Aalborg University

Master Thesis Report

Identifying Basketball Plays from Sensor Data; towards a Low-Cost Automatic Extraction of Advanced Statistics

Adrià Arbués Sangüesa

Vision, Graphics and Interactive Systems

Spring 2017

supervised by
Thomas B. Moeslund, Raul Benítez and Chris H. Bahnsen

June 7, 2017

Abstract

Advanced statistics have proved to be a crucial tool for basketball coaches in order to improve training skills. Indeed, the performance of the team can be further optimized by studying the behaviour of players under certain conditions. In the United States of America, companies such as STATS or Second Spectrum use a complex multi-camera setup to deliver advanced statistics to all NBA teams, but the price of this service is far beyond the budget of the vast majority of European teams. For this reason, a first prototype based on positioning sensors is presented. An experimental dataset has been created and meaningful basketball features have been extracted. 97.9% accuracy is obtained using Support Vector Machines when identifying 5 different classic plays: *floppy offense*, pick and roll, press break, post-up situation and fast break. After recognizing these plays in video sequences, advanced statistics could be extracted with ease.
Keywords: Accelerometric Wearable Sensors, Basketball, Player Tracking, Machine Learning, Play Classification, Advanced Statistics.

Contents

Chapter 1

Introduction

Statistics have always been a useful resource to understand the development of sport games. More concretely, in basketball matches, there are a lot of features that can be interesting in order to measure the players' performance; for this reason, a box score is manually generated in every single professional game. A sheet like the one shown in Figure 1.1 contains the following data:

- The number of points a player scored.
- Details about field goals: how many shots a player attempted, how many he/she scored/missed and the type of shot (2-points shot, 3-points shot or free throws).
- The number of rebounds (offensive and defensive) a player grabbed.
- The number of assists a player gave.
- The number of steals/turnovers.
- The number of blocked shots or the times a player was blocked.
- The number of fouls a player committed/drew.
- An evaluation ranking that sums up all the positive aspects and subtracts the negative ones.

BASKONIA 99										REB						TAP			FP			
D	Nombre	Min	P	T2	T2 %	T3	T3 %	T1	T1 %	T	D+O	A	BR	BP	C	F	C	M	F	C	+/-	V
0	Larkin, Shane	31:31	17	2/7	29%	3/7	43%	4/4	100%	1	1+0	9	0	4	0	0	1	0	1	4	-10	16
1	Bargnani, Andrea	10:35	8	1/3	33%	2/4	50%	0/0	0%	1	1+0	0	0	0	0	0	0	0	2	0	-4	3
7	Voigtmann, J.	25:31	10	4/7	57%	0/2	0%	2/2	100%	11	9+2	6	1	4	1	0	0	2	2	1	-7	18
8	Hanga, Adam	31:53	19	3/4	75%	4/5	80%	1/1	100%	4	3+1	2	1	1	2	0	0	0	4	3	-7	22
9	Sedekerskis, Tadas																					
10	Beaubois, Rodrigue	34:39	27	6/8	75%	5/8	63%	0/1	0%	1	0+1	2	1	2	1	2	0	0	4	4	0	25
12	Diop, Ilimane	14:51	2	1/2	50%	0/0	0%	0/2	0%	5	2+3	0	1	2	0	1	0	1	5	5	2	4
14	Tillie, Kim	32:31	8	3/4	75%	0/1	0%	2/2	100%	2	1+1	2	3	1	0	0	0	2	2	2	1	12
15	Laprovittola, N.	12:11	0	0/4	0%	0/0	0%	0/0	0%	1	1+0	2	1	2	0	0	1	0	2	3	12	-2
23	Shengelia, T.																					
34	Budinger, Chase	27:43	8	4/6	67%	0/3	0%	0/1	0%	7	5+2	2	1	0	0	0	1	1	2	1	1	10
55	Luz, Rafa	3:35	0	0/0	0%	0/0	0%	0/0	0%	0	0+0	0	0	0	0	0	0	0	0	0	-8	0
	Equipo		0	0/0	0%	0/0	0%	0/0	0%	2	0+2	0	0	0	0	0	0	0	0	0	0	2
	Total	225:0	99	24/45	53%	14/30	47%	9/13	69%	35	23+12	25	9	16	4	3	3	6	24	23	-4	110
E	Alonso, Sito																					
5f	Diop, Ilimane																					

Figure 1.1: Box score from a Spanish first division game, where the statistics of Baskonia's players can be seen (source `http://www.acb.com/fichas/CREY81005.php`).

