method	73
5.3.2	Dataset	74
5.3.3	Evaluation procedure	74
5.3.4	Experiments	76
5.3.5	Discussion	79
5.4	Deeper Comparison	80
5.5	Conclusion	81
6	Ball 3D localization over ballistic trajectories	83
6.1	Related works	83
6.2	Method	85
6.2.1	Stage 1 - Ball candidates detection	85
6.2.2	Stage 2 - Ball candidates characterisation	85
6.2.3	Stage 3 - Ballistic trajectories fitting	87
6.3	Evaluation procedure	88
6.4	Results	90
6.5	Discussion	92
6.6	Conclusion	92
7	Conclusion	95
7.1	Results summary	95
7.2	Challenges	97
7.3	Perspectives	98
	Bibliography	101



List of publications

This typescript marks the conclusion of a doctoral training which led to the publication of the following research articles:

- **Gabriel Van Zandycke**, and Christophe De Vleeschouwer. Real-time CNN-based segmentation architecture for ball detection in a single view setup. In *Proceedings of the 2nd International Workshop on Multimedia Content Analysis in Sports, Nice, France*, November 2019, page 51-58. doi:10.1145/3347318.3355517
- Seyed Abolfazl Ghasemzadeh, **Gabriel Van Zandycke**, Maxime Istasse, Niels Sayez, Amirafshar Moshtaghpour, and Christophe De Vleeschouwer. Deepsporthlab: a unified framework for ball detection, player instance segmentation and pose estimation in team sports scenes. In *32nd British Machine Vision Conference 2021, BMVC 2021, Online, November 22-25, 2021*, page 379. BMVA Press, 2021.
- **Gabriel Van Zandycke**, and Christophe De Vleeschouwer. 3D Ball Localization From a Single Calibrated Image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, USA*, June 2022, page 3472-3480. doi:10.1109/CVPRW56347.2022.00391
- **Gabriel Van Zandycke**, Vladimir Somers, Maxime Istasse, Carlo Del Don, and Davide Zambrano. Deepsporthradar-v1: Computer vision dataset for sports understanding with high quality annotations. In *MMSports@MM 2022: Proceedings of the 5th International ACM Workshop on Multimedia Content Analysis in Sports, Lisboa, Portugal*, 14 October 2022, pages 1–8. ACM, 2022. doi:10.1145/3552437.3555699
- Marcello Davide Caio, **Gabriel Van Zandycke**, and Christophe De Vleeschouwer. Context-Aware 3D Object Localization from Single

CONTENTS

Calibrated Images: A Study of Basketballs. In *Proceedings of the 6th International Workshop on Multimedia Content Analysis in Sports, Ottawa, Canada*, November 2023. doi:10.1145/3606038.3616173

List of symbols

Projective geometry

- λ Homogeneous component for vectors using homogeneous coordinates, page 6
- $\mathcal{D}$ Distortion operation applied to the linear projection of world coordinates to account for lens distortion, see equation (2.3), page 8
- $\mathcal{P}$ Projection operation that converts world coordinates into pixel coordinates, see equation (2.9), page 10
- $\mathcal{R}$ Rectification operation applied to the linear projection of pixel coordinates to account for lens distortion, see equation (2.4), page 9
- $\mathbf{c}^{\mathcal{W}}$ Camera position in world coordinate system, page 7
- $\mathbf{t}^{\mathcal{C}}$ Translation vector, i.e. world coordinate system's origin expressed in the camera coordinates system, see equation (2.1), page 7
- K Camera intrinsic matrix, see equation (2.5), page 9
- P Projection matrix for the pinhole camera model, see equation (2.10), page 11
- R Rotation matrix, see equation (2.1), page 7

Physics of projectiles

- c_D Drag coefficient, see equation (2.22), page 19
- c_L Lift coefficient for spinning projectiles, see equation (2.23), page 20
- $\mathbf{g}$ Vector vertical with respect to the Earth surface, pointing down, with a magnitude of $g \approx 9.81 \text{ m/s}^2$, see equation (2.19), page 18

LIST OF SYMBOLS

Ball detection

- Ψ Detector heatmap predicting ball segmentation mask, page 46
- τ Detection threshold on heatmap Ψ , page 47
- k Number of detections considered, page 47

Ball 3D Localization

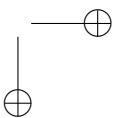
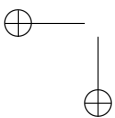
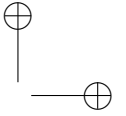
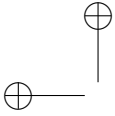
- α Balancing factor between $\mathcal{L}^c$ and $\mathcal{L}^d$, see equation (5.3), page 60
- $\hat{c}$ Estimate of ball presence in the input image patch, page 60
- $\hat{d}$ Estimate of ball diameter in pixels, page 60
- $\mathcal{H}_w$ Model head, page 60
- $\mathcal{L}^c$ Loss supervising ball presence classification, see equation (5.1), page 60
- $\mathcal{L}^d$ Loss supervising ball diameter regression, see equation (5.2), page 60
- $\mathcal{M}_w$ Model backbone, page 60
- ϕ Ball true diameter (in meters), page 61
- ρ Filter diameter of the morphological operation in the HCT baseline, page 59
- τ_h Higher bound for the Canny adaptive thresholding in the HCT baseline, page 59
- τ_l Lower bound for the Canny adaptive thresholding in the HCT baseline, page 59
- $\mathbf{b}^w$ Ball position in world coordinates system, see equation (5.6), page 61
- I Image patch centered on a ball candidate, page 59

Ballistic trajectories detection and fitting

- ϵ^* Custom error function that decouples detection error from diameter error, see equation (6.2), page 88
- ϵ^e Euclidean distance for ballistic trajectory fitting from 3D observations, page 90

LIST OF SYMBOLS

$\mathcal{L}^s$	Loss supervising ball state classification, see equation (6.1), page 87
μ	Sliding window initial duration (in milliseconds), page 87
ρ	Minimum ratio of inliers to accept a model, page 88
h	RanSaC iterations, page 88
m	Minimum number of inliers to accept a model, page 88
n	RanSaC sample size, page 88



Chapter 1

Introduction

The task of localizing objects in the three dimensional (3D) space is a fundamental problem in computer vision and robotics. It involves identifying the position and orientation of objects in the 3D world, and finds applications in domains like transportation with autonomous driving, entertainment with augmented reality and virtual reality, and sport where tracking the position of players and the ball provides essential insights into the game.

The most reliable approaches to object 3D localization typically involve the utilization of depth information from specific devices like active ToF cameras or LiDAR systems. These solutions are however considerably expensive and their angular resolution is limited. Hence, for relatively small objects at far ranges, the only reasonable alternative is to use standard cameras. When deployed at different angles, detections from each viewpoint can be triangulated to estimate the 3D position of objects. However, when only a single viewpoint is available, the task becomes a real challenge. This typically happens when complex and expensive multi-camera setups are not possible, or simply when the acquisition systems are not within the stakeholder's control.

In this thesis, we address the challenging task of 3D object localization from a single viewpoint with standard cameras, and focus on basketballs for which 3D localization is both hard, due to many interactions with players and poor contrasts with the scene, and extremely valuable. Since the object of interest is spherical, the object orientation problem doesn't need to be addressed. Like any other approach to 3D object localization, it requires to know the exact position and orientation of the camera in the scene, referred to as camera calibration. We build on convolutional neural networks which

Chapter 1. Introduction

have proven to outperform many classical computer-vision methods for many applications.

Knowing the 3D position of the ball in team sports is useful in many aspects. For instance, in the context of scoring and virtually assisted refereeing, it provides the information necessary for determining whether a shot is successful. It also provides valuable insights for players and coaches as it allows reviewing players' shots and passes techniques, helping them refine their skills. For automated capture systems, it provides a fundamental cue of the action being played, enabling automated replays. And finally, advanced 3D reconstruction can enhance the viewing experience for fans by providing immersive visualizations, such as overlaying graphics showing the ball's trajectory or providing virtual replays from different angles, adding excitement and engagement to broadcasts. Overall, a comprehensive understanding of the ball 3D position not only meets a demand within the sport ecosystem, but also contributes to the continual evolution and enrichment of the overall sport experience.

In this thesis, we focus on a top-view analysis of the game, which meets many applications described above. Therefore, as illustrated in figure 1.1, the metric we are interested in, is the 3D error between the true ball position and the estimated ball position, projected on the horizontal court floor.

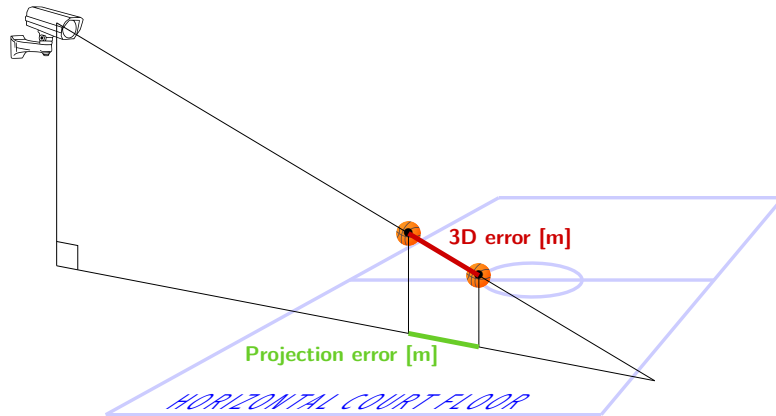
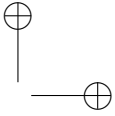
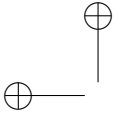


Figure 1.1: **Projection error.** By focussing on a top-view analysis of the game, the metric we are interested in is the 3D error between the true ball position and the estimated ball position, projected on the horizontal court floor.

The task of 3D ball localization in team sports scenes has already been

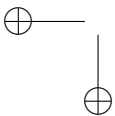
addressed, leading to industrial products like video assistant referee systems that provides the automated offside detection and the goal line technologies in soccer, or the automated line calling systems in tennis. However, all these solutions rely on expensive and complex multi-view setups that require to be deployed in every venue. Few research works investigate the task of ball 3D localization from a single viewpoint. However, they all restrict the 3D localization to ballistic trajectories where a motion model is used to infer 3D positions from 2D detections. This work stands therefore at the forefront of research and paves the way for numerous affordable solutions relying on ball 3D localization from a single viewpoint.

This thesis is organized as follows. **Chapter 2** introduces background topics necessary for the comprehension of the following chapters, including camera calibration, convolutional neural networks and ballistic motion models. In **chapter 3**, we address the important task of data acquisition and annotation. We present a method for annotation of basketballs from a single calibrated image, and study different approaches for ballistic trajectories annotation, with the goal of minimizing annotation efforts. **Chapter 4** presents a method for real-time ball detection in the image space, leveraging a fast convolutional neural network. In **chapter 5**, we address the task of ball 3D localization from a single viewpoint, the core of this thesis. We present two alternative approaches, one of which is applicable to any object and can be extended to other domains beyond sport. Finally, in **chapter 6**, we evaluate our image-based ball 3D localization model on sequences and use a ballistic motion model to improve individual predictions during ballistic trajectories.

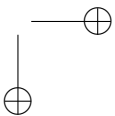


—

—



|



Chapter 2

Background

In this chapter, we introduce background topics necessary for the comprehension of the following chapters. Camera calibration is used throughout the thesis, convolutional neural networks are used in chapter 4 and 5, and ballistic motion models are used in chapter 3 and 6.

2.1 Notations

Throughout this manuscript, we use several notation conventions. Scalars are represented by minuscule letters a, b, x, y or z . Vectors are represented by bold minuscule letters $\mathbf{a}, \mathbf{b}, \mathbf{x}, \mathbf{y}$ or $\mathbf{z}$. Matrices are represented by capital letters A, B, X, Y or Z . Functions are represented by calligraphic letters $\mathcal{A}, \mathcal{F}, \mathcal{N}$.

2.2 Camera calibration

Camera calibration enables the establishment of the correspondences between points in the real world (3D coordinates) and the image pixels (2D coordinates) by modeling how light travels between the scene and the camera sensor. The distortion brought by the lens is modeled using a simplified version of the Brown-Conrady model (Brown, 1966; Conrady, 1919) used in the OpenCV library (Gary, 2015).

Chapter 2. Background

2.2.1 Homogeneous coordinates

Vectors representing 2D and 3D points are column vectors. To simplify the mathematics of perspective projection, we use homogeneous coordinates in which those vectors are represented with an additional component. For instance, a 3D point located in coordinates (x, y, z) is represented with a 4 elements vector $[\lambda x, \lambda y, \lambda z, \lambda]^T$, with $\lambda \in \mathbb{R}_0$.

2.2.2 Coordinate systems

Describing how the 2D to 3D correspondences are computed imply four different coordinate systems:

- The world 3D coordinate system, defined by a conventional origin and an orthonormal basis.
- The camera 3D coordinate system, defined by an origin at the camera center of projection called the *principal point*, a z -axis pointing towards the scene, an x -axis along the sensor width pointing towards the right, and a y -axis along the sensor height pointing towards the bottom, constituting an other orthonormal basis.
- The 2D plane $z = 1$ in the camera 3D coordinate system in which 3D positions are being projected.
- The 2D pixel coordinates on the camera sensor towards which the projected 2D positions are transformed.

As a notation, coordinates and vectors will be attributed with an exponent corresponding to the coordinate system they are expressed in. World coordinates system with $(\bullet)^W$, camera coordinates system with $(\bullet)^C$ and image space with $(\bullet)^I$.

2.2.3 Word to camera coordinates

The camera position and orientation relative to the world coordinate system is defined by the *extrinsic parameters*. The transformation from world coordinates (x^W, y^W, z^W) to camera coordinates (x^C, y^C, z^C) is composed of a rotation followed by a translation. The rotation is defined by a rotation matrix R using Euler angles in a right-handed orthogonal system¹ with an

¹Euler angles define a 3D rotation starting with a rotation around x followed by a rotation around y followed by a rotation around z .

2.2. Camera calibration

alias transformation². The translation is defined by a translation vector $\mathbf{t}^C$ representing the position of the origin of the world coordinate system in the camera coordinate system. Hence,

$$\lambda \begin{bmatrix} x^C \\ y^C \\ z^C \\ 1 \end{bmatrix} = \begin{bmatrix} R_{3 \times 3} & \mathbf{t}_{3 \times 1}^C \\ 0_{1 \times 3} & 1 \end{bmatrix} \begin{bmatrix} x^W \\ y^W \\ z^W \\ 1 \end{bmatrix} \quad (2.1)$$

If $\mathbf{t}^C$ is expressed in the camera coordinate system, the position of the camera expressed in world coordinate system is $\mathbf{c}^W := -R^{-1}\mathbf{t}^C = -R^T\mathbf{t}^C$ (since R is a rotation matrix).

2.2.4 Projection

The 3D coordinates are projected into the plane $z^C = 1$ with the following operation where $I_{3 \times 3}$ is the identity matrix and $\mathbf{0}_{3 \times 1}$ a column of 3 zeros:

$$\lambda \begin{bmatrix} x_u \\ y_u \\ 1 \end{bmatrix} = \begin{bmatrix} I_{3 \times 3} & \mathbf{0}_{3 \times 1} \end{bmatrix} \begin{bmatrix} x^C \\ y^C \\ z^C \\ 1 \end{bmatrix} \quad (2.2)$$

When applying the projection to a given 3D point, different values of λ will always give the same 2D point. However, when applying the inverse operation on a given 2D point, different values of λ yield different 3D points. This is obvious when simplifying the relation between the two points: the column $\mathbf{0}_{3 \times 1}$ cancels the homogeneous component of the 3D point.

2.2.5 Distortion model

The lens through which light passes before reaching the camera sensor brings two types of distortions illustrated in figure 2.1. The radial distortions, caused by the spherical nature of the lens, are causing lines far from the principal point to look curved; and the tangential distortions, caused by the misalignment of the lens with respect to the camera z-axis, are causing the image to look tilted.

²The rotation is applied to the basis, which means that the opposite rotation is applied to the coordinates.

Chapter 2. Background

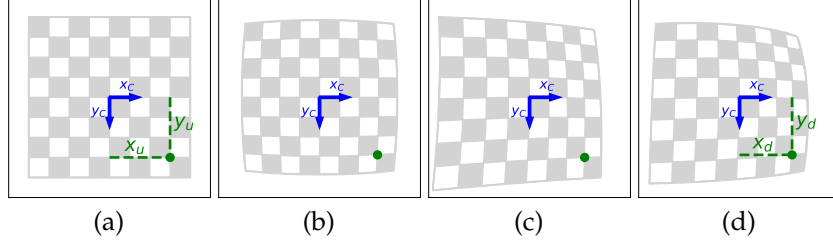


Figure 2.1: **Lens distortions visualized.** A checkerboard of 8×8 squares and a green dot seen from 4 cameras with different sets of distortion parameters. (a) No distortion, the green point undistorted coordinates are (x_u, y_u) . (b) Radial distortions only. (c) Tangential distortions only. (d) Distortions combined, the green point coordinates are now (x_d, y_d) in the camera coordinate system.

As a result, when converting 3D world coordinates to 2D pixel coordinates, a distortion operation $\mathcal{D}$ must be applied to the 2D coordinates after the projection. Conversely, when retrieving 3D world coordinates from 2D pixel coordinates, a rectify operation $\mathcal{R}$ must be applied before applying the inverse projection.

The following relationship models how the 2D coordinates after distortion $(x_d, y_d, 1)$ are computed from the 2D coordinates that result from the linear projection $(x_u, y_u, 1)$. Remember that those coordinates are expressed in the camera coordinate system, i.e. they are distances in meters (not pixels), see figure 2.1,

$$\begin{bmatrix} x_d \\ y_d \\ 1 \end{bmatrix} = \mathcal{D}(x_u, y_u) = \begin{bmatrix} \overbrace{x_u(1+k_1r_u^2+k_2r_u^4+k_3r_u^6+\dots)}^{\text{radial component}} + \overbrace{[2p_1x_uy_u+p_2(r_u^2+2x_u^2)]}^{\text{tangeantial component}} \\ \overbrace{y_u(1+k_1r_u^2+k_2r_u^4+k_3r_u^6+\dots)}^{\text{radial component}} + \overbrace{[2p_2x_uy_u+p_1(r_u^2+2y_u^2)]}^{\text{tangeantial component}} \\ 1 \end{bmatrix} \quad (2.3)$$

where $k_1, k_2, k_3, \dots$ are the radial distortion coefficients, t_1, t_2 are the tangential distortion coefficients, and $r_u^2 := x_u^2 + y_u^2$. In our model, we use 3 radial distortion coefficients, which makes a total of 5 coefficients. Those coefficients are found by running an optimisation algorithm on a set of 2D points to 3D points correspondences.

When doing the inverse operation, we need an expression of (x_u, y_u) as a function of (x_d, y_d) . Because the distortion operation described in equation 2.3 cannot be inverted analytically, we make an approximation and assume that the distortion at (x_d, y_d) is the same than the distortion at

2.2. Camera calibration

(x_u, y_u) , and use the same distortion coefficients $k_{1,2,3}$ and $p_{1,2}$:

$$\begin{aligned} 2p_1x_u y_u + p_2(r_u^2 + 2x_u^2) &\approx T_{p_{1,2}}^x := 2p_1x_d y_d + p_2(r_d^2 + 2x_d^2) \\ 2p_2x_u y_u + p_1(r_u^2 + 2y_u^2) &\approx T_{p_{1,2}}^y := 2p_2x_d y_d + p_1(r_d^2 + 2y_d^2) \\ (1 + k_1r_u^2 + k_2r_u^4 + k_3r_u^6) &\approx R_{k_{1,2,3}} := (1 + k_1r_d^2 + k_2r_d^4 + k_3r_d^6) \end{aligned}$$

Hence, we get the following analytical expression that models the rectification operation:

$$\begin{bmatrix} x_u \\ y_u \\ 1 \end{bmatrix} = \mathcal{R}(x_d, y_d) \approx \begin{bmatrix} (x_d - T_{p_{1,2}}^x)/R_{k_{1,2,3}} \\ (y_d - T_{p_{1,2}}^y)/R_{k_{1,2,3}} \\ 1 \end{bmatrix} \quad (2.4)$$

2.2.6 Pixel coordinates

Finally, the transformation between the 2D coordinates (x_d, y_d) in the plane $z^C = 1$ and the 2D pixel coordinates on the camera sensor (x^I, y^I) is represented by a single matrix K parameterised by the *intrinsic parameters* and combines the following affine transformations:

- The scale produced by the focal length, with f being the distance between the camera principal point and the sensor on which points are projected.
- The scale factors m_x and m_y relating pixels units to distance units (normally $m_x = m_y$ because pixels are squares).
- The translation from the camera principal point to a top-left origin, with (u_0, v_0) being the position of the principal point expressed in the image coordinate system.
- A skew operation between the sensor axis, with parameter γ (normally $\gamma = 0$ because pixels are squares).

$$K := \begin{bmatrix} f \cdot m_x & \gamma & u_0 \\ 0 & f \cdot m_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \quad (2.5)$$

Chapter 2. Background

In addition, the intrinsic parameters also include the sensor width and height in pixels that delimit the resulting image borders. The operation that transforms (x_d, y_d) coordinates into pixel coordinates $(x^{\mathcal{I}}, y^{\mathcal{I}})$ is therefore:

$$\lambda \begin{bmatrix} x^{\mathcal{I}} \\ y^{\mathcal{I}} \\ 1 \end{bmatrix} = [K_{3 \times 3}] \begin{bmatrix} x_d \\ y_d \\ 1 \end{bmatrix} \quad (2.6)$$

2.2.7 Full projection model

Combining the operations described by equations 2.1, 2.2, 2.3 and 2.5, we get the full projection model, described by the following equation.

$$\lambda \begin{bmatrix} x^{\mathcal{I}} \\ y^{\mathcal{I}} \\ 1 \end{bmatrix} = [K_{3 \times 3}] \mathcal{D} \left([R_{3 \times 3} \quad \mathbf{t}_{3 \times 1}^{\mathcal{C}}] \begin{bmatrix} x^{\mathcal{W}} \\ y^{\mathcal{W}} \\ z^{\mathcal{W}} \\ 1 \end{bmatrix} \right) \quad (2.7)$$

And the inverse model:

$$\lambda \begin{bmatrix} x^{\mathcal{W}} \\ y^{\mathcal{W}} \\ z^{\mathcal{W}} \\ 1 \end{bmatrix} = [R_{3 \times 3} \quad \mathbf{t}_{3 \times 1}^{\mathcal{C}}]^{-1} \mathcal{R} \left([K_{3 \times 3}]^{-1} \begin{bmatrix} x^{\mathcal{I}} \\ y^{\mathcal{I}} \\ 1 \end{bmatrix} \right) \quad (2.8)$$

Using vector notations, we define the operation $\mathcal{P}$ that transforms world coordinates into pixel coordinates:

$$\mathbf{x}^{\mathcal{I}} = \mathcal{P}(\mathbf{x}^{\mathcal{W}}) \quad (2.9)$$

2.2.8 Linear projection model

In many cases, there are not enough 2D to 3D correspondences to estimate the distortion coefficients. Besides, near the image center, the distortion is usually negligible anyway. For this reason, we sometimes refer to the linear projection model called the *pinhole camera model* and entirely described by a rectangular projection matrix P that converts 3D world coordinates into 2D

2.3. Convolutional Neural Networks

pixel coordinates.

$$P := \begin{bmatrix} K_{3 \times 3} & \mathbf{0}_{3 \times 1} \end{bmatrix} \begin{bmatrix} R_{3 \times 3} & \mathbf{t}_{3 \times 1}^c \\ \mathbf{0}_{1 \times 3} & 1 \end{bmatrix} = K_{3 \times 3} \begin{bmatrix} R_{3 \times 3} & \mathbf{t}_{3 \times 1}^c \end{bmatrix} \quad (2.10)$$

The opposite operation requires P^{-1} , the left pseudo-inverse of P .

2.2.9 Affine transformations

When affine transformation like cropping or scaling are applied on images, the camera matrix K is affected by the corresponding affine matrix A which describes how positions from the original image are converted into positions in the final image.

$$K \leftarrow A \times K \quad (2.11)$$

2.3 Convolutional Neural Networks

For the last decade, the computer vision community has strongly shifted their man-crafted models towards general purpose deep artificial neural networks achieving state-of-the-art results on a wide range of tasks like image classification, semantic segmentation, object detection or tracking. While many variations of neural networks exist, the convolutional neural networks (CNNs) are the prevalent choice for any task related to images.