Nowadays, the annotation procedure is quite rudimentary: there are up to three annotators per game (paid by the federation that organizes the match) tagging what is happening in the match, either by filling a sheet of paper or by using a button-pad software on their computers. Nevertheless, this procedure has an evident human-error: many actions can occur in few seconds and a brief distraction would result in a loss of information. Although advances had been made in digital basketball statistics (such as the inclusion of a shot-chart or other types of graphs), there is still plenty of room for improvement in this field. Coaches are interested in advanced statistics, which quantify intangible features such as the behaviour of a player depending on his/her physical conditions (relaxed or tired) or the prior probability of success of certain actions. These advanced statistics can also include visual displays, such as heat-maps, which might reflect player tendencies (*i.e* a player that only dribbles to his/her right side).

Many technological solutions have been implemented in basketball games and courts in the last decade, especially in the United States of America. For instance, in 2014, the National Basketball Association (NBA) build the *Instant Replay Center*; *Instant Replay* is a technological solution that allows referees to make better decisions by checking the video of conflicting actions during the game in a screen placed in the officials table. In these headquarters, 94 HD Monitors can be found, and an operator watches the game live, in order to provide the best images to officials in case if required.
Besides, to show different technological sports advances, every year, a *Technology Summit Conference* takes place the same days as the NBA *All-Star* (one of the biggest events of the year in United States); this year, New Orleans held

the 18th edition of this event.

In February 2016, the NBA extended their existing deal with North-American televisions (ESPN and TNT), which meant a notable increase in team salary caps [12]. The salary cap is the total budget a club has in order to build a team; it includes all the club expenses, from the salary of the players and coaching staff to the marketing and data analysis costs. This fact encouraged many general managers to invest a bigger part of the budget in technology. Their goal was clear enough: with an analysis of existing team data, resources can be optimized to win more games. This analysis can be done by using different advanced statistics. An example could be the number of points per game a certain player scores after executing a specific play in road games. Data is a powerful tool for coaches and can be used to identify team's strengths and weaknesses, to scout another team/a certain player or even to prevent injuries.
Within this framework, two companies offering advanced statistics services have recently emerged: STATS (more concretely, Sports VU) [33] and Second Spectrum [32]. Their products are based on a multi-camera configuration system (in the case of STATS, 6 cameras are used), and both companies manage to track all players and the ball at 25 frames per second. It is clear that this service was appealing to both NBA teams and to the league itself: STATS became the official statistics distribution partner and every single stadium has their camera setup installed. Besides, Second Spectrum is the official tracking provider. However, this model is not being used in Europe for a simple reason: the budget. As seen in the prestigious HoopsHype website, the lowest salary cap of a NBA team is 79 million dollars [18], whilst the biggest among European teams is not above 38 millions. As the technological solutions offered by STATS and Second Spectrum are expensive (annual licenses cost 1.100.000 $), alternatives must be found in order to make advanced statistics available to (at least) top European teams.
A cheaper technological approach would imply the use of positioning sensors; such wearable sensors can be conveniently placed in the player's shorts lace or even in their trainers. With these sensors, coaches receive physical data of their players, such as speed or acceleration, and visual statistics like heat maps. Nevertheless, technical and tactical details of the game are not being currently extracted.
The goal of this project is to enrich the sensor data with basketball knowledge by understanding which plays are occurring on court, in order to build a low-cost solution that could provide European teams with similar benefits to the ones the NBA has. For the presented test, positioning sensors are used to track the players, and a new approach to manually tag the ball in *real-time* is designed; having integrated both ball and players' data, different basketball-meaningful features are extracted for each play, thus training a model capable to successfully distinguish between four classic basketball plays: *floppy offense*, pick and roll, post-up situation and press break (a brief playbook can be found in Section 1.3.5).