2.3.1 Artificial neural networks as universal approximators

The fundamental building block of artificial neural networks is the artificial neuron characterized by equation 2.12. Inspired by the biological neural cells which output signals through their axon upon activation of the combined signals from their dendrites, artificial neurons simply apply an activation function to a weighted sum of their input values. Hence, For n input scalars $x_i, i \in \{1, \dots, n\}$, the neuron embodies n weights w_i , a bias b and an activation function $\mathcal{A} : \mathbb{R} \rightarrow \mathbb{R}$, and outputs a scalar y . The activation function introduces the non-linearity required to enable the approximation of non-linear signals, and a typical choice is to use the rectified linear unit

Chapter 2. Background

(ReLU (Nair and Hinton)) for which $\mathcal{A}(x) = \max(0, x)$.

$$y = \mathcal{A} \left(\sum_{i=1}^n w_i x_i + b \right) \quad (2.12)$$

Combining m neurons sharing the same input values makes a layer, characterized by equation 2.13 and parameterized by the weight matrix $W \in \mathbb{R}^{n \times m}$, the biases vector $\mathbf{b} \in \mathbb{R}^m$, and where $\mathcal{A}$ is applied element-wise. Such function is called a *dense layer* because every output in $\mathbf{y}$ is connected to every input in $\mathbf{x}$.

$$\mathbf{y} = \mathcal{A}(W\mathbf{x} + \mathbf{b}) \quad (2.13)$$

By stacking layers such as the output of one become the input of the following, with an initial input $\mathbf{x} \in \mathbb{R}^n$ and a final output $\mathbf{y} \in \mathbb{R}^m$, we get a neural network $\mathcal{N}_w : \mathbb{R}^n \rightarrow \mathbb{R}^m$ described by equation 2.14 and parameterized by w , which includes the weight matrices and biases vectors of each layer. The number of layers in a neural network ranges from a couple of layers to several hundreds or even thousands of them.

$$\mathbf{y} = \mathcal{N}_w(\mathbf{x}) \quad (2.14)$$

Activation functions at the last layer govern the networks outputs. Typically, the *identity* function $\mathcal{A}(x) = x$ is used for unbounded regressions while the *sigmoid* function $\mathcal{A}(x) = 1/(1 + e^{-x})$ is used for bounded outputs. It has been proven that those highly parameterized networks can approximate arbitrarily complex functions after adjustment of their parameters, provided that they have enough neurons (Hornik et al., 1990).

The process of adjusting the model parameters, referred to as *training*, generally starts from random values and relies on stochastic gradient descent optimization of a loss function $\mathcal{L}$ computed on *batches* of pairs $(\mathbf{y}, \hat{\mathbf{y}})$ to gradually update the parameters; where $\hat{\mathbf{y}}$ is the prediction provided by the neural network for input $\mathbf{x}$ and $\mathbf{y}$ being the expected output. Hence, at each iteration, w_i , the parameters at iteration i , are updated with the operation described with equation 2.15 where μ , the *learning rate*, is an hyper-parameter. Through training, neurons eventually get specialized and their arrangement enables the extraction of complex patterns.

$$w_{i+1} \leftarrow w_i - \mu \nabla_{w_i} \mathcal{L}(\mathbf{y}, \mathcal{N}_w(\mathbf{x})) \quad (2.15)$$

Once trained, the model can be use to make predictions on unseen data.

2.3. Convolutional Neural Networks

This process is called *inference*.

2.3.2 Neural networks for image processing

The ability of neural networks to extract features from raw signals makes them well suited to directly process the pixel intensities from images. Color images can be represented digitally by triplets of two dimensional arrays called *channels*, each containing intensities of one of the three primitive colors: red, green and blue (RGB). Hence, networks can be designed to expect inputs of dimension $h \times w \times 3$ where h and w are the image height and width respectively.

There exist many variations from the canonical neural networks described in the previous section, some of which are particularly adapted to image processing. Among them, the *convolutional layer* (LeCun et al., 1998) is an evolution of the dense layer described by equation 2.13 that preserve the spatial structure embodied within the images through shared weights. Similar to a classical discrete convolution, a two-dimensional n -channels input gets convoluted with m kernels of $k \times k \times n$ trainable weights. Hence, acting like a small dense layer $\mathcal{K} : \mathbb{R}^{k \times k \times n} \rightarrow \mathbb{R}^m$ sled across the two spatial dimensions, the operation extracts m features from $k \times k$ neighborhoods, creating m -channels outputs called *feature maps*. Input channels are generally padded with zeros to preserve the original spatial resolution.

Several other layers adapted to image processing and preserving spatial structure are worth mentioning. Similar to the convolutional layers which combine information from neighboring pixels using trainable filters, the *pooling layers* aggregate spatial information using an aggregation operator (typically the maximum function) over a $k \times k$ sliding window (Boureau et al., 2010); Instead of aggregating features with small fixed-sized windows, the *spatial pyramid pooling* (SPP) layers aggregate features over large areas defined by fixed-sized grids over the whole image (He et al., 2015); The *upsampling layers* use interpolation to upscale feature maps, thereby increasing spatial resolution (Shelhamer et al., 2017); And finally, the *pixel shuffling layers* which, unlike the name suggests, doesn't really shuffle pixels but rearrange pixels from $k \times k \times n$ channels to build n channels enlarged by a factor k in each spatial dimension (Shi et al., 2016). In addition, *batch normalization layers*, which stabilizes training by normalizing the feature maps (Ioffe and Szegedy, 2015), are generally added between trainable layers.

Stacking these layers such as the output feature maps of one become the

Chapter 2. Background

input of the following creates fully convolutional neural networks, through which the spatial structure of the input is preserved. The more layers the network has, the greater contextual information its output features will contain. This context is called *receptive field*.

CNNs often follow a two stages architecture: a large group of generic layers called the *backbone* are responsible for the features extraction from the input image, followed by one or multiple small groups of layers called *heads* which are task specific. Heads whose outputs don't have spatial dimensions are generally implemented with dense layers. The conversion from a spatial feature map to a vector is generally done by flattening the spatial dimensions.

2.3.3 Neural networks supervision

CNNs have demonstrated remarkable capabilities of extracting visual patterns from images, enabling them to be applied to a wide range of computer vision tasks. Learning these complex tasks is generally reached by breaking them down into simpler sub-tasks easier to supervise. For instance, semantic segmentation is a pixel-wise classification while object detection is a regression of objects bounding box coordinates (Girshick et al., 2014).

As described before, the last activation function governs the networks outputs. While regression tasks mostly rely on the identity or sigmoid activation functions applied to scalar outputs, classification tasks are generally formulated by predicting the likelihood distribution of belonging to each class, with a vector that sums to 1. This can be achieved using the *softmax* activation function characterized by the following equation.

$$\mathcal{A}(\mathbf{x}) = \frac{e^{x_i}}{\sum_j e^{x_j}} \quad (2.16)$$

The choice of loss function plays a critical role for networks supervision. There's a multitude of different loss functions carefully crafted for specific task formulations.

For regression tasks, the most simple one is probably the mean squared error (MSE) between the prediction $\hat{y}$ and the expected output y . However, its sensitivity to outliers or non-uniformly scaled outputs makes it inappropriate for many tasks. For this reason, the Huber loss (Huber, 1964) characterized by equation 2.17 which is linear for errors beyond a certain

2.3. Convolutional Neural Networks

threshold δ is generally preferred.

$$\mathcal{L}(\hat{y}, y) = \begin{cases} \frac{1}{2}(y - \hat{y})^2 & \text{for } |y - \hat{y}| \leq \delta \\ \delta(|y - \hat{y}| - \frac{1}{2}\delta) & \text{otherwise} \end{cases} \quad (2.17)$$

For classification tasks, the cross-entropy loss which measures the dissimilarity between the predicted likelihood distribution $\hat{\mathbf{y}}$ and the true distribution $\mathbf{y}$ is generally used. It is described by the following equation.

$$\mathcal{L}(\hat{\mathbf{y}}, \mathbf{y}) = - \sum_i y_i \log(\hat{y}_i) \quad (2.18)$$

2.3.4 Challenges of training neural networks

Training these highly parameterized models can be arduous due to a number of reasons.

At the core of many challenges lies the large number of parameters, essential for the representation of complex patterns. Tuning these parameters such that the network becomes able to extract relevant features and thereby generalize to unseen data requires a dataset large enough to capture a representative sample of all possible inputs. Collecting such datasets is challenging and generally requires time consuming manual labelling. Nonetheless, thanks to the versatility of neural networks, it is possible to *pre-train* the backbones on generic tasks for which very large datasets are available and *fine-tune* them for the specific tasks (Yosinski et al.; Bengio, 2009). Also, a common approach for increasing data diversity with a given dataset is to use *data-augmentation* which consists in creating multiple alterations of a single dataset sample to artificially generate new samples (Chatfield et al., 2014). For image processing, random cropping and scaling or color jittering of the original image is often used.

On top of this, the gradient-descent based optimization algorithms has only a limited view on the loss function complex landscape (Hecht-Nielsen, 1989). For this reason, a great number of iterations is required for adjustment of the network parameters. We call *epoch* cycling through the dataset once. Many epochs are generally required to train neural networks, and this usually translates in hours, days or even weeks of continuous processing on modern hardware.

Speaking of which, the implementation of neural networks is also subject to multiple hardware constraints. Today, the devices they are trained

Chapter 2. Background

on have limited memory available. Therefore, the size required to store the neural network along with the intermediate results for the optimization algorithm limits the amount of data that can be fed per iteration. This usually leads to a trade-off between the images crop size and the batch size. In addition, some functions used in neural networks can be numerically unstable or may overflow numerical representations. For this reason, several tricks like gradient clipping (Zhang et al., 2020) or batch normalization (Ioffe and Szegedy, 2015) must be applied to ensure reliable and efficient training and inference.

The gradient-descent algorithm uses the partial derivative of the loss function with respect to each neuron's weight to update them. However, that gradient tends to become exceedingly small for neurons in early layers of the network. This effect, called *vanishing gradient*, can strongly hinder training and is particularly noticeable for deep networks with activation functions whose derivative is close to 0 for large inputs (He et al., 2016). There exist several regularization techniques to mitigate this effect, like the use of ReLU activation functions (Nair and Hinton), the batch normalization layers (Ioffe and Szegedy, 2015), or specific weights initialization strategies (Glorot and Bengio, 2010).

Building the network is also a challenging endeavour. It should be deep enough to extract complex patterns (Krizhevsky et al.), but larger networks take longer to train and are more subject to overfitting the training data, leading to poorer generalization capabilities. Besides, defining the network internal structure is a challenging task itself. For this reasons, it is common to use well established backbones rather than crafting one for each application. Additionally, hyper-parameter tuning, the search for optimal meta parameters which characterize the network's structure, the loss function and the training parameters like the learning rate, can be a daunting task that increases the training time tenfold.

Also, the stochastic nature of the optimization algorithm leads to different results for different trainings. This observation is exacerbated for small datasets. For this reason, it is important to run multiple trainings with different initial conditions before drawing conclusions, which, again, increases the training time tenfold.

Finally, complex tasks often require multiple different outputs for a given input. While using multiple heads with a shared backbone enables simultaneous predictions with efficient computational complexity, learning multiple sub-tasks can be challenging. Training them jointly requires careful balancing of the different losses associated to each sub-task, and training

2.3. Convolutional Neural Networks

them successively can lead to catastrophic forgetting of the tasks learned beforehand (Kirkpatrick et al., 2017). Nonetheless, it has been shown that joint learning can improve overall performance of the individual tasks (Kendall and Gal, 2017), but it is however not necessarily guaranteed.

To sum up, the amount and diversity of challenges related to the training of neural networks generally makes the task laborious and fastidious. For this reason, it is recommended to use off-the-shelf models, preferably pre-trained, instead of building and training them from scratch.

2.3.5 Popular network architectures for computer vision

When addressing new tasks, it is recommended to use existing neural network backbones. This section introduces popular CNN architectures used in this thesis.

VGG The VGG family of CNNs was designed for image classification and object detection tasks by the "Visual Geometry Group" at Oxford University (hence its name), and played a strong role in popularizing the use of CNNs for computer vision (Simonyan and Zisserman, 2015). The most well-known architectures are VGG16 and VGG19 which contain 16 and 19 layers respectively. They are characterized by their simple architecture with a succession of convolutional and max-pooling layers, followed by few fully connected layers.

ResNet The ResNet name comes from "Residual Network" which is an architecture of CNNs designed to address the vanishing gradient problem (He et al., 2016). The key innovation of ResNet is the introduction of skip connections allowing information to bypass one or more layers. Hence, instead of trying to learn the desired output directly, the skipped layers focus on learning the residual which is the difference between the layers input and output. This approach enabled the training of very deep networks with hundreds or even thousands of layers. ResNet architectures come in various depths, from relatively shallow networks with a few layers to very deep networks with hundreds of layers. ResNet has been highly successful in various computer vision tasks, such as image classification, object detection and image segmentation, and has become a fundamental architecture for deep learning in computer vision.

Chapter 2. Background

ICNet This architecture was designed for real-time semantic segmentation of high-resolution images (Zhao et al., 2018). ICNet stands for "Image Cascade Network" and uses cascades of multiple network pathways that operate at different resolutions. The highest-resolution pathway, captures fine-grained details in the image by processing the input image at its original resolution. The middle-resolution pathway, is based on the ResNet architecture and captures more global contextual information by reducing the resolution of the input image. The lowest-resolution pathway operates with an input at a quarter of the original image resolution. It provides large contextual information at very low computational complexity. These pathways are fused together after appropriate upsampling to create an accurate full resolution segmentation map.

2.4 Ballistic motion models for basketballs

Objects in free fall are subject to a limited number of forces, which enable the calculation of the equations governing their motion. This section introduces the mathematics related to ballistic trajectories and provides the equations of motion for different scenarios.

2.4.1 Motion under gravity alone

Newton's laws of universal gravitation reveals that small bodies in free fall near the Earth's surface are attracted towards it by a force called *weight* proportional to their mass m and with a constant acceleration called g . Its magnitude $g \approx 9.81 \text{ m/s}^2$ varies by less than 1% for projectiles within 30km of the Earth's surface, and with similar variations for changes in latitude (de Mestre, 1990).

Therefore, the simplest approximation for a projectile's motion is to consider that the only force acting on it after it is launched is its weight, and consider a small neighbourhood where the earth's curvature is ignored. If $\mathbf{p}(t)$ is the body's position parameterized by the time t , its equation of motion is the following, where $\mathbf{g}$ is a vector vertical with respect to the Earth surface (considered flat), pointing down, with a magnitude of g .

$$\ddot{\mathbf{p}}(t) = \mathbf{g} \quad (2.19)$$

2.4. Ballistic motion models for basketballs

For an initial speed v_0 , integration of this differential equation with respect to time yields the following relationship where $\dot{\mathbf{p}}$ is the velocity vector at any time t .

$$\dot{\mathbf{p}}(t) = \mathbf{v}_0 + \mathbf{g}t \quad (2.20)$$

Note that this result indicates that the horizontal component of the velocity of a projectile in a vacuum is independent of t and is therefore a constant, equal to the initial horizontal speed.

When integrating one step further, we see that the projectile motion is described by a parabola with the following relationship where $\mathbf{p}_0$ is the initial position of the projectile at time $t := 0$.

$$\mathbf{p}(t) = \mathbf{p}_0 + \mathbf{v}_0 t + \mathbf{g} \frac{t^2}{2} \quad (2.21)$$

This solution relates strictly to a projectile traveling under the influence of constant gravity in a vacuum.

2.4.2 Motion with air friction

When the projectile moves through any fluid medium, other forces are present due to the slowing down influence of that medium's particles. We focus on the two predominant forces impacting ballistic trajectories of basketballs.

Drag The main effect of the medium is a force called *drag* which has a direction opposite to the projectile's velocity. In the air, the drag force $\mathbf{D}$ is generally assumed to be directly proportional to $\rho(z)$ the density of air at height z expressed in kg/m^3 , A the cross-sectional area of the projectile in m^2 , and the projectile's velocity squared. The velocity, expressed in m/s , is of magnitude v in the direction defined by the unit vector $\hat{\mathbf{v}}$.

$$\mathbf{D} = -\frac{1}{2} \rho(z) A v^2 c_D \hat{\mathbf{v}} \quad (2.22)$$

The drag coefficient c_D is non-dimensional and may be a function of the projectile's speed, the air temperature or the projectile roughness. However, for many projectiles travelling at speeds below 0.6 the medium's speed of sound, its value can be taken as constant (de Mestre, 1990).

Chapter 2. Background

Lift Considering a spherical projectile, when a rotation around its center of mass is applied, an asymmetry appears in the air flow moving past it. Such an asymmetry is manifested by an additional force called *lift* that acts on the sphere causing it to swerve. The lift $\mathbf{L}$ adds to the drag, it is orthogonal to the rotation (or *spin*) axis and this direction is defined by the unit vector $\hat{\mathbf{n}}$.

$$\mathbf{L} = \frac{1}{2} \rho(z) A v^2 c_L \hat{\mathbf{n}} \quad (2.23)$$

The lift coefficient c_L is proportional to the projectile spin rate.

Equation of motion By summing the forces active on the projectile during its trajectory and using Newton's second law, we have the following relationship where $k_0 = \frac{\rho A c_D}{2m}$ and $l_0 = \frac{\rho A c_L}{2m}$.

$$m \ddot{\mathbf{p}} = m \mathbf{g} - \overbrace{m k_0 v^2 \hat{\mathbf{v}}}^{\text{D}} + \overbrace{m l_0 v^2 \hat{\mathbf{n}}}^{\text{L}}, \quad (2.24)$$

which leads to the differential equation that describe the motion of the projectile's center of gravity.

$$\ddot{\mathbf{p}}(t) = \mathbf{g} - k_0 v(t)^2 \hat{\mathbf{v}} + l_0 v(t)^2 \hat{\mathbf{n}} \quad (2.25)$$

When this equation of motion is separated into Cartesian components, the result is a system of coupled highly non-linear differential equations which cannot be solved analytically.

2.4.3 Specific case of horizontal spin axis

When the spin axis is horizontal, $\hat{\mathbf{n}}$ and $\hat{\mathbf{v}}$ are co-planar with $\mathbf{g}$ and the projectile remains in the vertical plane containing the initial velocity vector. This case is illustrated in figure 2.2. The decomposition of equation 2.25 in Cartesian coordinates has a simple form that we can rewrite with a first-order system of differential equations using $\dot{\mathbf{p}} = \mathbf{v}$ and with α being the trajectory plane angle relative to $\hat{\mathbf{x}}$, and $v(t) := \sqrt{v_x(t)^2 + v_y(t)^2 + v_z(t)^2}$.

2.4. Ballistic motion models for basketballs

$$\begin{cases} \dot{\mathbf{p}}_x(t) = \mathbf{v}_x(t) \\ \dot{\mathbf{p}}_y(t) = \mathbf{v}_y(t) \\ \dot{\mathbf{p}}_z(t) = \mathbf{v}_z(t) \\ \dot{\mathbf{v}}_x(t) = -k_0 \mathbf{v}(t) \mathbf{v}_x(t) + l_0 \mathbf{v}(t) \mathbf{v}_x(t) \cos \alpha \\ \dot{\mathbf{v}}_y(t) = -k_0 \mathbf{v}(t) \mathbf{v}_y(t) + l_0 \mathbf{v}(t) \mathbf{v}_y(t) \sin \alpha \\ \dot{\mathbf{v}}_z(t) = -g - k_0 \mathbf{v}(t) \mathbf{v}_z(t) - l_0 \mathbf{v}(t) \mathbf{v}_x(t) \cos \alpha - l_0 \mathbf{v}(t) \mathbf{v}_y(t) \sin \alpha \end{cases} \quad (2.26)$$

This system of equations is still highly coupled and non-linear. For this reason its integration requires numerical methods.

2.4.4 Application to Basketballs

Assuming still air, constant air density, a spherical projectile, horizontal spin axis and neglecting earth curvature and rotation, the model described by the system of equations 2.26 provides a very good approximation of basketball long passes and shots in indoor games. Good parameter values are $\rho = 1.268 \text{ kg/m}^3$, $A = 0.0446 \text{ m}^2$, $m = 0.62 \text{ kg}$ and $c_D = 0.2$. The lift coefficient can go up to $c_L = 0.1$ which corresponds to 10 rotations per seconds (de Mestre, 1990).

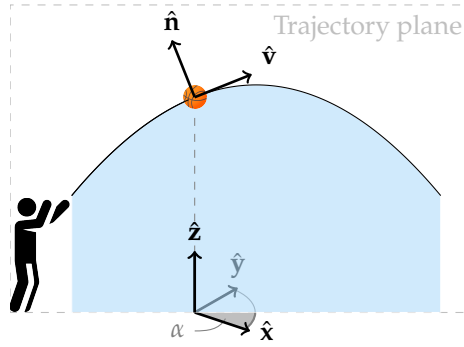
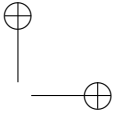
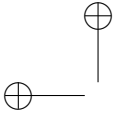
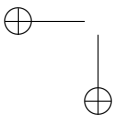


Figure 2.2: **Ballistic trajectory with horizontal spin axis.** Ballistic trajectory when $\hat{\mathbf{n}}$ and $\hat{\mathbf{v}}$ are co-planar with $\mathbf{g}$. In this case, the trajectory lies in a vertical plane and we denote α its angle with respect to the Cartesian coordinates system $(\hat{\mathbf{x}}, \hat{\mathbf{y}}, \hat{\mathbf{z}})$ where $\hat{\mathbf{z}}$ is vertical.

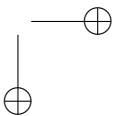


—

—



|



Chapter 3

Ball 3D annotations from a single calibrated viewpoint

Studying and evaluating methods for object 3D localization requires data with high quality annotations, and the training of modern learning-based models requires a lot of them. Besides, this computer-vision task relies on images with known calibration. However, not many public datasets of annotated ball 3D position with calibrated images exist, and they suffer both from quality and diversity.

Thanks to our strong collaboration with Keemotion, we got access to high-resolution calibrated images from their automated capture system installed in many professional basketball venues, and we invested a lot of efforts in designing and studying efficient methods addressing objects 3D annotation from a single calibrated viewpoint.

This chapter presents and evaluates a novel method for annotation of ball 3D position on individual calibrated images. Leveraging calibration data, by annotating the ball center and its vertical projection on the ground floor, it is possible to recover its 3D position. Unlike existing annotation methods, ours doesn't require expensive multiple viewpoint setups with synchronized cameras, which enables annotation in many practical scenarios when the acquisition system is not within the stakeholder's control, as long as camera calibration is known or can be inferred from the images. Moreover, our method is applicable to any object, making it usable to other domains beyond sport, as long as the scene provides cues of the object's height.

Chapter 3. Ball 3D annotations from a single calibrated viewpoint

We used our method to annotate the Keemotion images and created two datasets with annotated ball 3D position. For the benefit of the research community, we made them publicly available, establishing a solid benchmark for the study of ball 3D localization.

This chapter is organized as follow. The first section introduces the existing datasets for ball 3D localization. The second section presents the different balls datasets that were captured during this thesis and the third section describes our image-based ball 3D annotation method used to annotate the datasets. In addition, we study different strategies for ballistic 3D annotations in section four and a summary of the different datasets used in this thesis is provided in table 3.4, page 41, in section five.

3.1 Existing datasets

Not many public datasets with annotated ball 3D position exist. Most past works rely on private datasets, usually captured in one or few different environments (Cheng et al., 2018; Lampert and Peters, 2012; Maksai et al., 2016; Pingali et al., 2000; Ren et al., 2008; Silva et al., 2011; Sköld, 2015). To our knowledge, there are only two datasets with annotated 3D ball position publicly available:

ISSIA-dataset (D’Orazio et al., 2009) is a 2 minutes soccer sequence captured from six 1920×1080 synchronized cameras, at 25 frames per second. Ball position was annotated with a semi automated process on the six streams and ball 3D position can be retrieved using known calibration. However, as it is designed for player tracking, the ball is often out of the field of view when flying.

APIDIS-dataset (De Vleeschouwer et al., 2008) is a 15 minutes women basketball sequence captured by seven un-synchronized cameras at about 22 frames per second. The venue has rather weak lighting, and to avoid motion blur, the exposition time was kept low, resulting in dark and poorly contrasted images. Ball center was manually labelled by Parisot and De Vleeschouwer (2019) on the seven streams for 3 minutes of the game and a pseudo ground-truth of 3D position was built by triangulation and time interpolation on re-sampled streams of 800×600 pixels at 25 frames per second (Parisot and De Vleeschouwer, 2019). The difference with the original acquisition frame rate creates alignment issues between the images and the corresponding ball 3D annotations.

The soccer context does not offer the complexity encountered in basket-

3.2. Dataset acquisition

ball and the 2 minutes sequence from the *ISSIA-dataset* lacks in diversity. In this thesis, we focus on basketball, where balls are frequently occluded, often poorly contrasted, and suffer from noise and motion blur due to indoor lighting conditions. This makes it a challenging sport to address the ball 3D localization task. Despite a scene captured by multiple viewpoints, the *APIDIS-dataset* does not offer accurate ball 3D position. Indeed, due to the lack of synchronization between the multiple image streams, we observed dozen of pixels differences between the annotated 3D position projected in image space and the visual appearance of the ball. Additionally the dataset does not provide any information about ball visibility, in particular when it is hidden by the players as often encountered in basketball scenes.

3.2 Dataset acquisition

In collaboration with Keemotion (Delannay, 2012), world leader in autonomous capture of basketball games, we addressed the lack of high-quality and high-diversity ball 3D localization datasets. We used their acquisition systems during live productions to capture high quality images of professional games in multiple different venues.