This report will be organized in the following way: in this same Chapter, a basketball basic glossary can be found, and the different collaborations are explained; right after, in Chapter 2, the state-of-the-art is analysed. Afterwards, the scope of the project is delimitated in Chapter 3, and the design of the whole proposed system is detailed in Chapter 4; results are then evaluated in Chapter 5. Finally, conclusions are extracted and discussed in Chapter 6. Besides, two Appendices can also be found, such as some received entrepreneurship insights (A) and a weekly diary (B).

1.1 Collaboration with Universitat Politècnica de Catalunya-BarcelonaTECH (UPC)

Although this is the Master Thesis of an Aalborg University Program (Vision, Graphics and Interactive Systems), this project has been carried out in the *LASSIE* Lab at UPC, under the supervision of its coordinator, Raul Benitez Iglesias (raul.benitez@upc.edu). The name of the lab stands for Analysis of Interdisciplinary Signals and Systems, and it is one of the research groups inside CREB (Biomedical Engineering Research Centre). There are a total of nine people in *LASSIE*: two Bachelor students, two Master students (including me), four PhD students, and a professor (Raul). Meetings are hold every Friday; in these sessions, all the students present their weekly work the week in order to do some brainstorming, and right after, one of the members of the lab gives a brief talk to explain his/her contributions with more detail.

1.2 Obtaining a Dataset

My initial thesis goal was not only to extract advanced statistics and classify different plays, but also to develop some Computer Vision algorithms to perform multi-tracking of basketball sequences, where all 10 players and the ball would be tracked. However, after analysing the State-of-the-Art, I learnt that designing this kind of algorithm was such a difficult task that could involve several PhD students working on it. For this reason, an important decision had to be made: the goal of the project should be either creating basic tracking algorithms to deal with simple occlusions or *outsourcing* this tracking module and focus on game understanding.
Having discarded the design of the multi-player tracking algorithm, an annotated dataset had to be found, which was way trickier than expected. The first intention was to use one of the existing datasets that could be found on-line, such as the one included in the VATIC annotation tool [35], or the APIDIS [10] or OSUPEL [2] datasets. However, it was not possible to download the full dataset in none of these three cases: the link of VATIC was broken, the APIDIS dataset contained only 2 minutes of a single annotated sequence and no answer was received after asking for the OSUPEL dataset. The most complete dataset

that could be found was created by Appspot [30], but the annotations were done over random plays of one same game and the quality of the annotations was not accurate enough. In Appspot and VATIC datasets, annotations correspond to screen positions, so the exact court spot is unknown. A recap of the individual dataset properties is displayed in Table 1.1.

Dataset	**Content**	**Cameras**	**Coordinates**	**Issue**
VATIC	40 game minutes	1	Screen	Broken dataset link
APIDIS	2 game minutes	6	Court	No answer when asking for the whole dataser
OSUPEL	Unknown	Unknwon	Unknown	No answer when asking for the dataset
DATASPOT	Separate actions of several games	1/game	Screen	Separate plays, difficulties when mapping to court coordinates (zooming, panning...)

Table 1.1: Different annotated datasets found on-line

Having discarded the mentioned options, an emergent Spanish company was contacted: NBN23 [27]. This company provides tracking data to four teams of the Spanish basketball league by placing small sensors in the shorts' cord lace. After talking to them, a collaboration agreement was reached: they would provide me with tracking data (not only sensors data but also some recordings) in exchange of a possible inclusion of the designed methods of this project in their software.

The problem NBN23 is facing right now is the lack of a tracking method for the ball. With the data they are currently using, their software is really appealing for physiotherapists, but coaches do not find useful applications; their platform can help to prevent injuries or to detect the fatigue of the players, but it does not provide advanced statistics. Two approaches can be followed to integrate ball data: either tagging it manually (as it will be done in this report) or integrating a sensor in the ball that could also tell where it is. Actually, companies like Wilson [38] are already trying to integrate Bluetooth sensors in basketball spheres; their prototype was developed together with a mobile application that stores information about shots, such as the shooting arc or the amount of scored/missed shots (it is not 100 % accurate yet). At the moment, the emitted core data cannot be accessed anyhow.

1.3 Basketball Glossary

This section is devoted to provide a detailed description of some technical basketball concepts that are relevant to further understand some of the assumptions of the study.

1.3.1 The game

In a basketball game, two teams face each other. Although the team can have up to 12 players, only five can be on the court, so it is a *5-on-5* match, with each team trying to score in one of the two baskets (rims). In each possession, the team attacking is called the *offensive team*, and the one in charge of preventing it, the *defensive team*. The court measures 28 meters wide, and 15 high. The game is divided into four periods of 10 (Europe) or 12 (United States) minutes, and the clock is stopped every time there is an interruption (out-of-bounds, foul...); in professional games, three officials are in charge of making sure the players do not break the written rules of this sport. At the end of these 40 minutes, the team that scored more points is the winner; if the result is a tie, five minutes of over-time are played.

1.3.2 Basic Actions

Although basketball is a complex game with thousands of tactics that can be used to win the game, the basic important actions can be simplified in the following list:

1. When a player moves with the ball to any spot of the court, it is said that he/she is **driving** to that spot.

2. A **shot** is the action where a player attempts to score. There are different types of shots: layups, where the player runs with the ball and jumps to leave it as close as possible to the basket, short-range shots (closer than 4 meters to the rim), mid-range shots (from 4 to 6 meters away from the basket), long-range (further than 6 meters) and free-throws (from 4.5 meters without defense). If an offensive player (*Player 1*) attempts a shot and a defensive one (*Player 1b*) touches the ball during the ascending trajectory, *Player 1b* has **blocked** the shot.

3. An **assist** is a pass that is then transformed into a scored shot. For example, *Player 1* is driving the ball and he/she sees *Player 2* all alone under the basket; if *Player 1* shares the ball and *Player 2* scores, an assist will be counted for *Player 1*.

4. When there is an unsuccessful shot and the ball touches the rim, the goal of all 10 players on the court is to get the ball. The one who grabs it, will get a **rebound** in his/her statistics. Rebounds can be taken, grabbed or captured.

5. When a defensive player (*Player 1b*) takes the ball from an offensive one (*Player 1*), it is said that *Player 1* committed a **turnover** and that *Player 1b* got a **steal**. Actually, if the player who steals the ball drives so fast to his/her basket that no one can stop him/her from scoring, he/she will have made a **fast break**.

6. When a defensive player (*Player 1b*) hits an offensive one (*Player 1*) while trying to guard him, it is said that *Player 1b* committed a **foul**, which is drawn by *Player 1*. If this foul has been made in a shooting action, *Player 1* will shoot two free throws, thus penalizing defense. Note that players do not call their own fouls, but the officials do it, based on a universal (but subjective) criteria.

1.3.3 Positions

Basketball players have roles or positions according to their skills (technical and physical). A brief general summary of basketball classic positions could be the following one:

- **Point-guard (PG)**: the shortest players of the team with good ball-handles and leadership skills; they organize the team on offense.
- **Shooting-guard (SG)**: fast short players in charge of scoring points; they tend to be good shooters from long-distance.
- **Small-forward (SF)**: the tallest exterior players, apart from having scoring ability, help in rebounds and are committed to defense.
- **Power-forward (PF)**: physical (but not heavy) interior players with a remarkable mid/long-range shot.
- **Center (C)**: the tallest and heaviest players of the team; they are crucial in protecting the rim and grabbing rebounds.