The Keemotion camera setups are composed of an array of two or three fixed high-resolution cameras covering the whole basketball court from a *single viewpoint*, as illustrated in figure 3.1. The cameras are synchronized with an external trigger enabling a synchronous capture of images to construct a panorama of the scene. Leveraging known camera calibration to stitch images together, a local server located in each venue processes the multiple high-resolution streams to produce a narrow broadcast-quality footage that follows the game action.

To support the development of the Keemotion framing algorithm, it was decided to capture images from the high-resolution raw streams on which it operates. However, due to the limited resources available locally, not all servers can sustain a continuous recording while producing live events. For this reason, we created two complementary datasets. The first dataset is a collection of independent *raw-instants*, sets of raw high-resolution images captured from all the fixed cameras at the same instant. It features a wide variety of view angles, team jerseys and lighting conditions. The second dataset is composed of a limited number of several minutes long *raw-sequences* captured by the fixed cameras in a couple of different venues.

Chapter 3. Ball 3D annotations from a single calibrated viewpoint



Figure 3.1: **Panoramic view from the camera setup.** Two high resolution images captured at the same instant from a Keemotion Automated Camera System with a two cameras setup.

3.2.1 Basketball Instants Dataset

The acquisition of the *raw-instants* dataset was carried out over 37 different professional games in 15 different French arenas, each identified with a unique label, and resulted in 398 in-game basketball scenes featuring a large variety of game configurations and lighting conditions. With the

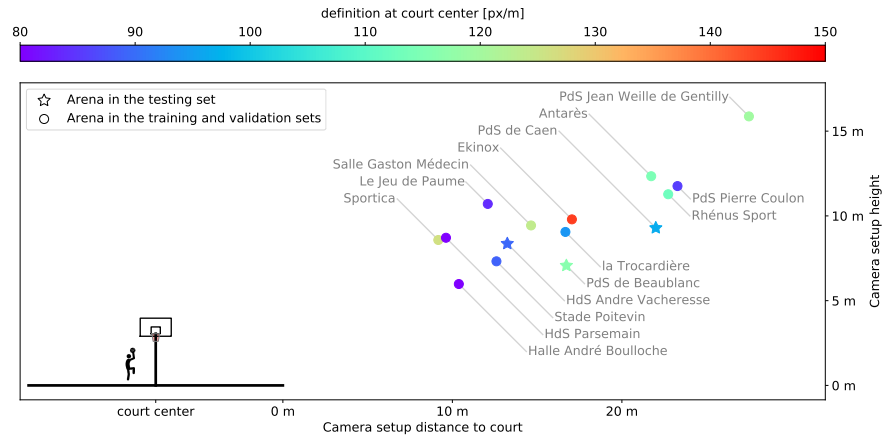


Figure 3.2: **Variety of camera configurations.** This cross section shows the camera setup height from the ground and distance to the court for the different arenas in which images were acquired. The camera definition depends on camera resolution, sensor size, lens focal length and camera setup distance to the court.

3.2. Dataset acquisition

cameras' resolution ranging from 2Mpx to 5Mpx, the resulting images have a definition varying between 80 and 150 *pixels-per-meters*, which describes the pixel density covering the scene and depends on camera resolution, sensor size, lens focal-length and distance to the court. Figure 3.2 depicts a cross section of a basketball court where the camera setup height and distance to the court is shown for each arena, along with the pixels-per-meter definition at the court center.

Two consecutive images from the fixed cameras were actually captured for each instants. Depending on the acquisition frame rate in the arena, the following frames have a delay of 33 ms or 40 ms. The pairs of successive images enable gaining cues of motion in the scene.

To facilitate evaluation of different methods and measure performance on unseen arenas, we define a testing set containing all images from arenas KS-FR-CAEN, KS-FR-LIMOGES and KS-FR-ROANNE. A mapping between arena labels and arena names is given in Table 3.1 and provides the amount of instants captured in each arena. This dataset was made freely available under the name *DeepSport-dataset*.

Arena label	Arena name (City)	Number of instants
KS-FR-STCHAMOND	Halle André Boullouche (Saint-Chamond)	12
KS-FR-FOS	HdS Parsemain (Fos-sur-Mer)	23
KS-FR-STRASBOURG	Rhénus Sport (Strasbourg)	8
KS-FR-VICHY	PdS Pierre Coulon (Vichy)	9
KS-FR-NANTES	La Trocardière (Nantes)	20
KS-FR-BOURGEB	Ekinox (Bourg-en-Bresse)	12
KS-FR-GRAVELINES	Sportica (Gravelines)	129
KS-FR-MONACO	Salle Gaston Médecin (Monaco)	9
KS-FR-POITIERS	Stade Poitevin (Poitiers)	5
KS-FR-NANCY	PdS Jean Weille de Gentilly (Nancy)	40
KS-FR-LEMANS	Antarès (Le Mans)	16
KS-FR-BLOIS	Le Jeu de Paume (Blois)	39
KS-FR-CAEN	PdS de Caen (Caen)	31
KS-FR-ROANNE	HdS Andre Vacheresse (Roanne)	3
KS-FR-LIMOGES	PdS de Beaublanc (Limoges)	8

Table 3.1: *DeepSport-dataset diversity*. The *DeepSport-dataset* was captured in 15 different arenas and 3 of them are kept for the testing set. It features a variety of angle of views, distance to the court and image resolution, and is freely available for research purpose.

Chapter 3. Ball 3D annotations from a single calibrated viewpoint

An additional set of 1470 instants was captured in 34 other arenas with similar characteristics but is not part of the public dataset. We call it the *DeepSport-dataset-extended* and it was not used unless specifically stated otherwise.

3.2.2 Basketball Sequences Dataset

Several *raw-sequences* from different arenas were recorded during top league matches. To enable the recording on the servers storage while delivering live productions, the recording quality was altered by reducing either the frame rate or the resolution. Table 3.2 provides a summary of the captured sequences. The two shorter sequences were recorded first and suffer from inaccurate timestamps. The longer sequences were recorded later and don't have the timestamp problem but were captured with a reduced image resolution.

Ballistic trajectories were manually segmented from the video sequences by labelling the frames where the ball is in free fall during a minimum of 250 milliseconds, ruling out the dribbles and short passes.

The trajectories extracted from the initial two sequences were made publicly available (Van Zandycke and De Vleeschouwer, 2022) and are referred to as the *Ballistic-dataset*. This dataset is composed of 35 ballistic trajectories with a total of 233 images and was used only for evaluation purposes in this thesis. The other sequences are referred to as the *Sequences-dataset* and were only used in chapter 3 and 6.

3.2.3 Practical constraints

Capturing professional sport datasets concurrently with the automated production on the Keemotion servers came with unique challenges. A 2019 standard Keemotion system typically produces three 25 frames-per-second continuous streams of 5 megapixel images, which corresponds to a staggering 375 megabytes per seconds (MB/s) of raw data. As a comparison, the write speed of typical affordable hard-drive spinning disk is generally limited to 150 MB/s and the more expensive solid-state-drive storage generally reach 500 MB/s. To avoid interfering with the ongoing live productions, the *raw-instants* were captured every 10 minutes and the *raw-sequences*, resampled and compressed, were captured in arenas equipped with solid state drive storage systems. Also, data were transferred overnight, when the live productions were over.

3.2. Dataset acquisition

Arena label	cameras	resolution	duration (minutes)	frame rate
KS-FR-STRASBOURG	2	2336 × 1752	3	~ 10
KS-FR-GRAVELINES	2	2336 × 1752	3	~ 6
KS-FR-BOURGEB	3	1180 × 996	21	25
KS-FR-GRAVELINES	2	1168 × 876	13	25
KS-FR-BLOIS	3	1180 × 996	20	25
KS-FR-DIJON	3	1180 × 996	19	25
KS-FR-EVREUX	3	1180 × 996	8	25
KS-FR-ROANNE	3	1180 × 996	18	25
KS-FR-LILLE	3	1180 × 996	13	25

Table 3.2: **Summary of the raw-sequences recorded with the Keemotion Automated Capture System.** Ballistic trajectories were annotated in the four sequences in **bold** and our public *Ballistic-dataset* was built from the two first sequences.

The multiple acquisition campaigns generated hundreds of gigabytes of data, both for the *raw-instants* datasets, before off-game scenes were filtered, and the *raw-sequences* dataset, which has a limited number of very large files. To upload, store, maintain and update these datasets, we designed and deployed a specific cloud infrastructure. It consists of a light database where items metadata and annotations are stored, and a file system like storage for images and video files. This structure enables fast filtering and efficient retrieval of dataset items.

Finally, working with temporal fitting of ballistic motion models requires knowing the precise acquisition timestamp of each frame. This was made possible using the professional cameras from the Keemotion capture system, which provide millisecond precision timestamps.

3.2.4 Open science with open data

Thanks to the partnership between Keemotion and the *Ligue Nationale de Basketball* (LNB) who owns the images right, we were authorized to make the *DeepSport-dataset* and *Ballistic-dataset* public. They are now freely available for research purposes (Van Zandycke et al., 2022; Van Zandycke and De Vleeschouwer, 2022), and not only enable the reproduction of most of the experiments conducted in this thesis, but provide a solid benchmark for studying ball 3D localization.

3.3 Ball 3D annotation from a single viewpoint

With the single viewpoint camera setups from Keemotion, we cannot use the multi-view approach generally adopted (Ren et al., 2008; D’Orazio et al., 2009; Sköld, 2015; Parisot and De Vleeschouwer, 2019). The annotation of ball 3D position is therefore a challenge. In this section, we propose a novel image-based method that consists in annotating the ball center as well as its vertical projection on the ground floor, and leverage known camera calibration to get the 3D position. We evaluate our method on our *Ballistic-dataset* and compare it against a simpler approach that consists in annotating the ball diameter in the image space.

3.3.1 Annotation method

To annotate the ball 3D position from a single image, we propose to annotate its center and its vertical projection on the ground floor in the image space, and leverage known camera calibration to retrieve the 3D world coordinates. This approach makes particularly sense when the ball is near the players, as often encountered in basketball scenes, as the annotator can localize it relative to their pose. When it is not held nor dribbled, the scene configuration generally provides cues of the shooter’s and catcher’s feet positions, which define a line on which the ball projection is expected to be.

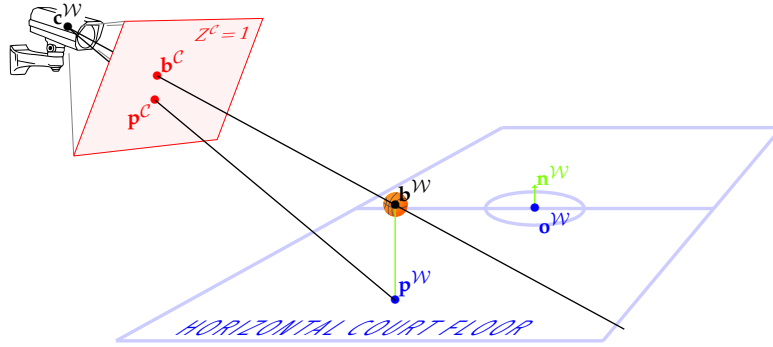


Figure 3.3: **Diagram of our annotation method.** Our method converts image annotation of ball center and its ground projection to 3D coordinates leveraging known calibration data.¹

¹Icons from flaticon.com

3.3. Ball 3D annotation from a single viewpoint

For a camera defined by the camera matrix K , located in $\mathbf{c}^{\mathcal{W}} = -R^T \mathbf{t}^{\mathcal{C}}$ where R is the rotation matrix associated to its orientation and $\mathbf{t}^{\mathcal{C}}$ the translation vector associated to its position, and whose lens distortion is modeled by $\mathcal{R}$, the projection on the plane $z^{\mathcal{C}} = 1$ of the annotated ball center $\mathbf{b}^{\mathcal{I}}$ and vertical projection $\mathbf{p}^{\mathcal{I}}$ are given by the following relationships:

$$\mathbf{b}^{\mathcal{C}} = K^{-1} \cdot \mathcal{R}(\mathbf{b}^{\mathcal{I}}) \quad (3.1)$$

$$\mathbf{p}^{\mathcal{C}} = K^{-1} \cdot \mathcal{R}(\mathbf{p}^{\mathcal{I}}) \quad (3.2)$$

The projection point expressed in world coordinates $\mathbf{p}^{\mathcal{W}}$ is found by intersection of the line joining $\mathbf{c}^{\mathcal{W}}$ and $\mathbf{p}^{\mathcal{C}}$, with the court plane defined by a normal vector $\mathbf{n}^{\mathcal{W}}$ and a point $\mathbf{o}^{\mathcal{W}}$. Hence, the ball 3D localization is the intersection of the line joining $\mathbf{c}^{\mathcal{W}}$ and $\mathbf{b}^{\mathcal{C}}$, with the vertical line going through $\mathbf{p}^{\mathcal{W}}$. This process is illustrated in figure 3.3.

In practice, after annotation of the ball center in the image space $\mathbf{b}^{\mathcal{I}}$, the annotation tool, illustrated in figure 3.4, leverages known calibration data to constrain the annotation of the ball vertical projection $\mathbf{p}^{\mathcal{I}}$ on a vertical line in the real world coordinates system. In addition, a cross aligned with the court geometry is displayed on the ground floor at the projection point. This helps understand the distance between the ball and players' feet along the court length and width.

3.3.2 Evaluation procedure

We present here the methodology used to evaluate our image-based ball 3D annotation method. We also propose a baseline, simpler to implement, against which our method will be evaluated.

Baseline To annotate how far from the camera the ball is, a natural approach is to annotate its diameter in the image space. Knowledge of the real ball diameter can then be used to convert the image annotations into 3D coordinates. Hence, given two diametrically opposed points $\mathbf{e}_-^{\mathcal{I}}$ and $\mathbf{e}_+^{\mathcal{I}}$ annotated on the edge of the ball disc in the image space and the true ball diameter ϕ , the 3D localization of the ball $\mathbf{b}^{\mathcal{W}}$ is given by the following relationship. It uses the ratio between the diameter $\|\mathbf{e}_-^{\mathcal{C}} - \mathbf{e}_+^{\mathcal{C}}\|$ and the real

Chapter 3. Ball 3D annotations from a single calibrated viewpoint



Figure 3.4: **Interface for ball 3D annotation on calibrated images.** With our annotation tool, the ball center is annotated first and its vertical projection on the court floor is annotated second. Camera calibration is used to constrain the projection in a vertical line, and a cross aligned with the field geometry is drawn on the ground to help the annotator locate the ball in the scene.

ball diameter ϕ to scale the middle point $(\mathbf{e}_-^c + \mathbf{e}_+^c)/2$.

$$\mathbf{b}^w = R^T \cdot \frac{\phi}{2} \cdot \frac{\mathbf{e}_+^c + \mathbf{e}_-^c}{\|\mathbf{e}_+^c - \mathbf{e}_-^c\|} + \mathbf{c}^w \quad (3.3)$$

where $\mathbf{e}_\pm^c$ are the projections on the plane $z^c = 1$ of the annotated points $\mathbf{e}_\pm^I$ after consideration of the lens distortion.

$$\mathbf{e}_\pm^c = K^{-1} \cdot \mathcal{R}(\mathbf{e}_\pm^I) \quad (3.4)$$

This approach relies on the assumption that the ball is far enough to make the hypothesis that those two diametrically opposed points are visible from the camera, which is reasonable in our scenario where the ball is

3.3. Ball 3D annotation from a single viewpoint

23.8 cm large and dozens of meters away from the cameras.

Dataset For evaluation of our annotation method, we use our *Ballistic-dataset* where balls were individually annotated with our projection method. A ballistic motion model was used to regularize and thereby improve the initial annotations. In practice, we used the second degree model described by equation 2.21, ignoring friction, ball spin and other effects. Given a trajectory composed of n annotated positions $\hat{\mathbf{p}}_i^{\mathcal{W}}$ at times t_i , with $i \in [1, n]$, the motion model parameters are obtained by the following relationship:

$$\mathbf{p}_0, \mathbf{v}_0 = \arg \min_{\mathbf{p}_0, \mathbf{v}_0} \sum_{i=1}^n \left(\mathbf{p}(t_i) - \hat{\mathbf{p}}_i^{\mathcal{W}} \right)^2 \quad (3.5)$$

The regularized positions are then computed by evaluating the model at $t = t_i$. Manual fine-tuning was ultimately performed to ensure the accuracy of the annotations, both for the ball center and its projection on the court floor. Trajectories that couldn't be modeled with the simple parabola model were discarded. The final dataset comprises 233 images whose high quality annotations can be considered as ground-truth to compare the distribution of errors resulting from the two annotation approaches.

Metrics To compare the two approaches, we use the absolute projection error. It measures the distance, expressed in meters, between the prediction and the ground-truth ball position, projected on the ground floor. In addition, the diameter error, in pixels, is also provided. For the projection approach, knowledge of real ball diameter is used to compute the diameter in the image space that corresponds to the annotated 3D position.

3.3.3 Experimental results

As shown in Figure 3.5, our ball center+projection approach is significantly better for ball 3D annotation in single calibrated viewpoints than the approach where the ball diameter is annotated. Indeed, with the diameter approach annotators have a tendency to underestimate the ball size, with a higher variance compared to annotations with the projection approach. It is important to realize that a few pixels diameter error can translate in meters in the 3D space, as we show in chapter 5. For this reason, annotation with the diameter approach would actually require sub-pixel precision.

Chapter 3. Ball 3D annotations from a single calibrated viewpoint

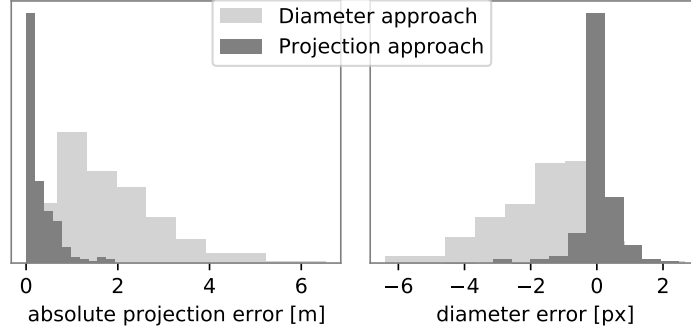


Figure 3.5: **Error distribution for two image-based ball 3D annotation approaches.** The approach that consists in annotating ball center and diameter shows higher error and a tendency towards underestimated diameters. The approach that consists in annotating ball center and vertical projection on the ground is more accurate and translates into a diameter error of less than 1 pixel.

3.3.4 Discussion

Although our annotation method offers a better 3D localization on our ball dataset compared to the diameter approach, it requires a more sophisticated annotation tool. Moreover, with our approach, annotators rely on cues from the surrounding scene to estimate ball projection, which is not always easy or even possible. For this reason, despite being theoretically applicable to any objects and therefore transferable to other domains beyond sport, our annotation method may not be adapted to any circumstances. We also observed that the complexity of estimating ball projection led to a longer training of the annotators.

Nevertheless, our approach only applies to single viewpoint annotations. When the addition of a second synchronized viewpoint is possible, it should be preferred, and in particular for the creation of an evaluation dataset. Indeed, with multiple viewpoints, it is possible to adopt a triangulation approach for annotation of 3D position of objects (D’Orazio et al., 2009), which is significantly easier to implement and produces more accurate annotations.

3.3.5 Conclusion

In this section, we propose a method for 3D ball annotation from a single image, leveraging camera calibration. To the best of our knowledge this method was never published and our evaluation shows that it achieves sub-pixel precision compared to an alternative approach. Moreover, the method doesn't make any assumption on the object whose 3D position is being annotated, making it relevant for other domains beyond sport.

3.4 Ballistic trajectories 3D annotation

In this section, we study multiple strategies for annotating ball 3D position during ballistic trajectories on multiple consecutive images from a single viewpoint. The strategies exploit a ballistic motion model, with the goal of minimizing annotation efforts.

3.4.1 Annotation strategies

When the ball follows a ballistic motion, its trajectory can be described by a motion model which is entirely defined by the initial conditions. To estimate those initial conditions we need to annotate at least as many data as there are degrees of freedom in the system. Providing more data transforms the resolution into an optimization problem where the distance between the annotated points and the model evaluated at the corresponding timestamps should be minimized. This helps reduce the sensitivity to annotation inaccuracies at the cost of more annotations.

Therefore, we propose to annotate a limited number n of images spread evenly along the trajectory, and fit a ballistic motion model to the initial annotations. The model is then evaluated at the images known acquisition timestamp to retrieve the final annotations for all images.

Motion models We consider three different models.

- **Parabola** The first model ignores air friction and is described by equation 2.21. The initial conditions are the initial position $\mathbf{p}_0$ and initial speed $\mathbf{v}_0$, at time t_0 .
- **Ballistic (no lift)** The second model is described by the system of first order differential equations 2.26 but assumes no ball spin ($l_0 = 0$). We add the drag coefficient c_D to the initial conditions.

Chapter 3. Ball 3D annotations from a single calibrated viewpoint

- **Ballistic (with lift)** The third model is also described by system 2.26 but involves a spin of constant intensity with an horizontal axis. We add the lift coefficient c_L to the initial conditions.

All three models assume a constant gravity vector $\mathbf{g}$ and ignore Earth rotation. In addition, the two models considering air friction assume a drag proportional to the square of ball speed, and still air with constant density.

Annotation approaches To provide initial annotations, we propose two different approaches.

- The first approach consists in annotating the balls center in the image space only and minimize the pixel distance between the initial annotations $\hat{\mathbf{p}}_i^{\mathcal{I}}$ and the model points $\mathbf{p}(t_i)$ projected in the image space using known camera calibration. Annotations using this approach don't need a sophisticated tool and are therefore very easy to produce. The distance to minimize is given by the following relationship, where $\mathcal{P}$ is the projection operation that transforms world coordinates into pixel coordinates.

$$\min \sum_{i=1}^n \left(\mathcal{P}(\mathbf{p}(t_i)) - \hat{\mathbf{p}}_i^{\mathcal{I}} \right)^2 \quad (3.6)$$

- The second approach leverages our image-based center+projection annotation tool to annotate the 3D ball positions $\hat{\mathbf{p}}_i^{\mathcal{W}}$ and minimize the distance in real world coordinates as represented by equation 3.7. Whereas the annotation of the ball vertical projection is easier to annotate for the two trajectory extremities, intermediate positions can be more challenging to annotate. For this reason, the first and last images should be annotated first, and a line joining their vertical projection on the ground floor should be displayed when annotating the intermediate images.

$$\min \sum_{i=1}^n \left(\mathbf{p}(t_i) - \hat{\mathbf{p}}_i^{\mathcal{W}} \right)^2 \quad (3.7)$$

Summary In summary we study two annotations approaches (2D annotations and 3D annotations), with three motion models fit over n annotations.

3.4.2 Evaluation procedure

We created a dedicated dataset to evaluate the different strategies. Metrics and implementation details are presented hereafter.

Dataset We use ballistic trajectories from the unpublished *Sequences-dataset* introduced in section 3.2.2 which offer more accurate timestamps than the *Ballistic-dataset*, which is critical for comparing the motion models evaluated at timestamps corresponding to the actual annotations. We annotated 21 ballistic trajectories with our annotation tool and trajectories were individually inspected to ensure high quality ground-truth.

Implementation With the exception of the parabola model computed from 3D coordinates, the minimization problem resulting from an excess of data over the number of degrees of freedom is non linear and requires numerical methods to be solved. We use the quasi-Newton method of Broyden, Fletcher, Goldfarb, and Shanno (BFGS) (Nocedal and Wright, 2006) with the default parameters from the implementation available with the Numpy Python library (Harris et al., 2020) to minimize the mean Euclidean distance characterized by equations 3.7 or 3.6, depending on the annotation approach.

Metric To evaluate the quality of the different motion models, we measure the 3D projected distance on the ground floor (in meters) between the positions resulting from their evaluation at the images timestamps and the associated annotated ground-truth. We evaluated the mean distance within each ballistic trajectory, and we report the average and standard deviation of these distances between the trajectories. In addition, we also computed the pixel distance between the model projected in the image space and the annotated ball center.

3.4.3 Experimental study

Table 3.3 reveals that initial annotations in the 2D space don't provide enough information to retrieve the 3D localization of the ball, even when all images are used to fit the motion models. Note that this result is not obvious. Indeed, even though the projection of a 3D parabola is a 2D parabola, the further away from the camera the ball is, the smaller pixel distances relate to world distances. This effect causes a variation of the perceived distance

Chapter 3. Ball 3D annotations from a single calibrated viewpoint

		Number n of initial annotations			
		2	3	4	all
2D	Parabola	-	.90 $\pm$.66	.96 $\pm$.79	.92 $\pm$.75
	Ballistic (no lift)	-	.89 $\pm$.69	.83 $\pm$.71	.93 $\pm$.74
	Ballistic (with lift)	-	.84 $\pm$.80	.97 $\pm$.85	1.26 $\pm$ 1.58
3D	Parabola	.05 $\pm$.03	.04 $\pm$.02	.03 $\pm$.02	.03 $\pm$.01
	Ballistic (no lift)	-	.03 $\pm$.02	.03 $\pm$.02	.02 $\pm$.01
	Ballistic (with lift)	-	.02 $\pm$.01	.02 $\pm$.01	.01 $\pm$.00

Table 3.3: **Projection error (in meters) of different strategies for ballistic annotation.** Ball center in 2D or ball center in 3D are annotated on n initial images and three motion models fitting the initial annotations are evaluated. When fitting 2D positions, the motion models cannot be used to recover the ball 3D positions. In 3D, the more realistic the model is, the better it is for annotation of 3D ballistic trajectories.

travelled between two consecutive frames, which should theoretically be enough to infer ball depth. However, in our experiments, the projection error reaches close to 1 meter and is even larger with a more realistic model that has more degrees of freedom, despite a pixel error below 1 pixel, even for the **Parabola** model. We can conclude that the models overfit the tiny sub-pixel errors made when annotating the ball center, at the cost of deviating from the true ballistic trajectory.