1.3.4 Screens

Screens are another common concept as well, and are usually set by big-players, which stay static in a certain position in order to retain the defender of a guard (fast-small players). Therefore, the small player can take advantage of the lack of a defender for few seconds. After the screen, if the big-player moves towards the rim, it is called *roll*, but if he/she moves to the three-point line, it is called *pop*.

1.3.5 Plays

In order to explain the plays that have been included in the gathered dataset, pictorial representations (like the ones coaches draw in their boards) are shown. Using the icons shown in Figure 1.2, three temporal frames are shown for each play, which explain the movement that is going on during the action. The representations of *floppy offense*, pick and roll, press break, post-up and fast-break situations can be seen in Figures 1.3, 1.4, 1.5, 1.6 and 1.7 respectively.

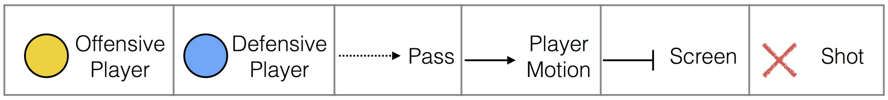

Figure 1.2: Icons used in the 2D representations of basketball plays.

Floppy Offense

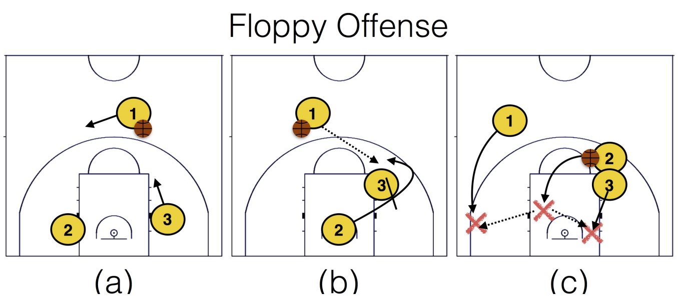

Figure 1.3: Temporal execution of *floppy offense*; (a) *Player 1* creates space and *Player 3* gets prepared to set a screen; (b) *Player 3* screens away *Player 2*, who receives the ball and drives to the basket; (c) *Player 2* ends up deciding if he/she shoots, looks for an open shot of *Player 1* or the roll of *Player 3*.

Pick and Roll

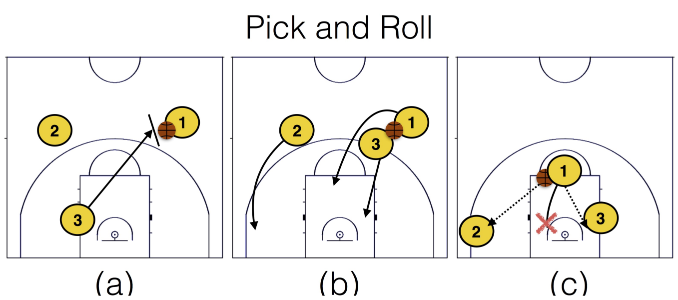

Figure 1.4: Temporal execution of a pick and roll sequence; (a) *Player 1* calls the play and *Player 3* sets him a screen; (b) *Player 1* drives to the basket, *Player 2* looks for a comfortable spot in the corner and *Player 3* continues to the basket; (c) *Player 1* ends up deciding if he/she shoots, looks for an open shot of *Player 2* or the roll of *Player 3*.

Press Break

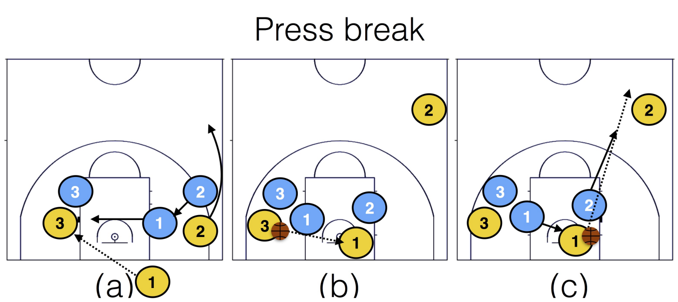

Figure 1.5: Temporal example of a press break situation, where the pass of *Player 1* to *Player 2* overcomes the defensive pressure. Note that there is not a universal way of breaking pressure, as it depends on the defensive team's reaction.

Post-up Situation

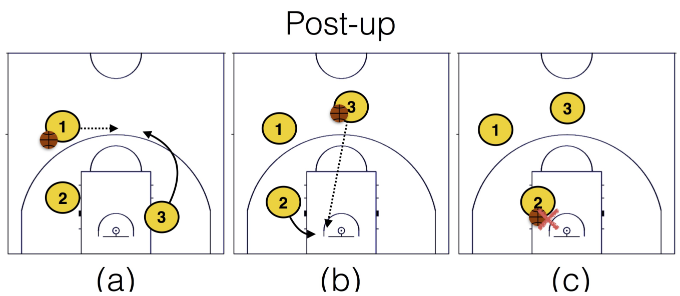

Figure 1.6: Temporal execution of a post-up situation; (a) *Player 1* tries to pass the ball to *Player 2*, but he/she is being guarded, so *Player 3* looks for a better passing position; (b) with a better angle, *Player 2* gives an assist to *Player 3*, who (c) ends up shooting from a close position to the basket.

Fast Break

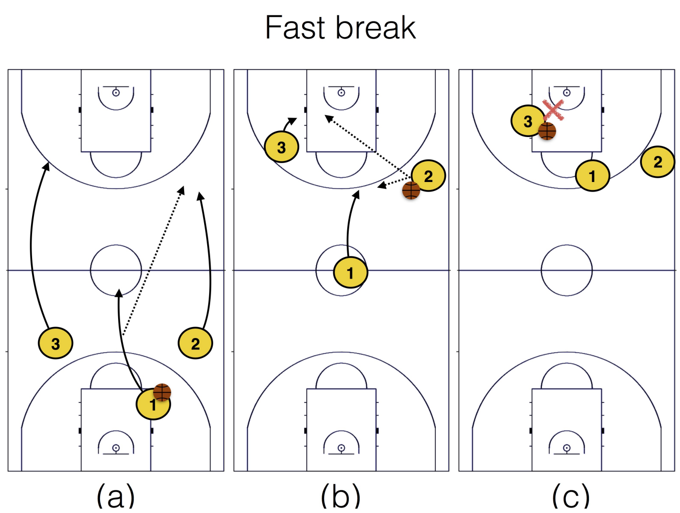

Figure 1.7: Temporal execution of a fast break; (a) when *Player 1* gets the ball, *Player 2* and *Player 3* run and look for an open position; in this example, *Player 2* receives the ball and (b) looks if *Player 1* or *Player 3* are all alone; (c) *Player 3* ends up shooting from a close position to the basket.

Chapter 2

Problem Analysis

As mentioned, the goal of this project is to build a *low-cost* method to extract advanced statistics and classify plays based on tracking data emitted by wearable sensors. Although the field that mixes basketball with Machine Learning and Computer Vision techniques is not the most explored one, several contributions have been made; for clarification purposes, the state-of-the-art techniques are split into the following groups:

1. **Spatial Analysis of Plays**. Based on tracking data, articles included in this group try to reach a high-level understanding of complex basketball concepts, such as ball movement or strategies.

2. **Metrics Quantification**. In this section, different advanced statistics are quantified based on tracking data. Articles are divided into **offensive**, **defensive** and **rebounding** metrics.

3. **Tracking Methods** designed for sport sequences are also analysed in order to compare how different it would be to track players with cameras instead of using wereable sensors.

As it will be observed, the vast majority of the included articles use Sports VU tracking data (provided by STATS). In addition, most of these articles were published before 2014. The reasoning is simple: when STATS delivered data to particular teams, this information was available on demand; nevertheless, no public data is available since the NBA league bought their services.
The collection of summarized articles as well as the different categories papers are split into can be seen in Table 2.1.