When fitting the motion models from initial annotations provided in 3D, annotation of the two ballistic extremities already provide enough information to fit a model (the **Parabola**). The average error is 5 centimeters in world coordinates and the pixel error is 2.6 pixels in image coordinates. This result is sufficient to get an approximate position of the ball. When the position in the image space is more critical however, a more realistic motion model like the **Ballistic (with lift)** model is required.

Figure 3.6 illustrates different annotation strategies. The **Parabola** model fit from 2D initial annotations has a large projection error despite fitting the ball centers correctly. Fitting the **Parabola** model using two initial 3D annotations (at the trajectory extremities) yields a very small projection error but has a moderate pixel error in the middle of the trajectory. Adding more initial annotations reduce those errors at the cost of increasing errors at the trajectory extremities. Finally, only a realistic **Ballistic (with lift)** motion model fits the data correctly, even with as few as 3 initial annotations, with

3.4. Ballistic trajectories 3D annotation

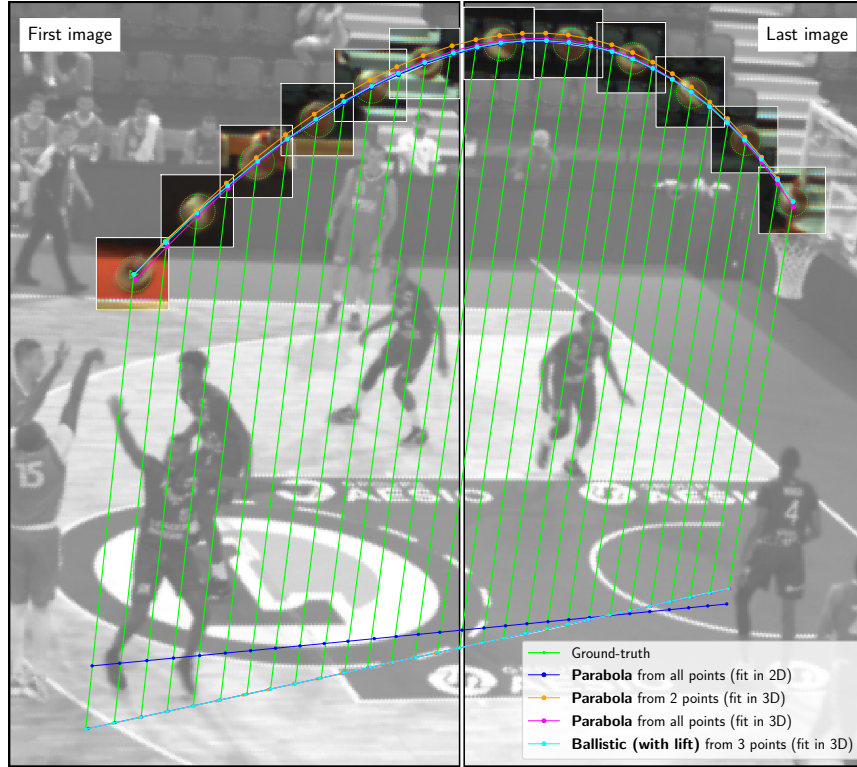


Figure 3.6: **Different strategies to annotate ballistic trajectories.** Ground-truth balls circumference and line to their vertical projection are displayed in **green**. The **Parabola** model that fits 2D ball centers is displayed in **blue** and yields a large projection error. The **Parabola** models fitting 3D annotations yield a small error in the image space, both when using only two images (in **orange**) and using all images (in **magenta**). The best results are obtained using a realistic **Ballistic (with lift)** motion model fit from 3D points, even with only 3 initial annotations as displayed with the model in **cyan**.

a low projection error and a low pixel error.

3.4.4 Discussion

Although only three points are required for the annotation of the whole trajectory, this method requires an advanced annotation interface, beyond what is generally offered in common annotation tools. Indeed, the interface

Chapter 3. Ball 3D annotations from a single calibrated viewpoint

relies on camera calibration to (i) implement the annotation of individual 3D positions at the trajectory extremities with the approach described in the previous section, (ii) display a line on the ground floor, joining the vertical ground projection of the two trajectory extremities, and (iii) leverage this line for the annotation of the intermediate point(s). In addition, we must stress the importance of accurate timestamps at which the physical motion model is evaluated to produce the final annotation, and which were available thanks to the professional-grade cameras used by the Keemotion system.

3.4.5 Conclusion

In this section, we leverage our image-based ball 3D annotation method to annotate ballistic trajectories on multiple consecutive images, with a very limited number of human annotations necessary. We show that 3D annotations are required, which increases the value of our image-based annotation method even further.

We also show that a realistic model that includes drag and lift forces is required to model ball motion from annotated 3D positions. This result contrasts with the simple parabola model used in all related works that estimate ball 3D localization from 2D detections during ballistic trajectories.

3.5 Summary

In this chapter, we propose a method for ball 3D annotation from a single image. In addition, we study different strategies for ballistic trajectories annotation, with the goal of minimizing annotation efforts. These two complementary methods enable annotation of video sequences from a single calibrated viewpoint. Our experimental study show that annotating only 3 images evenly spread along ballistic trajectories is enough to get the full ballistic annotations, but 3D annotation is required, which increases the value of our image-based method even further.

We used our image-based method to annotate two basketballs datasets made publicly available: the *DeepSport-dataset* and the *Ballistic-dataset*. A summary of the different datasets used in this thesis is provided in Table 3.4.

The *DeepSport-dataset* offers high resolution instants captured from fixed camera setups. Beyond providing pairs of consecutive images, it offers a representative sample of professional basketball scenes gathered on multiple different arenas. It features a large variety of game actions and lighting conditions, with cluttered scenes and complex backgrounds. This solves

3.5. Summary

	Arenas	Balls	Sizes [px]	Chapters		
				4	5	6
<i>DeepSport-dataset</i>	15	398	14–37	•	•	•
<i>DeepSport-dataset-extended</i>	34	1470	14–45		•	•
<i>Ballistic-dataset</i>	2	233	19–40		•	
<i>Sequences-dataset</i>	2	1398	10–19		○	•
<i>Sequences-dataset</i>	5	~50k [†]	N/A			•
<i>APIDIS-dataset</i>	1	4019 [‡]	5–27		•	

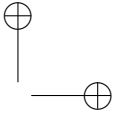
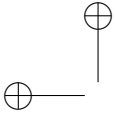
Table 3.4: **Summary of the different datasets used in this thesis.** The two datasets in bold were made publicly available. In chapter 5, the *Sequences-dataset* was used only in the last section that compares thoroughly our two approaches to 3D ball localization.

[†] Our ball detector was used with a very strict detection threshold to generate pseudo ground-truth of ball 2D position.

[‡] As described in section 3.1, ball 3D annotations from the *APIDIS-dataset* are not accurate.

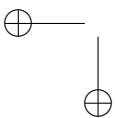
a weakness of several previous works suffering from a limited set of validation data, having very few different games considered (Parisot and De Vleeschouwer, 2019; Reno et al., 2018; Speck et al., 2017; D’Orazio et al., 2009).

The *Ballistic-dataset* features 35 ballistic trajectories from two different arenas and offers high quality ball 3D annotations.

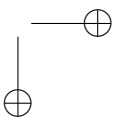


—

—



|



Chapter 4

Ball detection in an image

Locating the ball in team sports is an important complement to player detection (Parisot and De Vleeschouwer, 2017) both to feed sport analytics (Thomas et al., 2017) and to enrich broadcast content (Fernández et al., 2010). In the context of automated production of team sports events (Chen et al., 2011; Chen and De Vleeschouwer, 2010; Chen et al., 2016; Delanay, 2012), knowing the ball position in the image space with *accuracy* and *without delay* is even more critical.

4.1 Related works

In the past ten years, the task of detecting the ball has been largely investigated, leading to industrial products like the automated line calling system in tennis or the goal-line technology in soccer (Collins and Evans, 2008). However, the ball detection problem remains unsolved for cases of important practical interest because of two main issues.

First, the task becomes quite challenging when the ball is partially occluded due to frequent interactions with players, as often encountered in team sports. Earlier works have generally addressed this problem by considering a calibrated multi-view acquisition setup to increase the candidates detection reliability by checking their consistency across the views (Parisot and De Vleeschouwer, 2011; Ren et al., 2008; Intel, 2016; Collins and Evans, 2008; Lampert and Peters, 2012). However, those solutions require the installation of multiple cameras around the field, which is significantly more expensive than single viewpoint acquisition systems (Intel, 2016) and

Chapter 4. Ball detection in an image

appears difficult to deploy in many venues¹.

Second, the weakly contrasted appearance of the ball, as generally encountered in indoor environments, leads to many false detections. This problem is often handled by exploiting a ballistic trajectory prior in order to discriminate between true and false detected candidates (Chen et al., 2009, 2012; Chakraborty and Meher, 2013; Kumar et al., 2011; Zhou et al., 2013; Yan et al., 2006; Parisot and De Vleeschouwer, 2011). However, this prior induces a significant delay, making those solutions inappropriate for real-time applications. Furthermore, they require detecting the ball at high frame rates, inducing hardware and computational constraints and making them unusable for single instants applications.

Overall, for scenarios considering instant images in indoor scenes captured from a single view point, the ball detection problem remains largely unsolved. This is especially true for basketball (involving fast dynamics and many players interactions) where state-of-the-art solutions saturates around 40% detection rate, restricted in the basket area (Parisot and De Vleeschouwer, 2019). Those limitations force current real-time game analysis solutions to rely on a connected ball (Platteau, 2016), which requires to insert a transmitter within the ball ; or use multiple cameras and multiple servers to process the different data streams (Collins and Evans, 2008; Pingali et al., 2000; Intel, 2016).

In this chapter, we propose two learning-based methods reaching close to 70% ball detection rate on unseen venues, with very few false positives. In order to train and validate our methods, we use our *DeepSport-dataset* featuring many interactions between the ball and players and low-contrasted backgrounds.

Similar to most modern image analysis solutions, our method builds on a CNN (Lecun et al., 2015). In earlier works, several CNNs have been designed to address the object detection problem, and among the most accurate, *Mask R-CNN* (He et al., 2017) has been trained to detect a large variety of objects, including balls. The experiments presented in Section 4.4.2 reveal that applying the universal *Mask R-CNN* detector to our problem largely fails. However, fine tuning the pre-trained weights to deal with our dataset significantly improves the detection performance (despite staying $\sim 5\%$ below our method). However, *Mask R-CNN* is too complex to run in real-time on an affordable architecture.

To provide a computationally simple alternative to *Mask R-CNN*, some

¹From our personal experience of 100+ installations in professional basketball arenas.

4.2. Method

works have designed CNN models to specifically detect the ball in a sports context. (Reno et al., 2018) adopts a *classification* strategy by splitting the image in a grid of overlapping patches, each of which is fed to a CNN that assesses whether or not it contains a ball. However, the model is far from real-time², and only two different games are considered to train and validate the method. In the Robocup Soccer context, Speck et al. (2017) formulates the detection problem as a *regression* task that directly predicts the ball coordinates in the image space from the whole image. The performance remains relatively poor despite the reasonable contrast between the white ball and the green field. Furthermore, regression of absolute object coordinates is known to be poorly addressed by CNNs (Liu et al., 2018). Hence, this strategy is expected to poorly generalize to real team sport scenes.

In comparison to those initial attempts to exploit CNNs to detect the ball, the contributions of our work are multiple and multifaceted. Primarily, we propose to formulate the ball detection problem as a *segmentation* problem, for which CNNs are known to be quite effective (Chen et al., 2017). Such formulation is especially relevant in our team sport ball detection context since there is only one object-of-interest that we aim to detect. Hence, the CNN does not need to handle the object instantiation problem. In addition, we propose to use a pair of consecutive images coming from a fixed (or motion compensated) viewpoint to take advantage of the ball dynamics without the delay caused by a temporal regularization, allowing low-latency applications. We show that this approach allows fast and reliable ball detection in weakly contrasted and cluttered scenes. Furthermore, we show that the use of test-time data augmentation³ permits a significant increase in the detection accuracy at small false positive rates. This is in the continuation of recent works showing that test-time data augmentation can be used to estimate the prediction uncertainty of a CNN model (Ayhan and Berens, 2018; Wang et al., 2019).

4.2 Method

Our proposed solution is illustrated in figure 4.1. It formulates the ball detection problem as a segmentation problem where a fully convolutional neural network is trained to output a heatmap Ψ predicting the ball segmen-

²We measured 3.6fps on an Nvidia RTX 2080 Ti with an overlap of 10 pixels between the patches. See their paper for implementation details.

³Generating multiple transformed versions of an input sample, in order to predict an ensemble of outputs for that input sample

Chapter 4. Ball detection in an image

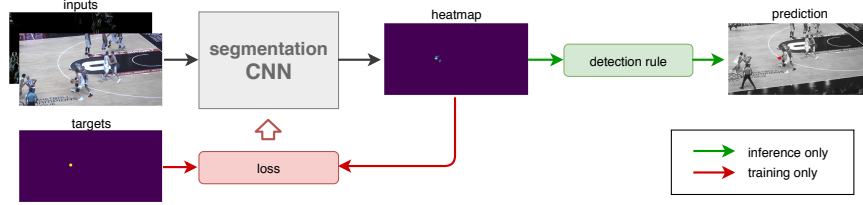


Figure 4.1: **Diagram of our segmentation based detector.** Our detector is based on a segmentation task performed by a fully convolutional network that outputs a heatmap of the ball position. At inference, a detection rule is used to predict the ball location from the heatmap.

tation mask. At inference, a detection rule is used to extract a ball candidate from the heatmap. The fully convolutional workflow enables working with any input sizes.

CNN implementation. Among fully convolutional neural networks, multi scale architectures are known to better trade-off complexity and accuracy on large images by combining wide shallow branches that manipulate fine image details through a reduced number of layers, with deep narrow branches that access smaller images and can thus afford a deeper sequence of layers (Yu et al., 2018; Mazzini, 2019; Poudel et al., 2019; Zhao et al., 2018). The major benefit of this trade-off is a fast processing allowing real-time applications.

In practice, the CNN used to evaluate our method is a custom implementation of ICNet where (i) the 3 input resolutions were changed from $\{(1), (1/2), (1/4)\} \times \text{input size}$ to $\{(1), (1), (1/2)\} \times \text{input size}$ in order to better handle the small size of the ball at the lowest resolutions; and (ii) the input layers were adapted to handle the 6 channels input data (see equation 4.1). The output heatmap Ψ is produced by applying a softmax on the last layer and removing the channel relative to the background.

From a computational point of view, the selected CNN offers satisfying real-time performances (see Section 4.4.2 Table 4.1) but any segmentation network could be used instead; and we could expect even higher detection speed using recent fast segmentation networks (Yu et al., 2018; Mazzini, 2019; Poudel et al., 2019).

Detection rule. We infer ball candidates as points lying above a threshold τ in the predicted heatmap Ψ . Since we know *a priori* there is only one ball

4.2. Method

in the scene, the detection rule is further constrained by limiting the number of ball candidates per scene. This approach, named *top-k*, detects (up to) the k highest spots in Ψ , as long as they are higher than the threshold τ . In practice, to avoid multiple detections for the same heatmap hotspot, highest points in Ψ are selected first, and their surrounding pixels are ignored for subsequent detections, in a greedy way. Note that the computational cost associated with this detection rule is negligible compared to the CNN inference.

Exploiting the ball dynamics. When not held by a player, the ball generally moves rapidly, either due to a pass between players, a shot, or dribbles. To provide the opportunity to exploit motion information, we propose to feed the network with the information carried by two consecutive images, each composed of the 3 conventional channels in the *RGB* space. Two strategies were considered. The first one, consisting in feeding the network with the concatenation of the two images on the channels axis, gave poor results and is not presented in this work. The second strategy consists in concatenating the image of interest together with its difference with the previous image. Hence, for two consecutive images a and b and their corresponding sub-scripted R (red), G (green) and B (blue) color channels, we have:

$$\text{Input} = (R_a, G_a, B_a, |R_a - R_b|, |G_a - G_b|, |B_a - B_b|) \quad (4.1)$$

Training. Because of the custom number of input channels required by our method, no pre-training of the network is done. The training is performed using the Stochastic Gradient Descent optimization algorithm applied on the mean cross-entropy loss, at the pixel level, between Ψ and the binary segmentation mask of the ball. The meta parameters were selected based on grid searches. The learning rate has been set to 0.001 (decay by a factor of 2 every 40 epochs), batch size to 4, and number of epochs to 150 in all our experiments. The weights obtained at the iteration with the smallest error on the validation set were kept for testing. The network is fed with 1024×512 pixels inputs obtained by a data augmentation process including mirroring (around vertical axis), up- and down- scaling (maintaining the ball size between 15 and 45 pixels, which corresponds to the size range of balls observed by the cameras used to acquire the dataset) and cropping (keeping the ball within the crop).

Chapter 4. Ball detection in an image

Terminology. We will use the term **scene** when referring to the original image captured by the camera, and **random-crop** to denote a particular instance of random cropping, scaling, and mirroring parameters. Multiple different random crops can thus be extracted from a single scene.

4.3 Validation Methodology

Dataset. The experiments were conducted on our *DeepSport-dataset*. As described in section 3.2.1, it offers a large variety of game configurations and various lighting conditions. The ball was manually annotated and images in which at least half of the ball wasn't visible were discarded.

K-fold testing. To validate our work, a K -fold training/testing strategy has been adopted in our experiments. It partitions the dataset into $K = 7$ subsets, named folds, and runs K iterations of the training/testing procedure. Each iteration preserves one fold for testing, and shuffle the other folds before splitting the samples in 90% for the training set and 10% for the validation set. To assess the generalization capabilities of our model to unseen games and arenas, the folds were defined such that each fold only contains images from arenas that are not present in the other folds.

ROC curves and metrics. For a given top- k detection rule, the accuracy is assessed based on ROC curves. Each ROC plots the detection rate (TPR) as a function of the false positive rate (FPR), while progressively changing the detection threshold τ . The detection rate measures the fraction of scenes for which the ball has been detected, while the false positive rate measures the mean number of false candidates that are detected per scene. In practice, a detection is considered as being a true (false) ball detection if it lies inside (outside) the surface covered by the ball in the annotated segmentation mask.

Test-time data augmentation. We also conducted experiments where multiple different random crops from the same scene were considered at test-time. When aggregating their output heatmap, we took care of combining random-crops with a high overlap (IoU > 0.9); see figure 4.7 (left).

4.4. Experimental results

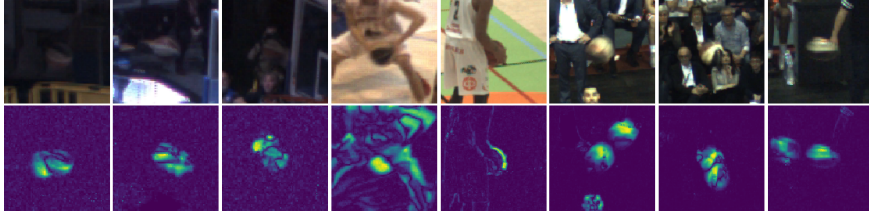


Figure 4.2: **Ball detections for which two consecutive images help.** Image samples (top) and corresponding mean differences to previous image (bottom) for scenes in which the ball is detected when providing the difference to the previous image to the network, but remains undetected by the model that ignores the difference.

4.4 Experimental results

This section evaluate our Ball Segmentation method named *BallSeg*. First, we validate the CNN input and the detection rule. Then, we compare *BallSeg* with *Mask R-CNN* fine tuned with our dataset. Finally, we study how test-time data augmentation impact the detection metrics. In particular, we combine multiple random-crops of the same scene to improve detection accuracy at the cost of increased computational complexity, by merging the associated heatmaps.

4.4.1 Inputs choice and detection rule validation

We compared the performances of *BallSeg* when feeding the network with and without the difference between the two consecutive images. Figure 4.3 (left) shows that presenting the input as described in equation 4.1 significantly improves the detection accuracy. Figure 4.2 reveals that using the difference with the previous image allows detecting the ball in low contrasted scenes.

Different top- k detection rules are compared in figure 4.3 (right). A top-1 detection rule achieves better true/false positive trade-offs at low false positive rates (for $\tau = 0.01$: TPR=0.66 and FPR=0.33). This is far better than all previous methods presented in the related works section, especially given that our dataset includes a large variety of scenes, presenting the ball in all kinds of game situations.

In the following results, unless specified otherwise, results use a top-1 detection rule, and the input described by equation 4.1.

Chapter 4. Ball detection in an image

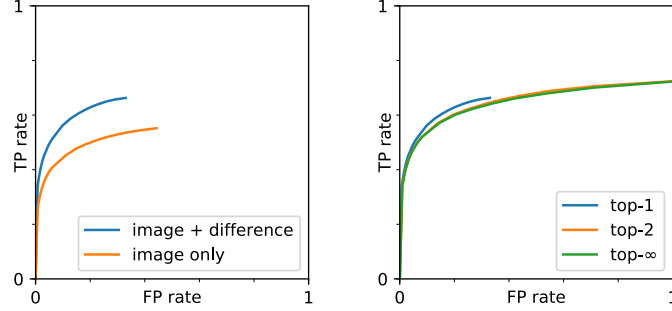


Figure 4.3: **BallSeg ROC curves.** Left: adding the difference with the previous image to the network input improves accuracy (top-1 detection rule). Right: top-1 detection rule achieves better true/false positives trade-offs.

4.4.2 Comparison with SotA object detection method

To better evaluate the value of our *BallSeg* model, we compared it against the universal *Mask R-CNN* model. We used the implementation of Abdulla (2017) which provides weights trained on the MS-COCO dataset (Lin et al., 2014) that already has a class for balls. For a fairest comparison, we also fine-tuned the *Mask R-CNN* on our dataset. Hence, the resulting model, denoted *Ball R-CNN*, was obtained by running a conventional optimization with stochastic gradient descent optimizer with a learning rate lr . First, we updated the front part of the network for n_f epochs, then we updated the whole network for n_w epochs. A grid search has been performed to select the (lr, n_f, n_w) triplet. Best K -fold mean test set accuracy was obtained with $lr = 10^{-3}$, $n_f = 10$ and $n_w = 20$.

$$\begin{aligned} lr &\in \{10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\} \\ n_f &\in \{0, 1, 10\} \\ n_w &\in \{0, 1, 10, 20, 100\} \end{aligned}$$

In figure 4.4, we observe that *BallSeg* outperforms *Ball R-CNN*. Besides, the fact that *Mask R-CNN* largely fails reveals the need to have sport-specific datasets for the complex task of ball detection in a team sport context. Figure 4.5 shows a visual comparison between *BallSeg* and *Ball R-CNN* on different image samples from our dataset.

4.4. Experimental results

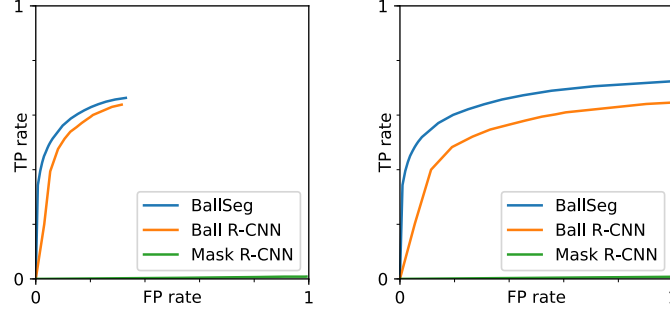


Figure 4.4: **Comparison between *BallSeg* and *Mask R-CNN*.** The two computationally complex *Mask R-CNN* instances are *Mask R-CNN* pre-trained on MS-COCO, and the same network fine tuned on our dataset (*Ball R-CNN*). Left: top-1 detection rule; Right: top-2 detection rule.

In addition to this qualitative analysis, Table 4.1 presents the measured computational complexity of the two models: *BallSeg* using our *ICNet* implementation and *Ball R-CNN* being *Mask R-CNN* from Abdulla (2017). We observe that, besides reaching a better accuracy than *Ball R-CNN*, *BallSeg* is significantly faster. Moreover, *ICNet* could potentially reach even faster inference speed once compressed with filter-pruning, with expected limited effect on its accuracy (Andrienko, 2019; Li et al., 2017).

4.4.3 Test-time data augmentation

This section investigates how different random crops of the same scene impacts the detection. It first observes that the ball segmentation errors rarely affect all random crops. In other words, prediction errors correspond

	single image $1024 \times 512 \times 3$	image + diff $1024 \times 512 \times 6$	image + diff $1280 \times 720 \times 6$
<i>Mask R-CNN</i>	4.33 fps	N/A	N/A
<i>BallSeg</i> (our method)	38.39 fps	24.67 fps	12.08 fps

Table 4.1: **Complexity comparison.** We compared the framerate of the two methods on an Nvidia GTX 1080 Ti, with a batch size of 2, without using filter-pruning optimization. The segmentation approach is significantly faster than *Mask R-CNN*.

Chapter 4. Ball detection in an image



Figure 4.5: **Comparison of our *BallSeg* with *Ball R-CNN*. *BallSeg* gives accurate results in many different configurations, with fewer false positives.**

4.4. Experimental results



Figure 4.6: **Detection rate using test-time data-augmentation.** 1000 random-crops were created for each scene. Ball detection rate as a function of the number of random crops considered for a scene, for a top-1 detection rule. The ball is detected in the scene if it is detected in at least one of the random crops considered.

to high entropy output distributions, which by definition correspond to a high uncertainty. This is in line with the observation made by Ayhan and Berens (2018) and Wang et al. (2019) that large diversity of predictions for transformed input are associated with high uncertainty levels. More interestingly, our experiments also reveal that false positives are not spatially consistent across the ensemble of heatmaps. This is an important novel observation, since it gives the opportunity to significantly increase ($\sim 10\%$) the detection accuracy at small false positive rate by aggregating the ensemble of heatmaps obtained by multiple different random-crops.

Random crops consistency and failure case analysis Figure 4.6 presents, for a top-1 detection rule, the ball detection rate for different number of random crops generated for a scene. We observe that for 93% of the dataset scenes, when 20 different random crops are generated for each scene, the ball is detected as top-1 in at least one of those random crops. This suggests that the detection rate could be improved by combining multiple random crops (see Section 4.4.3).

Accuracy/complexity trade-off This paragraph investigates how the true-false positive trade-off can be improved based on the computation of more than one heatmap per scene. It builds on the fact that the ball has a higher

Chapter 4. Ball detection in an image

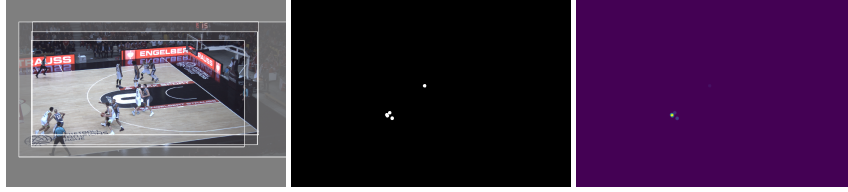


Figure 4.7: **Heatmaps averaging with test-time data-augmentation.** Averaging the heatmaps over random image samples helps to discriminate the ball among candidates. Left: Five random crops having similar IoU in a given scene are used; Middle: Scatter of their heatmaps most salient points; Right: Mean heatmap intensity using the five input heatmaps.

chance to be detected if more random crops are considered (see figure 4.6), and on the observation that the ball generally induces a significant spot in the heatmap of every random crop, even if this spot is not always the highest one.

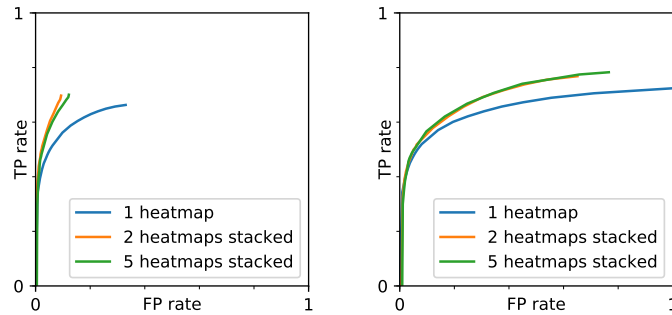


Figure 4.8: **Heatmaps averaging with test-time data-augmentation.** Left: top-1 and Right: top-2 ROC curves obtained when heatmaps coming from 1, 2 or 5 similar random crops of the same scene are averaged.

Figure 4.7 shows an example where the heatmaps predicted for distinct random crops of the same scene are generally more consistent at the actual ball location than for the false candidates. From this observation, we propose to merge the heatmaps of different random image samples by averaging their intensity. Figure 4.8 shows that applying a top-1 detection rule on the averaged heatmap significantly improves the true vs. false positive rate tradeoff ($\sim 10\%$ detection rate increase at small false positive rate), already when only two random image samples are averaged.

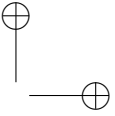
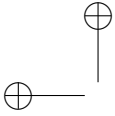
4.5. Conclusion

This gives the opportunity to improve accuracy at the cost of additional complexity, which in turn could support accurate automatic annotation of unlabelled data for more accurate models training, thereby following the data distillation paradigm (Radosavovic et al., 2018).

4.5 Conclusion

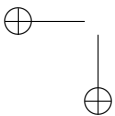
This chapter proposes a new approach for ball detection using a simple camera setup by (i) adopting a CNN-based segmentation paradigm, taking advantage of the ball uniqueness in the scene, and (ii) using two consecutive frames to give cues about the ball motion while keeping a very low latency.

Furthermore, the approach benefits from recent advances made by fast segmentation networks, allowing real-time inference. In particular, the segmentation network used to demonstrate the approach reduces drastically the computational complexity compared to the conventional *Mask R-CNN* detector, while achieving better results: close to 70% detection rate (with a very small false positive rate) on unseen games and arenas, based on an arduous dataset.

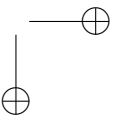


—

—



|



Chapter 5

Ball 3D localization from a single calibrated image

Accurately localizing objects in three dimensions (3D) is crucial for various computer vision applications, such as robotics, autonomous driving, and augmented reality. This task finds another important application in sports analytics with 3D ball localization. In this chapter, we present two methods for localizing basketballs in 3D from a single calibrated image.

5.1 Related works

Ball 3D localization has largely been studied (Kamble et al., 2019), leading to industrial products like the automated line calling system in tennis or the goal line technology in soccer (Bal and Dureja, 2012). Most existing solutions rely on multiple viewpoints (Kumar et al., 2011; Pingali et al., 2000; Ren et al., 2008; Lampert and Peters, 2012; Cheng et al., 2018; Parisot and De Vleeschouwer, 2011, 2019; Maksai et al., 2016) to triangulate the 2D ball positions detected in individual frames, thereby offering high reliability in case of occlusions, which are frequent and severe in some team sports like basketball or American football. Obviously, due to the exploitation of multiple synchronized cameras, those systems are expensive and complex to install, compared to single viewpoint setups. Yet, there are many practical cases where ball 3D localization is valuable even though only a single viewpoint is available. This typically happens when the end-user is interested in the 3D reconstruction and analysis of the play e.g. to collect

Chapter 5. Ball 3D localization from a single calibrated image

statistics related to the 3D movement of the ball and players.

Several works have addressed the question of ball 3D localization using single calibrated viewpoints. They generally rely on 3D ballistic trajectory fitting from the 2D detections (Chen et al., 2012; Shum and Komura, 2004; Chen and De Vleeschouwer, 2009; Kim et al., 1998; Sköld, 2015; Parisot and De Vleeschouwer, 2019; Labayen et al., 2014; Silva et al., 2011). This approach, very well described in (Sköld, 2015), is however limited to detections on video sequences (multiple consecutive images) and when ball is in free fall (ballistic trajectories). Nonetheless, in many game situations, the ball is only partially visible or not in free fall. Therefore, in this chapter, we drop assumptions of ballistic trajectory, time consistency, and clean visibility, and we aim at localizing the ball in 3D from a *single* calibrated image. In particular, we validate our method on basketball games where ballistic trajectories (typically long passes and shots) only represent less than 10% of game time¹.

5.2 Diameter approach

The first method relies on the estimation of the ball diameter in pixels and uses knowledge of the real ball size and camera calibration to locate the ball in the 3D space. We use a two-stage approach where a small convolutional neural network (CNN) is trained to predict ball diameter from the candidates generated by a detector. In addition, the CNN is also trained to confirm/refute the presence of a ball in the image patch, thereby improving the performance of the global image-wise ball detector. To the best of our knowledge, our approach is the first one to make ball 3D localization possible from a single viewpoint without resorting to temporal analysis with ballistic trajectory assumption.

5.2.1 Method

We propose to formulate the ball 3D localization task in a single calibrated image as the combination of two image analysis problems, namely the ball detection and the ball diameter regression problems. Calibration information and knowledge of the actual ball diameter, in meters, are then used to compute the ball localization in the 3D space from its diameter and position in the image pixels space.

¹Evaluated on our dataset of sequences, see section 3.2.2.

5.2. Diameter approach

Ball detection in image space

To detect balls in the image space, we adopt the State-of-the-Art solution *BallSeg* (Van Zandycke and De Vleeschouwer, 2019) presented in section 4.2 that uses a segmentation approach. Specifically, the model, based on an ICNet architecture (Zhao et al., 2018), is trained to predict a mask of the ball. At test-time, the top- k highest peaks are identified on the output heatmap Ψ to produce k ball candidates.

Ball diameter estimation in pixels

Baseline: A natural and well-known approach to estimate the size of circular and highly contrasted objects in an image consists in applying the Hough-circle transformation (HCT) (Duda and Hart, 1972) to the image gradient magnitude. As a baseline, to estimate the ball diameter in the image space, we propose to use the Hough transformation on Ψ , over a spacial neighborhood of the top ($k=1$) ball candidate. In practice, several pre-processing steps are applied prior to the Hough transformation. A morphological opening operation with a circular filter of diameter ρ is first applied to Ψ . Edges are then detected using adaptive thresholding with hysteresis (Canny, 1986), with high and low thresholds τ_h and τ_l . Finally, the Hough-circle transformation is computed from candidate diameters bounded in a range defined by the physical limits of the scenes (see section 5.2.4).

Our proposed CNN-based method: In this section, we propose to leverage a CNN to regress the ball diameter, in pixels, from an image patch I centered on the candidates proposed by any conventional ball detector. For a detector that produces k bounding boxes on the initial input image, when $k > 1$, it is likely that most candidates are false positives. Hence, in addition to regressing the ball diameter, our CNN is trained to predict whether the input patch contains a ball or not. As demonstrated in Section 5.2.4, this offers the possibility to increase the ball detection rate at constant false positive rate, compared to a detector configured to output the most confident detection.

The CNN model $\mathcal{M}_w$ adopted in this work follows a VGG (Simonyan and Zisserman, 2015) architecture that takes image patches I as input, and extracts a feature vector which is connected to a three fully connected layers head $\mathcal{H}_w$ inspired by the work from LeCun et al. (1998). The last layer has

Chapter 5. Ball 3D localization from a single calibrated image

2 neurons that produce the regression $\hat{d} := \mathcal{H}_w^d(\mathcal{M}_w(I))$ and classification $\hat{c} := \mathcal{H}_w^c(\mathcal{M}_w(I))$ outputs.

We trained $\mathcal{M}_w$ and $\mathcal{H}_w$ using batch stochastic gradient descent over w . The classification output $\hat{c}$, is supervised with binary cross entropy loss

$$\mathcal{L}^c(c, \hat{c}) = -c \cdot \log(\hat{c}) + (1 - c) \cdot \log(1 - \hat{c}) \quad , \quad (5.1)$$

with $c \in \{0, 1\}$ being the target class where 1 means I is centered on a ball. The regression output is supervised with Huber Loss (Huber, 1964) defined by

$$\mathcal{L}^d(d, \hat{d}) = \begin{cases} \frac{1}{2}(d - \hat{d})^2 & \text{for } |d - \hat{d}| \leq \delta \\ \delta(|d - \hat{d}| - \frac{1}{2}\delta) & \text{otherwise} \end{cases} \quad , \quad (5.2)$$

where $d \in \mathbb{R}_0^+$ is the target diameter in pixels, and the meta-parameter δ represents the threshold at which the loss turns from linear to quadratic. The two losses were weighted with a parameter α

$$\mathcal{L} = \alpha \cdot \mathcal{L}^d + (1 - \alpha) \cdot \mathcal{L}^c \quad , \quad (5.3)$$

but $\mathcal{L}^d$ was ignored for non ball inputs.

At test time, given k candidates predicted by our detector, we extract k crops I_i , with $i \in [1, k]$, from the initial RGB input image. The final ball detection candidate is the one that has maximum c_i .

Turning ball diameter into 3D localization

For a camera defined by a focal length f , pixel density μ_x and μ_y in the x and y direction respectively, a skew coefficient γ , and a principal vector (u_x, u_y) , we have the camera matrix

$$K := \begin{bmatrix} f \cdot \mu_x & \gamma & u_x \\ 0 & f \cdot \mu_y & u_y \\ 0 & 0 & 1 \end{bmatrix} \quad . \quad (5.4)$$

Given the diameter d and ball position (b_x, b_y) in pixels in the image space, the 3D rays of the ball center $\mathbf{b}^c$ and two diametrically opposed ball edges

5.2. Diameter approach

$\mathbf{e}_-^c$ and $\mathbf{e}_+^c$, expressed in the camera coordinate system, are

$$\mathbf{b}^c = K^{-1} \cdot \mathcal{R} \left(\begin{bmatrix} b_x \\ b_y \\ 1 \end{bmatrix} \right) \quad \text{and} \quad \mathbf{e}_\pm^c = K^{-1} \cdot \mathcal{R} \left(\begin{bmatrix} b_x \\ b_y \pm \frac{d}{2} \\ 1 \end{bmatrix} \right), \quad (5.5)$$

where $\mathcal{R} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ is the function that rectifies image coordinates to handle lens distortions.

For a camera placed at $\mathbf{c}^w \in \mathbb{R}^3$ in the world coordinates system, with an orientation defined by the rotation matrix R , and the true ball diameter ϕ , the 3D localization in the ball is then given by $\mathbf{b}^w$:

$$\mathbf{b}^w = R^T \cdot \frac{\phi \cdot \mathbf{b}^c}{\mathbf{e}_{+y}^c - \mathbf{e}_{-y}^c} + \mathbf{c}^w \quad (5.6)$$

5.2.2 Datasets

Our deep learning method requires a dataset to train and evaluate on. However, as described in chapter 3, datasets for 3D ball localization are very limited, both in term of quality and diversity. For this reason, we used our *DeepSport-dataset* for training and adopted the split defined in section 3.2.1.

For evaluation, in addition to the *DeepSport-testset* where many balls are partly occluded in cluttered scenes as often encountered in basketball games, we also used our *Ballistic-dataset* which features samples where the ball moves freely and does not include occluded balls or balls held in player hands. The *APIDIS-dataset* has also been used for evaluation to illustrate the generalization capabilities of our model and assess the impact of annotation errors. Finally, we also made an experiment using the *DeepSport-dataset-extended* to evaluate the benefit of training on a larger dataset. A description of the different datasets used in this chapter is provided in table 5.1, and samples of each dataset are given in figure 5.6. Images in which the ball was not at least half visible were discarded.

To reduce the sensitivity to hard and easy samples with the small *DeepSport* testset, we repeated its evaluation 5 times using test-time data-augmentation. The data-augmentation uses random scaling such that the balls diameter are uniformly distributed between 14 and 37 pixels, corresponding to the range of ball sizes in the dataset.

Chapter 5. Ball 3D localization from a single calibrated image

	Resolution	ball size range [px]	Arenas	Images
<i>DeepSport</i> training set	2 – 5 Mpx	14 – 37	12	297
<i>DeepSport</i> validation set	2 – 5 Mpx	14 – 37	12	55
<i>DeepSport</i> -testset	2 – 5 Mpx	14 – 37	3	46
<i>Ballistic-dataset</i>	2336×1752	19 – 40	2	233
<i>APIDIS-dataset</i>	800×600	5 – 27	1	4019
<i>DeepSport-dataset-ext.</i>	2 – 5 Mpx	14 – 45	34	1470

Table 5.1: **Summary of the different datasets used in this chapter.** The *DeepSport-dataset* was used both for training and for evaluation, as it offers the largest environment diversity with high quality annotations. The other datasets were used for evaluation only. The numbers reported correspond to the images in which the ball could be projected. The extended version of *DeepSport* was used to assess the impact of the training set size on the performances.

5.2.3 Evaluation procedure

Implementation details and metrics used to conduct this research are presented hereafter.

Ball detector *BallSeg* was trained using the procedure described in chapter 4, with a random scaling strategy where ball size was kept between 14 and 37 pixels, corresponding to the range of ball sizes in the *DeepSport-dataset* (see table 5.1). To analyze the impact of working on smaller balls, we also considered halving the size range in specific trainings (see Section 5.2.4). In our context, we are only interested in the ball being played. Therefore, only the top ($k = 1$) candidate was kept, unless stated otherwise.

Baseline method A 64×64 pixels neighbourhood of Ψ around the detection candidate was used to apply the different image processing steps; with $\rho = 37$, $\tau_l = 10$ and $\tau_h = 20$, selected by maximizing the performances on a validation set. The range of candidate diameters for the Hough transformation was constrained, with a 2 meters margin, by the physical boundary of the basketball court.

Proposed method In practice, although our CNN method can be combined with any ball detector we used the detections produced by *BallSeg* as input to our model, allowing a fair comparison with the baseline. As

5.2. Diameter approach

described in section 5.2.1, $\mathcal{M}_w$ is a VGG feature extractor and $\mathcal{H}_w$ is a 3 fully connected layers head. We took VGG16 pre-trained on ImageNet (Deng et al., 2009) from TensorFlow library (Developers, 2022), and used Glorot initializer (Glorot and Bengio, 2010) to initialize the last layers. Our CNN was trained using stochastic gradient descent over w , on batches of 4 different images, with $k = 4$ candidates output by *BallSeg*, making an effective batch size of 16 image patches I . Training took 100 epochs using Adam optimizer (Kingma and Ba, 2015), with a learning rate schedule starting at 10^{-4} and reducing by a factor 2 for 2 epochs every 10 epochs starting at epoch 50. The loss parameters $\delta = 1.0$ and $\alpha = 0.5$ were selected empirically using the validation set. The best performances were achieved with an image patch size of 64×64 pixels.

Metrics To assess the diameter estimation accuracy, we used the mean absolute error ($\text{MAE}_{[\text{px}]}$) in pixels between the labels d and the predictions $\hat{d}$. The corresponding projection error ($\text{MAPE}_{[\text{m}]}$) is the distance in meters between the real position and the estimated position, computed with equation 5.6, projected on an horizontal plane. It depends on the camera pixel density, focal length and distance to the scene. We also computed the relative distance error ($\text{MARE}_{[\%]}$) that represent the ratio between the projection error and the distance to the cameras. To assess the ball detection performance, we used the Area under the Curve (AuC) associated to the Receiver Operating Characteristic (ROC) curve, plotting the True Positive (TP) rate against the False Positive (FP) rate for a varying detection threshold, as defined in the previous chapter. Here, the TP rate measures the fraction of images in which the ball is correctly detected while the FP rate measures the number of false candidates that are detected.

Experiments were repeated with 8 different initializations of the networks (for both *BallSeg* retrained each time, and the randomly initialized layers of our CNN). Hence, unless stated otherwise, we report the mean metric over the 8 repetitions, along with the associated standard deviation.

5.2.4 Experiments

Ball diameter estimation

The comparison between the baseline using a Hough-circle transformation and our CNN approach is presented in figure 5.1. In this setup, we evaluated on the true positive candidates produced by the detector *BallSeg*. We

Chapter 5. Ball 3D localization from a single calibrated image

observe that our CNN-based approach reduces the mean absolute diameter estimation error $\text{MADE}_{[\text{px}]}$ by 3 pixels on each dataset, reaching less than 2 pixels on the DeepSport testset and the *Ballistic-dataset*.

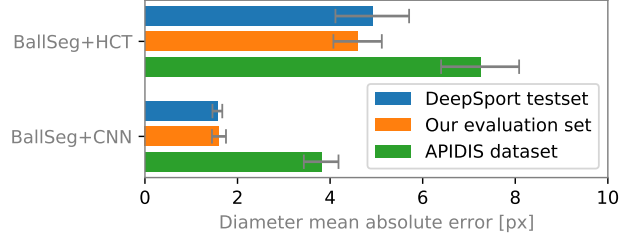


Figure 5.1: **Comparison against a Hough-Circle-Transform approach.** Our CNN method is compared against the Hough-circle transformation baseline (HCT) for diameter estimation on different Basketball evaluation sets. Our method reduces the mean absolute error by a factor two or more.

The fact that the prediction accuracy does not improve between the DeepSport testset and the *Ballistic-dataset*, whose annotation were confirmed by a ballistic motion model, suggests that errors are not caused by annotation errors. On the other hand, the large error observed with the *APIDIS-dataset* corresponds to the large differences between those dataset (see table 5.1). For this reason, in the following experiments, we mainly focused our evaluation on the DeepSport testset and the *Ballistic-dataset*.

Ball 3D localization error

The estimated diameter in pixels combined with camera calibration and knowledge of the ball real diameter enables computing the ball 3D position using equation 5.6. The error analysis given in table 5.2 shows that a diameter error ($\text{MADE}_{[\text{px}]}$) of 1.6 pixels translates to a projection error ($\text{MAPE}_{[\text{m}]}$) of less than two meters on the *Ballistic-dataset* with very accurate annotations. However this relationship highly depends on the camera pixel density, focal length and distance to the scene. Relative to the distance to the camera ($\text{MARE}_{[\%]}$), it corresponds to a 10% error. Recall that our image-based approach provides raw estimations, before any possible time filtering.

5.2. Diameter approach

	TP[%]	MADE _[px]	MAPE _[m]	MARE[%]
<i>BallSeg</i> + HCT (baseline)	47 ± 7 83 ± 2	4.9 ± .8 4.6 ± .5	6.3 ± 1.0 5.1 ± .5	28 ± 5 24 ± 4
<i>BallSeg</i> + CNN (proposed method)	47 ± 7 83 ± 2	1.6 ± .1 1.6 ± .2	2.3 ± .2 1.8 ± .2	10 ± .9 10 ± .7
Oracle + CNN	100 ± 0 100 ± 0	1.9 ± .1 1.5 ± .1	2.8 ± .2 1.7 ± .1	12 ± .6 10 ± .5

Table 5.2: **Error analysis for ball 3D localization.** Error analysis for three different approaches on two different evaluations sets: *DeepSport testset* first, and *Ballistic-dataset* second. *BallSeg* + HTC is the baseline that consists in applying Hough-circle transformation on the heatmap Ψ output by *BallSeg*. *BallSeg* + CNN is our proposed method that estimate diameter from image patches I around detection candidates generated by *BallSeg*. Oracle + CNN is our proposed method applied on oracle detections instead of using a detector. The proportion of TP from the detector (TP[%]) is given for each dataset. The MADE_[px] on oracle detections reaches 1.5 pixels on the *Ballistic-dataset*.

Impact study of the detector

As described in Section 5.2.1, we adopt a two-stage approach where the balls diameter estimated at the second stage depends on the detections proposed by the first stage. To assess the performance of our CNN independently of the detection stage, the quality of diameter estimation was also evaluated on oracle detections by using input image patches I around the true ball positions. The error analysis given in table 5.2 shows that the performances using *BallSeg* detections (1.6 pixels on both datasets) are close to the ones obtained using oracle detections (1.9 and 1.5 pixels).

This study reveals that our CNN gets the ability to correctly estimate the ball size on images where the ball is challenging to detect. Moreover, the similar results between the *DeepSport testset* (featuring many occlusions) and the *Ballistic-dataset*, indirectly reveals that our method is robust to partial occlusions.

Ball detection quality

With its classification output, our CNN model can be used to revise the detection confidence produced by the detector it is combined with. Further-

Chapter 5. Ball 3D localization from a single calibrated image

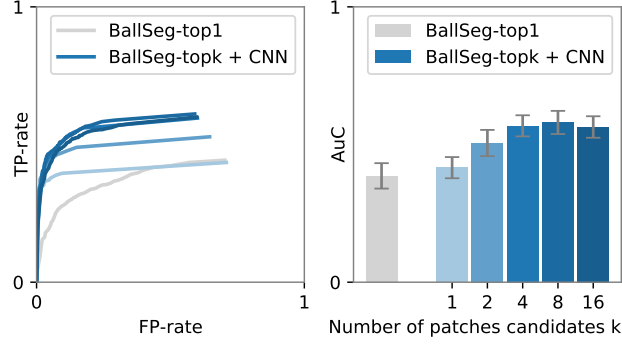


Figure 5.2: **Impact of k on the detection quality of our CNN approach.** (Left) ROC curves obtained by varying the detection threshold, on the accumulated detection candidates over the 8 repetitions (see Section 5.2.3). (Right) Area under the ROC curves for the different configurations. Our CNN is able to recover balls missed by *BallSeg*-top1 but who ranked in the top k with $k \geq 2$.

more, by considering k ball candidates from the detector and keeping the candidate from image patch I_j such that

$$j = \arg \max_{i \in [1, k]} \mathcal{H}_w^c(\mathcal{M}_w(I_i)) \quad , \quad (5.7)$$

our CNN becomes able to recover some of the detections missed when the detector adopts a top-1 detection strategy.

We conducted experiments with *BallSeg* for different values of k and, as revealed by figure 5.2, our model improves the detection quality by a large margin compared to *BallSeg*-top1 alone. More specifically, the ROC curves in figure 5.2 reveal two positive effects. First, our CNN is more effective in discriminating between true positives and false positives, as attested by the larger TP-rate at small FP-rate compared to *BallSeg* for $k = 1$. Second, the number of true positives is largely increased with $k > 1$.

Impact of ball resolution.

In the DeepSport training dataset, ball sizes range from 14 to 37 pixels. However, they are many other applications for 3D ball localization where ball appears smaller, due to lower resolution images like in the *APIDIS-dataset*, or simply because the physical size of the ball is smaller, as encountered

5.2. Diameter approach

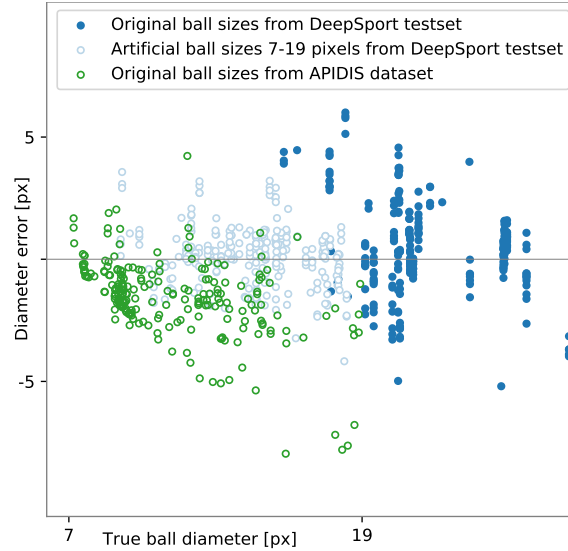


Figure 5.3: **Diameter error distribution with respect to ball true diameter.** Only 4 out of 8 repetitions (see Section 5.2.3) are displayed to declutter the figure. Our CNN was trained on half resolution images (empty circles) and evaluated both on the DeepSport testset where images were scaled accordingly, and the *APIDIS-dataset* where balls larger than 19 pixels or smaller than 7 pixels were ignored. This experiments shows that our CNN approach performs very well at small diameters, validating the approach for many more applications. The distribution shift noticeable on the *APIDIS-dataset* may be caused by the limited quality of that dataset.

for other sports like volleyball or tennis. To evaluate the error distribution one could expect from smaller balls, we trained our CNN on half resolution images and evaluated it on sub-sampled images from the DeepSport testset where images were randomly scaled such that balls appear between 7 and 19 pixels.

Results presented in figure 5.3 reveal that the error distribution is centered around 0 and shows a standard deviation similar compared to the model working at normal resolution. On the *APIDIS-dataset* where balls are smaller (see table 5.1), it appears that the model underestimates the ball diameter.

Chapter 5. Ball 3D localization from a single calibrated image

Impact of image patch size

Multiple patch sizes were considered. As the figure 5.4 reveals, 64×64 image patches resulted in the most accurate estimation of the ball diameter. This study was conducted on oracle detections from the *Ballistic-dataset*, clean from any occlusions.

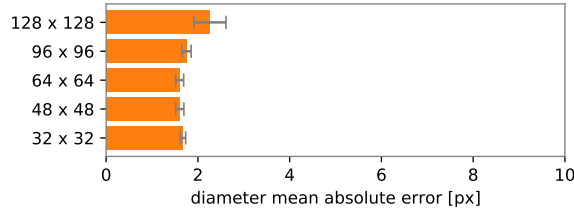


Figure 5.4: **Diameter estimation evaluated on different input image patch I sizes.** This study was conducted on oracle detections from the *Ballistic-dataset*, clean from any occlusions. 64×64 yields the best results.

Impact of training set size

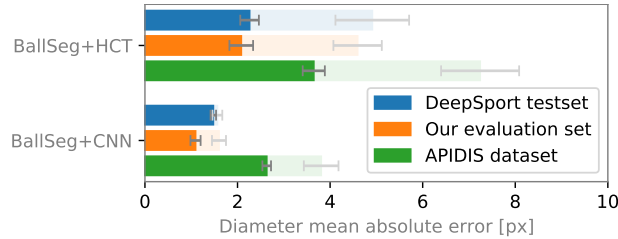


Figure 5.5: **Evaluation with a larger dataset.** Evaluation of diameter estimation when training the models (both *BallSeg* and our CNN) on the *DeepSport-dataset-extended* (see table 5.1). Results from figure 5.1 are shown dimmed for an easier comparison. *BallSeg*+HCT sees a huge performance gain from being trained on a larger dataset, but our CNN model performs better with a pixel error down to one pixel on the *Ballistic-dataset*.

We also evaluated our method by training both the detector *BallSeg* and our CNN on the *DeepSport-dataset-extended* (see table 5.1). Figure 5.5 reveals that our CNN benefits from being trained on a larger dataset, but also that the baseline experiences a significant gain with a drop of more than 50% of the mean absolute diameter error on all datasets. Visual inspection of the

5.2. Diameter approach

output heatmap Ψ shows that ball is much better segmented, resulting in a better diameter estimation using the Hough circle transform approach. Evaluation on the *APIDIS-dataset* consistently yields weaker performances, corroborating the observation that this dataset suffers from low contrasted images and poor annotations.

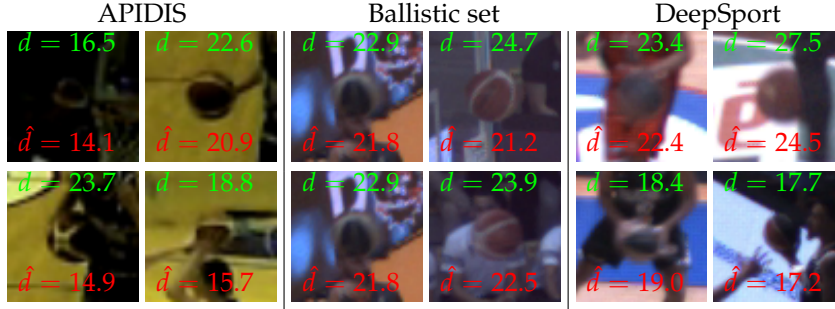


Figure 5.6: **Image samples from the three datasets.** d is the true ball diameter in pixels and $\hat{d}$ is the prediction from one of our models. Samples from the *DeepSport-dataset* were extracted from the testset. Samples from *APIDIS-dataset* were scaled to match ball size range set during training.

5.2.5 Discussion

To our knowledge, the method presented in this section is the first to address ball 3D localization from a single viewpoint. It achieves a projection error of 1.6 meters on our diverse *DeepSport-testset*, and down to 1.2 meters on our *Ballistic-dataset* with balls free from occlusion.

Despite the diversity of works mentioned in the related works section, our results are difficult to put into perspective and compared against multiple viewpoints solutions. Indeed, when multiple viewpoints are available, most related works use the ball 3D localization as a proxy to solve other tasks like ball detection and tracking (Kumar et al., 2011; Pingali et al., 2000; Lampert and Peters, 2012; Ren et al., 2008; Parisot and De Vleeschouwer, 2011, 2019; Maksai et al., 2016) or ball state classification (Cheng et al., 2018) and only report metrics specific to the downstream tasks they address. The only work that report 3D position metrics is the one from Chen et al. (2012) which achieve an average error of 0.12 and 0.18 meters on two synthetic volleyball sequences. This indicate a factor 10 gap between the single view setup and the multi view setup configurations. Nonetheless, our ex-

Chapter 5. Ball 3D localization from a single calibrated image

periments suggest that there is margin for improvements just by training on a larger dataset, which is usually required anyway for any industrial applications.

Our experiments also reveal that our model handles smaller resolution balls from the *DeepSport-dataset* very well, validating the approach for other sports as well. We conjecture that the distribution shift observed on the *APIDIS-dataset* is caused by the poorly contrasted images of that dataset (see figure 5.6).

Finally, our method, which serves as a baseline for the task, relies on a basic VGG backbone whose complexity has not been evaluated. For industrial applications, the use of a more efficient backbone should probably be considered.

5.3 Height approach

The second method we propose relies on the estimation of the ball height in pixels in the image space and uses known camera calibration to convert it into 3D. The observation that motivates this approach is that in basketball, much like in other sports, human observers rely on contextual cues such as the position, occlusion, and size of nearby players, crowd, and court objects like basketball boards to accurately locate the ball during a game. If tasked with estimating the 3D position of the ball in a paused video, a person would use the position of the players' feet on the court as a reference, while making an informed guess about the height at which the ball is held. During passes or shots, the task becomes even more challenging and cues like players gaze orientation are required to improve the height perception.

Building upon previous work (Van Zandycke and De Vleeschouwer, 2019; Ghasemzadeh et al., 2021), we assume that the ball has already been detected in the image, allowing us to crop the image region around the ball detection. By providing sufficient context within the crop, our model predicts the height of the ball in image space, measured in pixels. By placing the ball at the center of the crop, we implicitly convey its location to the model.

This approach is versatile and easily transferable to other sports such as soccer, volleyball, handball, and potentially extensible to other domains beyond sports.

5.3. Height approach

5.3.1 Method

In this section, we propose to estimate the 3D location of balls by training a neural network model to predict the distance in pixels between its center and its ground projection in image space. Retrieving the 3D ball coordinates is then easily achieved by leveraging the projection matrix from the camera calibration data.

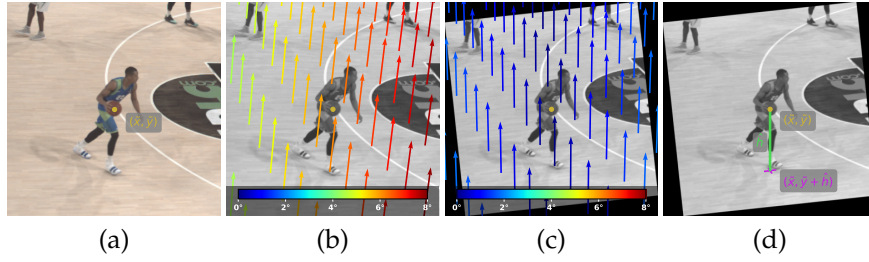


Figure 5.7: **Method for 3D localization from height estimation.** The predicted height and the given 2D position are converted to 3D coordinates using the camera calibration data. (a) Original image with ball position $(\tilde{x}, \tilde{y})$. (b), (c) Colored vectors perpendicular to the court floor show how verticals are oriented relative to image y^I -axis, with a scale given in degrees. (c) Image is rectified aligning the world z^W -axis with the image y^I -axis. (d) Predicted height $\hat{h}$ is used to find the ball vertical projection on the ground floor $(\tilde{x}, \tilde{y} + \hat{h})$ which can be projected to $(b_x^W, b_y^W, 0)$ using the calibration data. The coordinate b_z^W is finally obtained by re-projecting $(\tilde{x}, \tilde{y})$ to the planes $x^W = b_x^W$ or $y^W = b_y^W$.

Ball height estimation in pixels Our model has a Convolutional Neural Network (CNN) backbone to extract image features, which are subsequently 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, ensuring sufficient contextual information is provided.

Pixels height to real-world coordinates Let us define a 3D coordinates system (x^W, y^W, z^W) with the horizontal ground plane defined by $z^W = 0$, for the real world, and a 2D coordinate system (x^I, y^I) , for image space. Details of the camera calibration allow us to project 3D points from the real world to the image space, and to convert 2D points in image space into light

Chapter 5. Ball 3D localization from a single calibrated image

rays in 3D. To transform the estimated height of the ball in image space into its corresponding 3D coordinates in the real world, we apply the following procedure illustrated in figure 5.7.

First, the input image is undistorted, ensuring that straight lines in the real world correspond to straight lines in image space. Then, the image is rotated, aligning the z -axis in the real world with the vertical direction y^I in the image space. After rotation, the predicted height of the ball in pixels $\hat{h}$ effectively represents the vertical displacement between the ball's position in the image $(\tilde{x}, \tilde{y})$, located at the center of the crop, and its projection onto the ground in image space $(\tilde{x}, \tilde{y} + \hat{h})$. As the camera is calibrated, we can project $(\tilde{x}, \tilde{y} + \hat{h})$ into 3D space on the $z^W = 0$ ground plane, determining the position of the ball's projection on the ground $(b_x^W, b_y^W, 0)$. Finally, to obtain the 3D world coordinates of the ball, we perform a re-projection process. We trace the path of the light ray passing through the ball in the image space, determining the two 3D points at which it intersects the $y^W = b_y^W$ and $x^W = b_x^W$ vertical planes. By taking the average of these two points, we retrieve the 3D ball coordinates.

5.3.2 Dataset

To evaluate the effectiveness of our proposed method for ball 3D localization in basketball we use the high-resolution images from our *DeepSport-dataset* with the split presented in section 3.2.1. Unlike the previous section where the *DeepSport* testset was repeated with test-time data-augmentation using a uniform distribution of ball diameters, we used the raw testset without any modification unless stated otherwise.

Beside evaluation of our model on the *DeepSport* testset, in section 5.3.4, we benchmark it also on our *Ballistic-dataset* which features highly reliable 3D ball annotations from ballistic trajectories.

5.3.3 Evaluation procedure

Implementation details and metrics used to conduct this research are presented hereafter.

Baseline As a baseline for our approach, we use the model presented in the previous section whereby the ball 3D position is obtained by estimating its diameter in the image space and combining knowledge of real ball size with calibration data.

5.3. Height approach

Architecture We study the impact of the feature extractor by experimenting with several popular architectures, including VGG (Simonyan and Zisserman, 2015), ResNet (He et al., 2016), MobileNet (Howard et al., 2017) and EfficientNet (Tan and Le, 2019); all pre-trained on ImageNet (Deng et al., 2009) from the TensorFlow library (Developers, 2022). The regression head, identical to the one used in the previous section, contains 3 layers and outputs a scalar supervised to predict the ball height in pixels.

Context size We thoroughly investigate the impact of the square crop size on our model's performance. It is evident that a small crop around the basketball would lack sufficient context for the model to accurately estimate its projection onto the ground. By enlarging the crop size and incorporating elements from the surrounding scene, we anticipate a significant enhancement in model performance. However, it is important to note that as the crop size increases, the improvements in performance may eventually plateau, leading to diminishing returns.

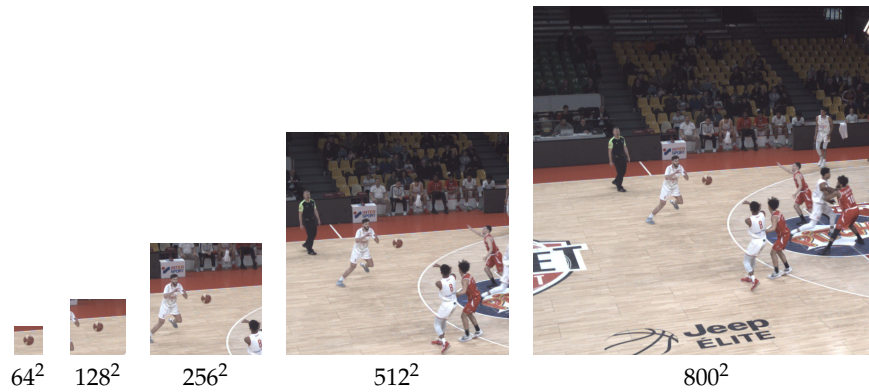


Figure 5.8: **Various context sizes for a given image in the dataset.** Larger context sizes offer more contextual information to the model.

Image scale We investigate the influence of the input image scale (i.e. the number of pixels used to represent a given context or scene) on our model's performance, as it directly affects the representation of the basketball and all relevant objects and people within the scene. Moreover, the combined effect of image scale and context size also influences the inference speed of the model, a critical consideration for real-time analysis in production

Chapter 5. Ball 3D localization from a single calibrated image

systems. Figure 5.9 represent various scale for a given context size. Lower scale images result in lower resolution images.

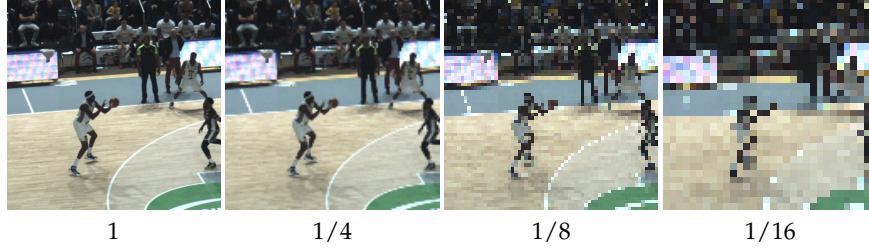


Figure 5.9: **Various image scales for a given image in the dataset.** A lower scale image of 1/8 has its resolution reduced by a factor 8 in each dimension.

Training procedure The backbone is pre-trained on ImageNet (Deng et al., 2009) from the TensorFlow library (Developers, 2022), and we initialize the regression head with the Glorot initializer (Glorot and Bengio, 2010). The neural network is then trained on image patches centered around the oracle ball positions for 100 epochs using Adam optimizer (Kingma and Ba, 2015) with a learning rate of 10^{-4} . The final height prediction is supervised with a Huber loss (Huber, 1964) with $\delta = 1.0$. Each experiment is repeated with 8 different initialization of the regression head and every reported metric is composed of mean and standard deviation computed over the 8 repetitions.

Metrics We evaluate the performance of our 3D ball localization method using the $\text{MAPE}_{[m]}$ metrics presented in the previous section. In addition, we define the Mean Absolute Height Error ($\text{MAHE}_{[px]}$) which measures the average error in pixels for the predicted height of the ball in image space.

The metrics are computed on oracle ball positions in image space. This reveals the true errors from our proposed approach, de-correlated from any error that a detector upstream could introduce.

5.3.4 Experiments

Evaluation against the baseline

As a first experiment, we validate our proposed method for 3D ball localization against the model which relied on ball diameter estimation presented

5.3. Height approach

in the previous section. For the height model, we select a reasonable set of hyperparameters: the crop size is fixed to 512 pixels (about 1/3rd of the average dataset image height) and the backbone is VGG16. These choice of hyperparameters will be taken as default in the following sections. To ensure a fair comparison, the diameter approach was evaluated on oracle detections as well, on the same *DeepSport* testset.

	MAHE _[px]	MAPE _[m]
Height approach	34 ± 3	1.25 ± 0.11
Diameter approach	N/A	2.88 ± 0.22

Table 5.3: **Comparison of performance metrics between the height approach and diameter approach.** On the DeepSport-testset, in which ball elevation is mostly under 3 meters (see figure 5.10), the height approach performs better by a large margin.

As shown in table 5.3, the height approach offers a lower projection error on the DeepSport-testset compared to the diameter approach, which validates the effectiveness of this approach for accurate 3D ball localization.

Impact of backbone

We considered several popular backbone architectures including VGG, ResNet, MobileNet and EfficientNet. Our findings show that they all have similar performance when trained on the *DeepSport-dataset*. We anticipate that when exposed to a larger dataset, larger models would perform better, at the cost of longer inference time.

Impact of context size

Here, we investigate the influence of context size on the predictive power of our model for 3D ball localization (see figure 5.8). Smaller crop sizes provide faster inference and require fewer resources, but may lack sufficient context for accurate height estimation. Conversely, larger crop sizes offer more contextual information, potentially improving model performance. Our goal is to identify the ideal crop size for a production environment where both performance and speed are important. We explore a range of crop sizes: 64, 96, 128, 256, 320, 480, 512, 640, and 800 pixels. For each crop size, we report in table 5.4 the resulting MAPE_[m].

Chapter 5. Ball 3D localization from a single calibrated image

Crop Size	MAPE _[m]
64 × 64	1.71 ± 0.17
96 × 96	1.52 ± 0.13
128 × 128	1.46 ± 0.06
256 × 256	1.04 ± 0.11
320 × 320	1.17 ± 0.09
480 × 480	1.18 ± 0.07
512 × 512	1.25 ± 0.11
640 × 640	1.16 ± 0.07
800 × 800	1.08 ± 0.06

Table 5.4: **Impact of context size on MAPE_[m]**. The crop size is varied whilst keeping the image scale fixed.

As expected, a small context size such as 64×64 pixels around the detection yields a higher MAPE_[m]. As the context size increases, MAPE_[m] decreases, indicating improved performance. However, MAPE_[m] already plateaus beyond a context size of 256 pixels. We thus identify the context size of 256 pixels as the optimal choice, striking a balance between model performance and efficient inference, with promising implications for real-time applications.

Impact of image scale

In this section, we present a comprehensive study to assess the robustness of our model with respect to input image scale. In practice, we shrink the images at various ratios (1, 1/2, 1/4, and 1/8), and the square crop used as input to the model is also scaled accordingly. Hence, the model has access to the same contextual information independently of the scale.

From the results in table 5.5, we observe that the proposed height method exhibits remarkable robustness concerning image resolution. Despite shrinking the images up to 1/8th of the original size, the model's performance remains consistent. This robustness stands in stark contrast to the approach presented in the previous section where high-resolution images were a prerequisite for achieving accurate ball localization.

Benchmark on ballistic dataset

We also evaluate our method on the *Ballistic-dataset* featuring unoccluded moving balls. The obtained Mean Absolute Projection Error (MAPE_[m]) on

5.3. Height approach

Scale ratio	De-facto image resolution	MAPE _[m]
1	512 × 512	1.25 ± 0.11
1/2	256 × 256	1.33 ± 0.07
1/4	128 × 128	1.30 ± 0.06
1/8	64 × 64	1.28 ± 0.10
1/16	32 × 32	1.48 ± 0.03

Table 5.5: **Performance of the model with respect to input image size.** reducing image resolution (while keeping a context size identical, equivalent to 512 × 512 at original resolution) doesn't affect the performances much up to one eighth of the original resolution (in each dimension).

this dataset is 2.68 ± 0.12 m, which is significantly higher than the results obtained on the DeepSport testset; see table 5.3. The poor performance on this evaluation set is promptly explained by observing the distinct distribution of ball heights in the two datasets; see figure 5.10. The *Ballistic-dataset* shows a skew towards higher ball heights, with nearly half of the samples (102/233) having balls above 2 meters, while the *DeepSport-dataset* primarily comprises balls below this mark, with only 60 training samples reaching 2 meters.

To mitigate this limitation, we attempted to balance the number of balls above and below 2 meters in the training set. This approach resulted in an improved performance on the ballistic dataset, reducing the MAPE_[m] to 2.21 ± 0.27 m. Nonetheless, the limited number of high-ball samples in the training dataset remains a problem: feeding the same few high-ball samples to the neural network many times during training can not help much the model to generalise to those situations. We anticipate that a larger dataset, featuring more samples with high-ball instances, could narrow the gap between the performance on the *Ballistic-dataset* and the DeepSport testset.

5.3.5 Discussion

In this section, we introduced another method for 3D ball localization, achieving impressive results with a projection error as low as 1.1 meters. Notably, our proposed approach exhibits remarkable robustness with respect to image resolution, allowing the use of low-resolution images with the same crop size, thereby providing more context while maintaining good performance and inference speed.

Although our method demonstrates significant improvements in various

Chapter 5. Ball 3D localization from a single calibrated image

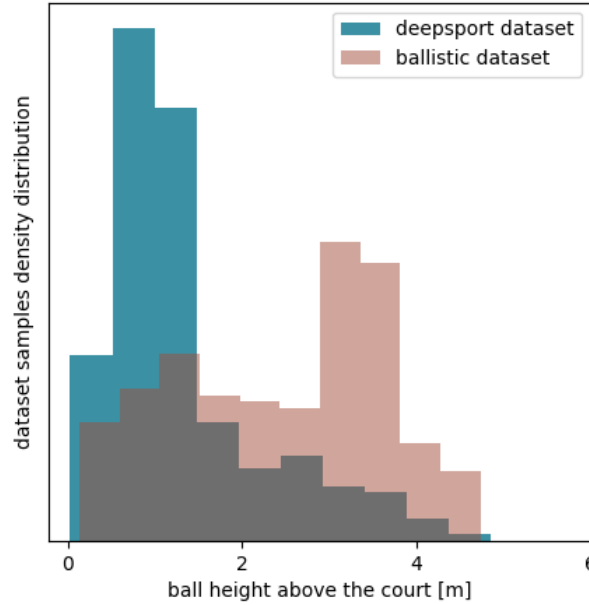


Figure 5.10: **Ball height distribution in our datasets.** The *DeepSport-dataset* (training and evaluation sets combined) features much less images below 2 meters compared to the *Ballistic-dataset*.

scenarios, it underperforms on the *Ballistic-dataset*. This outcome is not surprising given its skewed distribution of ball heights, which significantly differs from the one on which our models were trained. However, only a proper study with a large dataset containing enough high balls could reveal how our height approach is limited in conditions where the ball is high above the court.

5.4 Deeper Comparison

In this section, we evaluate both approaches for 3D ball localization while using the full potential of our datasets. For training, we used the larger *DeepSport-dataset-extended* combined with the annotated ballistic trajectories from the *Sequences-dataset* which include high balls. For evaluation, we used both the *DeepSport-testset* and the *Ballistic-dataset* (see table 5.1).

Table 5.6 shows the evaluation of the two approaches using the mean

5.5. Conclusion

absolute projection error (MAPE), in meters, on both evaluation sets. Corroborating results from table 5.3, the height approach is consistently better than the diameter approach when evaluated on the eclectic *DeepSport-testset*. Moreover, it appears that training on a larger dataset which includes high balls significantly reduces the gap between the *Ballistic-dataset* and the *DeepSport-testset* for the height approach, making it consistently better on all evaluation sets. We evaluated the correlation between the errors from the two approaches using Pearson coefficient. As our analysis reveals, there is no correlation between the errors from the two approaches.

	<i>DeepSport-testset</i>	<i>Ballistic-dataset</i>
Diameter estimation	$3.38 \pm .46$	$1.48 \pm .33$
Height estimation	$0.69 \pm .04$	$1.19 \pm .06$
Pearson coefficient	0.06	0.17

Table 5.6: **MAPE for both approaches and Pearson coefficient for their correlation.** When using the full potential of our datasets, the height approach performs better, on both evaluation sets, with a mean absolute projection error down to 70 centimeters on our *DeepSport-testset*.

5.5 Conclusion

This chapter has presented the first methods to localize a ball in the 3D space from a single monocular image and work on any image where the ball is (event partly) visible, provided that we have camera calibration (which is required for 3D localization).

The first method estimates the ball diameter in the image space and uses knowledge of real ball diameter (which is generally known for many applications) to infer the 3D position of the ball. The second method doesn't even require this knowledge and can be readily extended to other sports, given the creation of similar datasets tailored to each sport. Moreover, its underlying concept of height detection in the image space and subsequent 3D position reconstruction holds promise for general 3D object localization.

These image-based approach provides raw estimations that could be improved based on temporal filtering when needed. Investigation of such temporal regularization is presented in the next chapter.

Overall, this work opens up new possibilities for enhanced ball tracking

Chapter 5. Ball 3D localization from a single calibrated image

and understanding, advancing computer vision in the sport domains and possibly beyond.

Chapter 6

Ball 3D localization over ballistic trajectories

In many sports, the 3D position of the ball is an important resource to feed games analysis, but knowing its precise position during ballistic trajectories is of even greater importance. Indeed, the detection and characterization of ballistic trajectories carries meaningful information, enabling shots characterization or passes accuracy estimation.

In this chapter, we exploit a physical motion model to improve individual frames predictions during ballistic trajectories. To achieve this, we leverage the diameter-based ball 3D localization approach presented in section 5.2.1 which shows good performance, including when the ball is high (see section 5.3.4), and propose a simple approach based on a sliding window and a RanSaC filtering algorithm to detect and fit ballistic trajectories.

6.1 Related works

Detecting and exploiting ballistic motion for ball 3D localization has been studied for many sports like volleyball (Cheng et al., 2018; Chen et al., 2012), soccer (D’Orazio et al., 2009; Ren et al., 2008), basketball (Chakraborty and Meher, 2013; Parisot and De Vleeschouwer, 2019; Chao et al., 2023) and more (Chen et al., 2009; Pingali et al., 2000; Lampert and Peters, 2012; Chao et al., 2023). Most of these works rely on a multi-camera acquisition setup which goes beyond the working hypothesis of this thesis.

Chapter 6. Ball 3D localization over ballistic trajectories

When only a single viewpoint is available, some works detect ballistic trajectories by fitting a generic parabola to the detections in the image space, ignoring the physic governing the motion and making the hypothesis that the camera sensor aligns with verticals (Chakraborty and Meher, 2013; Chen et al., 2012). However, this assumption is too strong for wide angle lenses or cameras tilted along their roll axis, like the ones used in the Keemotion automated capture system (Delannay, 2012). Moreover, because they ignore time, these approaches particularly struggle to identify precisely the ballistic trajectories boundaries when the ball is held or controlled by players but lies within the trajectory parabola. Addressing the concern of boundaries detection, another work detect ballistic trajectories from soccer games by assuming an intersection with the ground (Kim et al., 1998). While this hypothesis probably holds for many ballistic trajectories in soccer games, it is definitely not applicable to sports like volleyball or basketball.

Finally, some works simply ignore the question of ballistic trajectories detection. They only fit a ballistic motion model on balls that are known to be in free fall and all use a simple parabola motion model, ignoring air friction. Ribnick et al. (2009) limit their study to an observation of the model's initial conditions and their deviation with respect to the ground-truth over 3 trajectories. And, posterior to the experiments conducted in this thesis, Chao et al. (2023) use synthetic data generated from ground-truth positions of basketball free-throw shots, without considering any false positives or missed detections, offloading the task of filtering outliers.

Overall, the only work addressing ball 3D localization over long sequences from a single viewpoint, exploiting a motion model during ballistic trajectories without any assumptions on their boundaries, is the one from Parisot and De Vleeschouwer (2019) from which this work is inspired. They use a sliding window with a random sample consensus algorithm (RanSaC) leveraging a simple parabola motion model (ignoring air friction) to detect ballistic trajectories from instantaneous detections. In their single-view scenario, the use of a physics grounded motion model enable ball 3D localization from 2D detections during ballistic trajectories. Nevertheless, as demonstrated with manually annotated 2D positions in section 3.4, this approach fails to produce good 3D trajectories.

In this chapter, we present a method for ball 3D localization over long sequences from a single viewpoint, and exploit a physical motion model to improve initial 3D localization during ballistic trajectories. We validate our method on basketball sequences from the *Sequences-dataset* presented in section 3.2.2.

6.2 Method

Our method works with three consecutive stages depicted in figure 6.1. The first two stages process individual instants (images or sets of images captured at the same instant) to detect and characterize the ball in the scene. The last stage combines detections from consecutive instants to detect ballistic trajectories and fits a ballistic motion model to regularize and thereby improve the initial individual 3D localization. Each stage is described further hereafter.

6.2.1 Stage 1 - Ball candidates detection

Ball candidates are detected on individual instants using a simple object detection model. Instead of producing a single detection, the model produces k candidates per instant. In practice, we propose to use *BallSeg* presented in chapter 4, with a detection rule producing $k = 4$ candidates.

6.2.2 Stage 2 - Ball candidates characterisation

Extending the work presented in section 5.2.1, a neural network model processes the image patch around each candidates and is trained to produce three outputs simultaneously. First, it predicts the presence of a ball in the image patch with a confidence score. Only, the candidate with the highest score is kept and the others are discarded. Second, it predicts the ball diameter in pixels which, combined with real ball diameter and using equation 5.6, enables an initial estimation of ball 3D localization. Third, it predicts whether the ball is flying or not.

In practice, we use a CNN similar to the one presented in section 5.2.1. It is composed of a VGG 19 (Simonyan and Zisserman, 2015) backbone, whose features are flattened, and three heads, one per output. Inspired by LeCun et al. (1998), each head is a succession of three fully connected layers whose final layer is adapted to the task it addresses. The ball presence detection task is supervised with a binary cross entropy loss $\mathcal{L}^c$ from equation 5.1. The ball diameter regression task is supervised with Huber Loss (Huber, 1964) $\mathcal{L}^d$ defined by equation 5.2. Finally, the ball state classification task is supervised with the softmax cross entropy loss defined by

$$\mathcal{L}^s(\mathbf{s}, \hat{\mathbf{s}}) = - \sum_{i=0}^{|\mathcal{S}|} s_i \cdot \log \hat{s}_i \quad (6.1)$$

Chapter 6. Ball 3D localization over ballistic trajectories

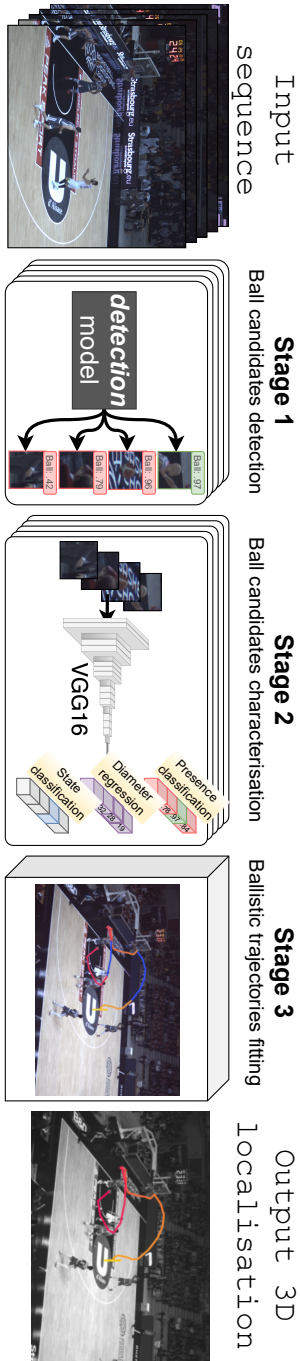


Figure 6.1: **Method for ball 3D localization with ballistic trajectory filtering.** We propose a three stages approach. **Stage 1:** k ball candidates are detected on individual instants using a simple object detection model. In this figure, the top $k = 4$ candidates are kept. **Stage 2:** Image patches around the k ball candidates are processed to: (i) Predict the presence of a ball with a confidence score. The candidate with the highest score is retained. (ii) Predict the ball diameter in pixels. Using calibration data, it can be converted into an initial 3D localization. (iii) Predict whether the ball is flying or not with a classification output. **Stage 3:** A sliding window with a ballistic motion model fitting algorithm processes the individual ball detections. When it detects a ballistic trajectory, the associated motion model is evaluated at the instants' timestamps to improve the initial predictions.

where $\mathcal{S}$ is the pre-defined set of classes among which the ball should be classified, $\mathbf{s}$ is the one-hot encoded target vector where the component $s_i = 1$ for the correct class i and $s_i = 0$ for all other classes, and $\hat{\mathbf{s}}$ is the network prediction vector.

6.2.3 Stage 3 - Ballistic trajectories fitting

Inspired by the work from Parisot and De Vleeschouwer, the third stage builds on a sliding window with a ballistic motion model fitting algorithm that processes the individual ball detections.

Considering μ milliseconds of detections, the window is sled forward in time until a ballistic trajectory is detected. The window is then iteratively extended with additional detections, as long as the fitting algorithm successfully fits a ballistic motion model to the data. Once the fitting algorithm fails, the ballistic motion model is evaluated at every instants of the trajectory to update the initial 3D localization. Finally, oldest detections are dropped until the sliding window is shrunk to its original size and the process ultimately starts again with the successive detections.

The fitting algorithm used by the sliding window is responsible to detect and fit a ballistic motion model. It processes a given window of detections and operates in four successive steps.

1. **State-based filtering.** First, the presence of flying balls is evaluated in the window. If less than q detections are classified as flying by the previous stage, the fitting algorithm is interrupted, no model is returned, the window is moved forward in time.
2. **Initial model prediction.** A RanSaC algorithm estimates a motion model from the random sample of n detections that maximizes the number of inliers, with h iterations. Since the identification of inliers doesn't require a complex ballistic model, we propose to use a simple parabola model, described by equation 2.21. Furthermore, to eliminate the effect of noisy depth estimations from the previous stage, fitting is performed in the image space by projecting predictions from the parabola model using known calibration. Hence, initial detections are considered inliers if they lie below r pixels away from the model evaluated at the corresponding timestamp. In practice, similar to the work from Parisot and De Vleeschouwer (2019), we set the number of samples to $n = 4$, with $h = 100$ iterations.
3. **Inliers-based filtering.** To proceed with the last step, we propose to

Chapter 6. Ball 3D localization over ballistic trajectories

discard windows for which (i) the first sample is not an inlier, (ii) the number of inliers is lower than m and (iii) the ratio of inliers relative to the number of samples between the first and last inliers is lower than ρ .

4. **Motion model fitting.** The final motion model, returned by the fitting algorithm, is evaluated from the set of inliers. Unlike other related works that use the simple parabola model. We propose to use a more realistic motion model described by equation 2.25. Furthermore, instead of optimizing the 3D euclidean distance as done in other works, we work with an error function ϵ^* that disentangles the 2D ball detection error made by the first stage, from the diameter regression error made by the second stage. Hence, the error used in the optimization process uses a weighted sum of the position error ϵ_p , which corresponds to the euclidean distance in the image space between the initial estimation of ball center and the model evaluated at the corresponding timestamp, and the diameter error ϵ_d .

$$\epsilon^* = \epsilon_p + \zeta \cdot \epsilon_d \quad (6.2)$$

In summary, compared to the work from Parisot and De Vleeschouwer (2019), we propose to (i) filter windows using an estimation of ball state, (ii) use a more realistic motion model that considers drag and lift caused by air friction to fit the instantaneous observations, (iii) and fit the motion model with an error function that decouples image position error from depth estimation error.

6.3 Evaluation procedure

Implementation details and metrics used to evaluate our method are presented hereafter.

Sequences dataset We evaluate our method using two sequences from the *Sequences-dataset*, where ball 3D position was annotated during ballistic trajectories using our method presented in section 3.4. Sequence with arena label KS-FR-GRAVELINES is used for meta-parameters search and sequence with arena label KS-FR-BROURGEB is used for final evaluation. Images with those arena labels were all excluded from the training and validation of the subsequent models.

6.3. Evaluation procedure

Ball detection We trained *BallSeg* on *DeepSport-dataset* and *DeepSport-dataset-extended* with the training procedure described in chapter 4. At inference, on the sequences dataset which was captured by multiple cameras (see section 3.2.2), the set of images covering the scene and captured at the same instant were stitched to produce one single image on which *BallSeg* was applied. This approach makes best use of the global aggregation layer that helps making a decision on the whole scene rather than splitting the decision for each image.

Ball characterization The VGG network was trained in two phases. First, the backbone and the two heads corresponding to the ball presence classification task and the ball diameter regression task were trained jointly, following the training schedule presented in section 5.2.3, on *DeepSport-dataset* and *DeepSport-dataset-extended*. Again, $\mathcal{L}^d$ was ignored for non ball inputs. Then, the backbone parameters were frozen and only the state head was trained, supervised by $\mathcal{L}^s$. To achieve this, we used the *Sequences-dataset* in which ball state was annotated, and used our ball detector *BallSeg* with a strict detection threshold to generate $\sim 50k$ pseudo-ground-truth ball thumbnails of known state.

Sliding window Working with basketball, we consider a sliding window of $\mu = 500$ milliseconds which limits detection of dribbles and very short passes.

Fitting algorithm The other parameters were selected empirically using the validation sequence from KS-FR-GRAVELINES. We performed a large meta parameter search, optimizing the ballistic model projection error. The parameter $\zeta = 0.1$ was selected from fitting oracle ballistic trajectories, and $r = 3.0$, $m = 7$, $\rho = 0.3$ and $q = 1$ were selected by processing the whole sequence.

Metrics The main evaluation metric is the MAPE presented in chapter 5 which measures how far the detected balls are from the ground-truth on the horizontal plane. With cameras placed at various elevation angle relative to the court, this error eliminates the vertical component of the 3D error, producing a metric oriented to top view analysis of the game. In addition, we also measure the recall of our ballistic fitting algorithm, i.e. its ability to actually detect when the ball is flying. We ignore false positives for

Chapter 6. Ball 3D localization over ballistic trajectories

two reasons. First, several short passes are detected despite not having been annotated. Second, during annotation of ball state, annotator were instructed to be very strict with the class for balls being flying, resulting in shorter trajectories than what they actually are.

Baselines We evaluate our proposed method by comparing it with the individual instants ball 3D localization. In addition, we also evaluate the benefit of using a realistic ballistic motion model compared to the simpler parabola model adopted in all related works. Our custom error function ϵ^* used by the fitting algorithm was also challenged and compared against the more naive error ϵ^e which minimizes the 3D euclidean distance between the instantaneous predictions and the motion model. Finally, we also evaluated the performance of the fitting algorithm on oracle trajectories where the trajectories boundaries (start and end timestamps) were provided using annotations. This eliminates any possible impact of the sliding window algorithm on the metric.

6.4 Results

Fitting algorithm strategies

Table 6.1 presents the results for the different configurations. Detecting and fitting ballistic trajectories improves over a baseline where balls' 3D localization are predicted individually, with a MAPE reduced from 1.45 meters to 1.06 meters for the **Ballistic (with lift)** motion model. Fitting oracle trajectories provides an even better prediction, with an MAPE 20% to 25% lower than when using the sliding window to detect them.

Using our custom fitting error function ϵ^* leads to a consistent improvement compared to the naive euclidean distance on 3D coordinates ϵ^e used in other works, with a MAPE going as low as 79 centimeters when using the more realistic **Ballistic (with lift)** motion model on clean oracle trajectories. This observation is not surprising since our initial prediction of ball 3D localization is very noisy in the depth dimension by the nature of our approach which estimates the ball diameter in the pixel space. Nevertheless, this confirms that using a custom error function tailored to the task and the data is capital for achieving good results.

It appears that using a more realistic motion model doesn't improve significantly the ability to model the true ballistic motion from noisy de-

6.4. Results

tections, as the **Ballistic (with lift)** motion model only improves over the simple **Parabola** model in the case of oracle trajectories.

Trajectories detection

Table 6.1 shows that whatever the approach: using the naive euclidean distance error ϵ^e or our custom error function ϵ^* , the sliding window algorithm manages to detect ballistic trajectories with a recall of 0.55, meaning that close to half of the balls within ballistic trajectories are missed by our algorithm.

Ball state classification

Evaluation of the ball state classification task alone reveal that our model reaches a precision of 0.8 for the flying class on the testing set, meaning that 20% of flying balls are not classified as such. For this reason, as we implement it, the state estimation alone is not reliable enough for the segmentation of ballistic trajectories. Moreover, another experiment revealed that ignoring the ball state filtering step doesn't affect the metrics. However, if available, this information benefits the overall method by skipping windows in which no single ball is classified as flying, resulting in an algorithm

	Motion model	Error function	MAPE	Recall
Oracle trajectories	Parabola	naive ϵ^e	0.84	1.0
		ours ϵ^*	0.82	1.0
	Ballistic (with lift)	naive ϵ^e	0.85	1.0
		ours ϵ^*	0.79	1.0
Sliding window	Parabola	naive ϵ^e	1.10	0.53
		ours ϵ^*	1.05	0.55
	Ballistic (with lift)	naive ϵ^e	1.10	0.52
		ours ϵ^*	1.06	0.55
Individual frames without motion model fitting			1.45	N/A

Table 6.1: **Evaluation of our three stages method to predict ball 3D position within ballistic trajectories.** The parabola model is compared against a more realistic model that takes air friction into account. Our custom error function ϵ^* compares favourably against the naive euclidean distance ϵ^e generally adopted.

Chapter 6. Ball 3D localization over ballistic trajectories

approximately 3 times faster on a single CPU core.

6.5 Discussion

Our three stages method offers a good approach for ball 3D localization over long sequences, with clear improvements over a baseline that doesn't build on ballistic trajectories to improve individual predictions. Nonetheless, our sliding window algorithm shows limitations in detecting correctly the trajectories, with a recall limited to 55%.

We explored several approaches to improve our method. For instance, we considered training the three tasks of the second stage jointly, using the implementation of Kendall and Gal (2017) which has shown good results for multitask learning. However, despite our efforts, we weren't able to improve the state classification task without affecting the ball diameter regression task, fundamental component of our method. Also, for the third stage, several variations were experimented, without improvements. Like many hand crafted algorithms, countless modifications could be implemented, making it more complex and presumably better. However, we decided to limit our efforts and focus on the evaluation of (i) our tailored error function ϵ^* against the euclidean distance ϵ^e used in all other works, and (ii) the comparison between a realistic motion model that consider air friction and the simple parabola used in all other works. Admittedly, improvements over our proposed method are definitely possible, but go beyond the objectives of this chapter.

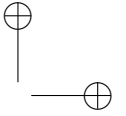
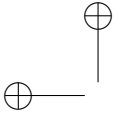
6.6 Conclusion

This chapter presents a three stages method to detect and localize ball in 3D from a single viewpoint. To achieve this, we propose to estimate ball 3D localization on individual frames by combining an estimation of its diameter in the image space with knowledge of real ball size and camera calibration data, and apply a ballistic trajectory fitting algorithm to improve the localization during long passes and shots.

Validation on a basketball sequence reveals that our method reaches an error of ~ 1 meter on ballistic trajectories. When the ball is held by players or dribbled, we can expect an error of 1.8 meters from the results presented in the previous chapter. This might appear as being a lousy approximation. However, in many applications (like 3D analysis of the play

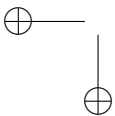
6.6. Conclusion

to collect statistics), having an automated estimation of ball 3D localization from a single camera can be extremely valuable, even with an imprecision of two meters.

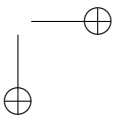


—

—



|



Chapter 7

Conclusion

This thesis addressed the challenging task of object 3D localization from a single viewpoint, focusing on basketballs in team sport scenes. Prior to this work, this task had only been addressed with multiple viewpoints or on very controlled environments with private datasets. Thanks to our strong partnership with Keemotion, world leader in autonomous production of basketball games, we created a unique dataset of unmet diversity and made it publicly available for the benefit of the research community. We used our challenging dataset to tackled two essential problems: ball detection and ball 3D localization. The key contributions of each chapter are summarized hereafter. In addition, we discuss some high level challenges met, specific to this thesis, and provide some perspectives for future works that could build upon this one.

7.1 Results summary

In **chapter 3**, we addressed the important task of collecting datasets of ball 3D position. To annotate the images, captured from a single viewpoint, we developed a method specific to the annotation of basketballs' 3D position when only a single viewpoint is available. Our method can be adapted to any object or point as long as calibration is known or can be inferred from the images. We also studied different approaches for the annotation of multiple consecutive images when the ball follows a ballistic trajectory, with the goal of minimizing annotation efforts. We showed that (i) 2D annotations are not sufficient to reconstruct the 3D trajectory correctly (ii)

Chapter 7. Conclusion

for best results, a realistic ballistic motion model that considers air friction is necessary, and (iii) the manual annotation of 3 points is enough to produce good annotations of 3D positions along the whole trajectory. The datasets created with our annotation method are of unmet diversity and made freely available for the research community.

In **chapter 4**, we presented a segmentation approach to object detection when there's only one single object of interest in the scene, and evaluated it on the detection of basketballs. Our method leverages a fast multi-scale CNN that combines a wide shallow branch capturing fine image details through a reduced number of layers, with two deep narrow branches that efficiently captures global context. In addition to being faster, it compares favourably against a state-of-the-art bounding-boxes object detection approach. To provide a sense of motion in the scene without the delay or the complexity induced by the processing of sequences, we propose to provide two consecutive images. In particular, the image of interest along with the channel-wise pixel difference with the other image. We show that this approach helps detecting balls when its visual appearance has low contrasts with the background while being in motion, as often encountered in basketball scenes. We also validated the benefits of test-time data-augmentation, which enables increasing the detection accuracy at the cost of additional complexity.

Chapter 5 addressed the central part of this thesis, object 3D localization from a single image, and presented two different approaches evaluated on basketballs. Both approaches rely on the detections provided by a detector in the image space and process smalls crop around the detections to estimate the 3D positions. While the first approach is limited to objects of known dimension by inferring 3D from an estimation of their size in the image space, the second uses an estimation of their height relative to the ground floor. Hence, by dropping any assumption on the objects properties, the second approach is transferable to other domains beyond sport. The first approach is more robust to high balls with a mean absolute projection error of 1.7 meters on our dataset of ballistic trajectories. The second approach is better on common basketball scenes with a mean absolute projection error as low as 1.25 meters.

Finally, in **chapter 6**, we applied our single viewpoint ball 3D localization method to sequences and showed that fitting a ballistic motion model during ballistic trajectories improves on the individual predictions. To detect ballistic trajectories, we used a sliding window with a RanSaC algorithm. For the fitting algorithm, we proposed a custom error function adapted to

7.2. Challenges

the single viewpoint context that decouples the small image position errors from the large depth estimation errors, and showed that it outperforms a naive approach that simply uses the euclidean distance.

7.2 Challenges

From the initial stages of this thesis where we were confronted to very practical constraints such as the capture of large volumes of data with limited resources during live productions, to the various difficulties brought by the training of neural networks, this thesis has been strewn with challenges. Among them, two in particular impacted our work significantly.

Despite the direct access to the Keemotion capture systems, our research faced many difficulties stemming from a scarcity of annotated data. On the instants dataset, among the 5000 instants captured, two third were filtered out due to no game being played, resulting in a much smaller dataset than anticipated. As a consequence, our results generally suffered from high variance. To circumvent this limitation, we had to repeat experiments multiple times, increasing the complexity of the research process. Moreover, some research questions even had to be dropped. Furthermore, limitations related to image rights prevented the publication of the full dataset, which entailed many time lost operating with two datasets sizes: one on which experiments could be reproducible by the research community, and one which provided more consistant results. Similar limitations also rose with our dataset of sequences where additional annotation campaigns were deemed not necessary but would actually have made our work easier and probably faster. Hence, when collecting data for machine learning algorithms where data scarcity is a know issue, it is critical to properly plan the acquisition and annotation campaigns.

One other important difficulty met during this thesis has been the will to marry divergent objectives. Indeed, this research has been driven by two very different goals: on the one hand, the completion of a doctoral training, which requires focusing on unique research questions, well positioned against the state of the art, studying each component thoroughly, and sometimes without clear conclusion; and on the other hand, the will to improve the Keemotion automated capture system, tied to specific tasks and datasets, and which implicitly brought implementation considerations along with some expectations of results and efficiency. While it has been enriching to work in this environment, the divergent objectives led to several

Chapter 7. Conclusion

impasses. For instance, while studying neural networks, it has been tempting to address more fundamental research question as they rose. However, undertaking them only delayed the completion of this thesis. Indeed, our very niche tasks with practical whereabouts and specific datasets don't offer the proper environment to study such generic questions which should be addressed with as few hypothesis as possible.

7.3 Perspectives

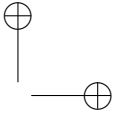
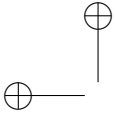
In the end, this thesis sets a solid foundation for the study of single view object 3D localization in sports by proposing the most diverse dataset for ball 3D localization and two baselines working on individual frames. Our baselines rely on initial detections from a conventional ball detector. Albeit our *BallSeg* detector was used in this study, any object detector could be used instead, offering possibly even better performance both in accuracy and latency. Another approach would be the implementation of an end-to-end solution, but it was not considered during this thesis.

For continuous 3D localization of balls, our instant based methods offer two alternatives, possibly complementary, on which a ball 3D tracking algorithm could be built, but this remains an open topic of research. Our attempt using a naive sliding window suffers from the weaknesses of hand-crafted algorithms and should probably be replaced by a learning based approach. In particular, attention-based layers, famous for the processing of temporal relationships, could process a time window, enabling the filtering of individual detections, or even the recovering of missed detections using spatio-temporal consistency. However, this would require a much larger dataset, with additional annotations.

Acknowledgements

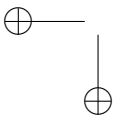
This work has been funded by the Walloon Region project DeepSport and by Sportradar AG which acquired the Keemotion company in 2021. We thank Sportradar AG for enabling and sponsoring this work, and for its commitment to scientific research.

Computational resources have been provided by the supercomputing facilities of the Université catholique de Louvain (CISM/UCL) and the Consortium des Équipements de Calcul Intensif en Fédération Wallonie Bruxelles (CÉCI) funded by the Fond de la Recherche Scientifique de Belgique (F.R.S.-FNRS) under convention 2.5020.11 and by the Walloon Region.

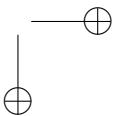


—

—



|



Bibliography

Waleed Abdulla. Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow, 2017. URL https://github.com/matterport/Mask_RCNN.

Oles Andrienko. ICNet and PSPNet-50 in Tensorflow for real-time semantic segmentation, 2019. URL <https://github.com/oandrienko/fast-semantic-segmentation>.

Murat Seçkin Ayhan and Philipp Berens. Test-time data augmentation for estimation of heteroscedastic aleatoric uncertainty in deep neural networks. *Medical Imaging with Deep Learning (MIDL)*, 1(Midl):1–9, 2018. URL <https://openreview.net/pdf/14549241ea4494092dc7f8cecd9e6f4046d84f01.pdf>.

Baljinder Bal and Gaurav Dureja. Hawk Eye: A Logical Innovative Technology Use in Sports for Effective Decision Making. *Sport Science Review*, 21(1-2):107–119, may 2012. ISSN 2066-8732. doi: 10.2478/v10237-012-0006-6. URL https://www.researchgate.net/publication/271314040_Hawk_Eye_A_Logical_Innovative_Technology_Use_in_Sports_for_Effective_Decision_Making.

Yoshua Bengio. Learning Deep Architectures for AI. *Foundations and Trends® in Machine Learning*, 2(1):1–127, nov 2009. ISSN 1935-8237. doi: 10.1561/22000000006. URL <http://dx.doi.org/10.1561/22000000006>.

Y. Lan Boureau, Jean Ponce, and Yann Lecun. A theoretical analysis of feature pooling in visual recognition. In *ICML 2010 - Proceedings, 27th International Conference on Machine Learning*, pages 111–118, 2010. ISBN 9781605589077.

BIBLIOGRAPHY

- DC Brown. Decentering Distortion of Lenses. *Photometric Engineering*, 32(3):444–462, 1966. URL https://books.google.be/books/about/Decentering_Distortion_of_Lenses.html?id=dgbEzgEACAAJ&redir_esc=y.
- John Canny. A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, 1986. ISSN 01628828. doi: 10.1109/TPAMI.1986.4767851.
- Bodhisattwa Chakraborty and Sukadev Meher. A trajectory-based ball detection and tracking system with applications to shooting angle and velocity estimation in basketball videos. *2013 Annual IEEE India Conference, INDICON 2013*, 2013. doi: 10.1109/INDICON.2013.6725963.
- Vanyi Chao, Ankhzaya Jamsrandorj, Yin May Oo, Kyung-Ryoul Mun, and Jinwook Kim. 3D Ball Trajectory Reconstruction of a Ballistic Shot from a Monocular Basketball Video. In *IECON 2023- 49th Annual Conference of the IEEE Industrial Electronics Society*, pages 1–6, Singapore, nov 2023. Institute of Electrical and Electronics Engineers (IEEE). ISBN 9798350331820. doi: 10.1109/iecon51785.2023.10312079.
- Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Return of the devil in the details: delving deep into convolutional nets. *Proceedings of the British Machine Vision Conference 2014*, 2014. URL [http://www.robots.ox.ac.uk/\\$\sim\\$vgg/research/deep_eval/](http://www.robots.ox.ac.uk/$\sim$vgg/research/deep_eval/).
- Fan Chen and Christophe De Vleeschouwer. Autonomous production of basketball videos from multi-sensored data with personalized viewpoints. In *2009 10th International Workshop on Image Analysis for Multimedia Interactive Services, WIAMIS 2009*, pages 81–84, 2009. ISBN 9781424436101. doi: 10.1109/WIAMIS.2009.5031437.
- Fan Chen and Christophe De Vleeschouwer. Personalized production of basketball videos from multi-sensored data under limited display resolution. *Computer Vision and Image Understanding*, 114(6):667–680, jun 2010. ISSN 1090235X. doi: 10.1016/j.cviu.2010.01.005.
- Fan Chen, Damien Delannay, and Christophe De Vleeschouwer. An autonomous framework to produce and distribute personalized team-sport video summaries: A basketball case study. *IEEE Transactions on Multimedia*, 13(6):1381–1394, 2011. ISSN 15209210. doi: 10.1109/TMM.2011.2166379.

BIBLIOGRAPHY

- Hua Tsung Chen, Ming Chun Tien, Yi Wen Chen, Wen Jiin Tsai, and Suh Yin Lee. Physics-based ball tracking and 3D trajectory reconstruction with applications to shooting location estimation in basketball video. *Journal of Visual Communication and Image Representation*, 20(3):204–216, 2009. ISSN 10473203. doi: 10.1016/j.jvcir.2008.11.008.
- Hua Tsung Chen, Wen Jiin Tsai, Suh Yin Lee, and Jen Yu Yu. Ball tracking and 3D trajectory approximation with applications to tactics analysis from single-camera volleyball sequences. *Multimedia Tools and Applications*, 60(3):641–667, 2012. ISSN 15737721. doi: 10.1007/S11042-011-0833-Y/TABLES/9.
- Jianhui Chen, Hoang M. Le, Peter Carr, Yisong Yue, and James J. Little. Learning online smooth predictors for realtime camera planning using recurrent decision trees. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2016-Decem, pages 4688–4696, 2016. ISBN 9781467388504. doi: 10.1109/CVPR.2016.507.
- Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking Atrous Convolution for Semantic Image Segmentation. *Arxiv*, jun 2017. URL <https://arxiv.org/abs/1706.05587><http://arxiv.org/abs/1706.05587>.
- Xina Cheng, Norikazu Ikoma, Masaaki Honda, and Takeshi Ikenaga. Simultaneous physical and conceptual ball state estimation in volleyball game analysis. *2017 IEEE Visual Communications and Image Processing, VCIP 2017*, 2018-Janua:1–4, 2018. doi: 10.1109/VCIP.2017.8305067.
- Harry Collins and Robert Evans. You cannot be serious! Public understanding of technology with special reference to "hawk-Eye". *Public Understanding of Science*, 17(3):283–308, 2008. ISSN 09636625. doi: 10.1177/0963662508093370. URL www.sagepublications.com.
- A. E. Conrady. Decentred Lens-Systems. *Monthly Notices of the Royal Astronomical Society*, 79(5):384–390, mar 1919. ISSN 0035-8711. doi: 10.1093/mnras/79.5.384. URL <https://dx.doi.org/10.1093/mnras/79.5.384>.
- Neville de Mestre. *The Mathematics of Projectiles in Sport*. Cambridge University Press, apr 1990. ISBN 9780521398572. doi: 10.1017/cbo9780521398572. URL <https://www.cambridge.org/9780521398572>.

BIBLIOGRAPHY

org/core/books/mathematics-of-projectiles-in-sport/
8ACF835459654EB5B48027EC9B6DF21B.

Christophe De Vleeschouwer, Fan Chen, and Damien Delannay. Distributed video acquisition and annotation for sport-event summarization. *NEM summit*, 2008. URL http://apidis.org/Dataset/all/metadata/NEM_Summit_APIDIS08_final.pdf<http://hdl.handle.net/2078.1/90154>.

Damien Delannay. The automated production solution connecting the arena for coaches, leagues, and fans | Keemotion, 2012. URL <https://keemotion.com/>.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255. Institute of Electrical and Electronics Engineers (IEEE), mar 2009. doi: 10.1109/CVPR.2009.5206848.

TensorFlow Developers. TensorFlow, feb 2022. URL <https://zenodo.org/record/5949169>.

T. D’Orazio, M. Leo, N. Mosca, P. Spagnolo, and P. L. Mazzeo. A semi-automatic system for ground truth generation of soccer video sequences. In *6th IEEE International Conference on Advanced Video and Signal Based Surveillance, AVSS 2009*, pages 559–564, 2009. ISBN 9780769537184. doi: 10.1109/AVSS.2009.69.

Richard O. Duda and Peter E. Hart. Use of the Hough Transformation to Detect Lines and Curves in Pictures. *Communications of the ACM*, 15 (1):11–15, jan 1972. ISSN 15577317. doi: 10.1145/361237.361242. URL <https://dl.acm.org/doi/abs/10.1145/361237.361242>.

Iván Alén Fernández, Fan Chen, Fabien Lavigney, Xavier Desurmontz, and Christophe De Vleeschouwer. Browsing sport content through an interactive H.264 streaming session. In *2nd International Conference on Advances in Multimedia, MMEDIA 2010*, pages 155–161, 2010. ISBN 9780769540689. doi: 10.1109/MMEDIA.2010.28.

Bradski Gary. The OpenCV Library. *Dr. Dobb’s Journal of Software Tools*, 25 (2236121):120–123, 2015. ISSN 1044-789X. URL <https://www.elibrary.ru/item.asp?id=4934581>.

BIBLIOGRAPHY

- S.A. Seyed Abolfazl S.A. Seyed Abolfazl Ghasemzadeh, Gabriel Van Zanduycke, Maxime Istasse, Niels Sayez, Amirafshar Moshtaghpour, and Christophe De Vleeschouwer. DeepSportLab: a Unified Framework for Ball Detection, Player Instance Segmentation and Pose Estimation in Team Sports Scenes. In *arXiv*, pages 1–10. BMVC, 2021. URL <http://arxiv.org/abs/2112.00627>.
- Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2014. ISBN 9781479951178. doi: 10.1109/CVPR.2014.81. URL <http://arxiv>.
- Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Journal of Machine Learning Research*, volume 9, pages 249–256, 2010. URL <http://www.iro.umontreal>.
- Charles R Harris, K Jarrod Millman, Stéfan J van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, sep 2020. doi: 10.1038/s41586-020-2649-2. URL <https://doi.org/10.1038/s41586-020-2649-2>.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1904–1916, 2015. ISSN 01628828. doi: 10.1109/TPAMI.2015.2389824.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2016-Decem, pages 770–778, 2016. ISBN 9781467388504. doi: 10.1109/CVPR.2016.90. URL <http://image-net.org/challenges/LSVRC/2015/>.
- Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 2017-Octob, pages 2980–2988. IEEE, dec 2017. ISBN

BIBLIOGRAPHY

9781538610329. doi: 10.1109/ICCV.2017.322. URL <http://ieeexplore.ieee.org/document/8237584/>.

Robert Hecht-Nielsen. Theory of the backpropagation neural network. pages 593–605, 1989. doi: 10.1109/IJCNN.1989.118638.

Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks. *Neural Networks*, 3(5):551–560, jan 1990. ISSN 0893-6080. doi: 10.1016/0893-6080(90)90005-6.

Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv*, apr 2017. URL <http://arxiv.org/abs/1704.04861>.

Peter J. Huber. Robust Estimation of a Location Parameter. *The Annals of Mathematical Statistics*, 35(1):73–101, mar 1964. ISSN 0003-4851. doi: 10.1214/aoms/1177703732. URL <https://projecteuclid.org/journals/annals-of-mathematical-statistics/volume-35/issue-1/Robust-Estimation-of-a-Location-Parameter/10.1214/aoms/1177703732.full><https://projecteuclid.org/journals/annals-of-mathematical-statistics/volume-35/issue-1/Robust-Est>.

Intel. Intel True View. *CES 2016*, 2016.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *32nd International Conference on Machine Learning, ICML 2015*, volume 1, pages 448–456, 2015. ISBN 9781510810587.

Paresh R. Kamble, Avinash G. Keskar, and Kishor M. Bhurchandi. Ball tracking in sports: a survey. *Artificial Intelligence Review*, 52(3):1655–1705, 2019. ISSN 15737462. doi: 10.1007/s10462-017-9582-2.

Alex Kendall and Yarin Gal. What uncertainties do we need in Bayesian deep learning for computer vision? In *Advances in Neural Information Processing Systems*, volume 2017-Decem, pages 5575–5585. Neural information processing systems foundation, mar 2017. URL <http://arxiv.org/abs/1703.04977>.

BIBLIOGRAPHY

- Taeone Kim, Yongduek Seo, and Ki Sang Hong. Physics-based 3D position analysis of a soccer ball from monocular image sequences. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 721–726. IEEE, 1998. doi: 10.1109/iccv.1998.710797.
- Diederik P. Kingma and Jimmy Lei Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. International Conference on Learning Representations, ICLR, dec 2015. URL <https://arxiv.org/abs/1412.6980v9>.
- James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences of the United States of America*, 114(13):3521–3526, mar 2017. ISSN 10916490. doi: 10.1073/pnas.1611835114. URL <https://www.pnas.org/doi/abs/10.1073/pnas.1611835114>.
- Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. URL <http://code.google.com/p/cuda-convnet/>.
- K. C.Amit Kumar, Pascaline Parisot, and Christophe De Vleeschouwer. Demo: Spatio-temporal template matching for ball detection. In *2011 5th ACM/IEEE International Conference on Distributed Smart Cameras, ICDSC 2011*, 2011. doi: 10.1109/ICDSC.2011.6042940.
- Mikel Labayen, Igor G. Olaizola, Naiara Aginako, and Julian Florez. Accurate ball trajectory tracking and 3D visualization for computer-assisted sports broadcast. *Multimedia Tools and Applications*, 73(3):1819–1842, dec 2014. ISSN 1380-7501. doi: 10.1007/s11042-013-1558-x. URL <http://link.springer.com/10.1007/s11042-013-1558-x>.
- Christoph H. Lampert and Jan Peters. Real-time detection of colored objects in multiple camera streams with off-the-shelf hardware components. *Journal of Real-Time Image Processing*, 7(1):31–41, 2012. ISSN 18618200. doi: 10.1007/s11554-010-0168-3.

BIBLIOGRAPHY

- Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2323, 1998. ISSN 00189219. doi: 10.1109/5.726791.
- Yann Lecun, Yoshua Bengio, and Geoffrey Hinton. Deep learning, may 2015. ISSN 14764687. URL <http://colah.github.io/>.
- Hao Li, Hanan Samet, Asim Kadav, Igor Durdanovic, and Hans Peter Graf. Pruning filters for efficient convnets. In *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, 2017.
- Tsung Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2014. ISSN 16113349. doi: 10.1007/978-3-319-10602-1_48. URL <http://arxiv.org/abs/1405.0312>.
- Rosanne Liu, Joel Lehman, Piero Molino, Felipe Petroski Such, Eric Frank, Alex Sergeev, and Jason Yosinski. An intriguing failing of convolutional neural networks and the CoordConv solution. In *Advances in Neural Information Processing Systems*, volume 2018-Decem, pages 9605–9616, 2018. URL <https://github.com/uber-research/coordconv>.
- Andrii Maksai, Xinchao Wang, and Pascal Fua. What players do with the ball: A physically constrained interaction modeling. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2016-Decem, pages 972–981, 2016. doi: 10.1109/CVPR.2016.111.
- Davide Mazzini. Guided upsampling network for real-time semantic segmentation. In *British Machine Vision Conference 2018, BMVC 2018*. BMVA Press, jul 2019. URL <https://arxiv.org/abs/1807.07466v1>.
- Vinod Nair and Geoffrey E Hinton. Rectified Linear Units Improve Restricted Boltzmann Machines.
- Jorge Nocedal and Stephen J. Wright. Numerical optimization. In *Springer Series in Operations Research and Financial Engineering*, Springer Series in Operations Research and Financial Engineering, pages 1–664. Springer New York, 2006. ISBN 978-0-387-30303-1. doi: 10.1201/b19115-11. URL <http://link.springer.com/10.1007/978-0-387-40065-5>.

BIBLIOGRAPHY

- Pascaline Parisot and Christophe De Vleeschouwer. Graph-based filtering of ballistic trajectory. In *Proceedings - IEEE International Conference on Multimedia and Expo*, 2011. doi: 10.1109/ICME.2011.6012126.
- Pascaline Parisot and Christophe De Vleeschouwer. Scene-specific classifier for effective and efficient team sport players detection from a single calibrated camera. *Computer Vision and Image Understanding*, 159:74–88, jun 2017. ISSN 1090235X. doi: 10.1016/j.cviu.2017.01.001.
- Pascaline Parisot and Christophe De Vleeschouwer. Consensus-based trajectory estimation for ball detection in calibrated cameras systems. *Journal of Real-Time Image Processing*, 16(5):1335–1350, 2019. ISSN 18618219. doi: 10.1007/s11554-016-0638-3.
- Gopal Pingali, Agata Opalach, and Yves Jean. Ball tracking and virtual replays for innovative tennis broadcasts. *Proceedings-International Conference on Pattern Recognition*, 15(4):152–156, 2000. ISSN 10514651. doi: 10.1109/ICPR.2000.902885.
- Louise-Marie Platteau. *The use of new technologies in Basketball*. PhD thesis, Univertisteit Gent, 2016.
- Rudra P.K. Poudel, Ujwal Bonde, Stephan Liwicki, and Christopher Zach. ContextNet: Exploring context and detail for semantic segmentation in real-time. In *British Machine Vision Conference 2018, BMVC 2018*, 2019.
- Ilija Radosavovic, Piotr Dollar, Ross Girshick, Georgia Gkioxari, and Kaiming He. Data Distillation: Towards Omni-Supervised Learning. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 4119–4128. IEEE, jun 2018. ISBN 9781538664209. doi: 10.1109/CVPR.2018.00433. URL <https://ieeexplore.ieee.org/document/8578531/>.
- Jinchang Ren, James Orwell, Graeme A. Jones, and Ming Xu. Real-time modeling of 3-D soccer ball trajectories from multiple fixed cameras. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(3):350–362, 2008. ISSN 10518215. doi: 10.1109/TCSVT.2008.918276.
- Vito Reno, Nicola Mosca, Roberto Marani, Massimiliano Nitti, Tiziana D’Orazio, and Ettore Stella. Convolutional neural networks based ball detection in tennis games. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, volume 2018-June, pages 1839–1845. IEEE Computer Society, 2018. doi: 10.1109/CVPRW.2018.00228.

BIBLIOGRAPHY

- Evan Ribnick, Stefan Atev, and Nikolaos P. Papanikolopoulos. Estimating 3D positions and velocities of projectiles from monocular views. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(5):938–944, 2009. ISSN 01628828. doi: 10.1109/TPAMI.2008.247.
- Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully Convolutional Networks for Semantic Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4):640–651, 2017. ISSN 01628828. doi: 10.1109/TPAMI.2016.2572683.
- Wenzhe Shi, Jose Caballero, Ferenc Huszar, Johannes Totz, Andrew P. Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2016-Decem, pages 1874–1883. IEEE Computer Society, dec 2016. ISBN 9781467388504. doi: 10.1109/CVPR.2016.207.
- Hubert Shum and Taku Komura. A spatiotemporal approach to extract the 3D trajectory of the baseball from a single view video sequence. In *2004 IEEE International Conference on Multimedia and Expo (ICME)*, volume 3, pages 1583–1586, 2004. ISBN 0780386035. doi: 10.1109/icme.2004.1394551.
- Hugo Silva, André Dias, José Almeida, Alfredo Martins, and Eduardo Silva. Real-Time 3D Ball Trajectory Estimation for RoboCup Middle Size League Using a Single Camera. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7416 LNCS:586–597, 2011. ISSN 03029743. doi: 10.1007/978-3-642-32060-6_50.
- Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. Computational and Biological Learning Society, apr 2015. URL <https://arxiv.org/abs/1409.1556v6><http://www.robots.ox.ac.uk/>.
- Jonas Sköld. *Estimating 3D-trajectories from Monocular Video Sequences*. PhD thesis, KTH ROYAL INSTITUTE OF TECHNOLOGY SCHOOL, 2015.
- Daniel Speck, Pablo Barros, Cornelius Weber, and Stefan Wermter. Ball localization for RoboCup soccer using convolutional neural networks. In *Lecture Notes in Computer Science (including subseries Lecture Notes in*

BIBLIOGRAPHY

Artificial Intelligence and Lecture Notes in Bioinformatics), volume 9776 LNAI, pages 19–30. Springer Verlag, 2017. doi: 10.1007/978-3-319-68792-6_2.

Mingxing Tan and Quoc V. Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In *36th International Conference on Machine Learning, ICML 2019*, volume 2019-June, pages 10691–10700. International Machine Learning Society (IMLS), may 2019. ISBN 9781510886988. URL <https://arxiv.org/abs/1905.11946v5>.

Graham Thomas, Rikke Gade, Thomas B. Moeslund, Peter Carr, and Adrian Hilton. Computer vision for sports: Current applications and research topics. *Computer Vision and Image Understanding*, 159:3–18, jun 2017. ISSN 1090235X. doi: 10.1016/j.cviu.2017.04.011. URL <http://dx.doi.org/10.1016/j.cviu.2017.04.011>.

Gabriel Van Zandycke and Christophe De Vleeschouwer. Real-time CNN-based segmentation architecture for ball detection in a single view setup. *MMSports 2019 - Proceedings of the 2nd International Workshop on Multimedia Content Analysis in Sports, co-located with MM 2019*, pages 51–58, 2019. ISSN 23318422. doi: 10.1145/3347318.3355517.

Gabriel Van Zandycke and Christophe De Vleeschouwer. 3D Ball Localization From A Single Calibrated Image, 2022. ISSN 21607516. URL <https://github.com/gabriel-vanzandycke/>.

Gabriel Van Zandycke, Vladimir Somers, Maxime Istasse, Carlo Del Don, and Davide Zambrano. DeepSportradar-v1: Computer Vision Dataset for Sports Understanding with High Quality Annotations. In *MM-Sports 2022 - Proceedings of the 5th International ACM Workshop on Multimedia Content Analysis in Sports*, pages 1–8. Association for Computing Machinery, Inc, aug 2022. ISBN 9781450394888. doi: 10.1145/3552437.3555699. URL <http://arxiv.org/abs/2208.08190><http://dx.doi.org/10.1145/3552437.3555699>.

Guotai Wang, Wenqi Li, Michael Aertsen, Jan Deprest, Sébastien Ourselin, and Tom Vercauteren. Aleatoric uncertainty estimation with test-time augmentation for medical image segmentation with convolutional neural networks. *Neurocomputing*, 338:34–45, apr 2019. ISSN 18728286. doi: 10.1016/j.neucom.2019.01.103.

Fei Yan, Alexey Kostin, William Christmas, and Josef Kittler. A novel data association algorithm for object tracking in clutter with application to

BIBLIOGRAPHY

tennis video analysis. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 634–641, 2006. doi: 10.1109/CVPR.2006.36.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks?

Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, and Nong Sang. BiSeNet: Bilateral segmentation network for real-time semantic segmentation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 11217 LNCS, pages 334–349, 2018. ISBN 9783030012601. doi: 10.1007/978-3-030-01261-8_20.

Jingzhao Zhang, Tianxing He, Suvrit Sra, and Ali Jadbabaie. Why gradient clipping accelerates training: A theoretical justification for adaptivity. In *8th International Conference on Learning Representations, ICLR 2020*, sep 2020.

Hengshuang Zhao, Xiaojuan Qi, Xiaoyong Shen, Jianping Shi, and Jiaya Jia. ICNet for Real-Time Semantic Segmentation on High-Resolution Images. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 11207 LNCS, pages 418–434. Springer Verlag, 2018. ISBN 9783030012182. doi: 10.1007/978-3-030-01219-9_25. URL <https://github.com/hszhao/ICNet>.

Xiangzeng Zhou, Qiang Huang, Lei Xie, and Stephen Cox. A two layered data association approach for ball tracking. In *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, pages 2317–2321, 2013. doi: 10.1109/ICASSP.2013.6638068.