Category	Name	Authors	Year
Spatial Analysis of Plays	“How to Get an Open Shot”: Analyzing Team Movement in Basketball using Tracking Data	Lucey et al.	2014
	Classifying NBA Offensive Plays Using Neural Networks	Wang and Zemel	2016
	Possession Sketches : Mapping NBA Strategies	Miller and Bornn	2017
Offensive Metrics	CourtVision : New Visual and Spatial Analytics for the NBA	Kirk Goldsberry	2012
	POINTWISE : Predicting Points and Valuing Decisions in Real Time with NBA Optical Tracking Data	Cervone et al.	2014
	“ Body Shots ”: Analyzing Shooting Styles in the NBA using Body Pose	Felsen, Lucey	2017
Defensive Metrics	The Dwight Effect: A New Ensemble of Interior Defense Analytics for the NBA	Goldsberry et al.	2013
	Counterpoints : Advanced Defensive Metrics for NBA Basketball	Franks et al.	2015
	Recognizing and Analyzing Ball Screen Defense in the NBA Learning to Classify Defensive Schemes	McIntyre et al.	2016
Rebound Metrics	Deconstructing the Rebound with Optical Tracking Data	Maheswaran et al.	2012
	The Three Dimensions of Rebounding	Maheswaran et al.	2014
	To Crash or Not To Crash : A quantitative look at the relationship between offensive rebounding and transition defense in the NBA	Wiens et al.	2013
Tracking	A Template-Based Multi-Player Action Recognition of the Basketball Game	Perse et al.	2006
	An Analysis of Basketball Players’ Movements in the Slovenian Basketball League Play-Offs Using the SAGIT Tracking System	Perse et al.	2008
	Occupancy Analysis of Sports Arenas Using Thermal Imaging	Gade et al.	2012
	Player Tracking and Analysis of Basketball Plays	Chesire et al.	2015
	Detecting events and key actors in multi-person videos	Ramanathan et al.	2016

Table 2.1: Collection of summarized articles in the state-of-the-art section

2.1 Spatial Analysis of Plays

Name	Goal
"How to Get an Open Shot": Analyzing Team Movement in Basketball using Tracking Data	Discover the factors that may explain why shooting percentages drop if a shot is contested (NBA data).
Classifying NBA Offensive Plays Using Neural Networks	Classify a closed-set of plays using a pictorial representation of data as an input of a Convolutional Neural Network.
Possession Sketches : Mapping NBA Strategies	Cluster different kinds of plays grouping structure similarities of tracking data.

Table 2.2: Included articles in the *Spatial Analysis of Plays* section.

In the paper written by Lucey *et al.* [21], the authors analysed how teams manage to have open shots (those shots where the shooter does not have a defender close to him) in order to improve shooting percentages. The motivation of this paper emerged when checking the statistics of the NBA teams, as the authors realized that there is a notable drop in shooting percentages when attempting pressured shots (almost a 15% decrease in some cases). First, their algorithm assigns a role (position) to every player at the beginning of the action. Then, the different factors that may affect when attempting a shot are checked from a more analytic point of view; these include features such as the closest distance from a defender, the speed of the player when he/she shoots or the number of seconds the player kept the ball before shooting. Finally, different plays are retrieved using tracking data, which clusters similar plays into permutations from the original one (the exact same action will not occur twice in a game). Extracted results show that one of the most relevant features is the defending swaps that may occur during the game, also called mismatches. Although it is rather a statistics paper, this project aims to the same goal: to extract relevant information from tracking data to improve the understanding of the game.

Using the same Sports VU raw data, Wang and Zemel [36] designed an algorithm to classify a closed-set of plays using Recurrent Neural Networks (RNN), with the purpose of generating detailed reports with a high-level basketball understanding. Their approach turns tracking data into pictorial representations, in order to deal with an image classification problem. The trade-off of using RNN is that, on the one hand, there is no need of manually extracting features from the set of images, but on the other hand, there is no control over the features the algorithm learns from the examples. Positions of the players are guessed by comparing their shooting tendencies and frequencies in different positions in the court (*e.g* an exterior player usually moves behind the 3-point line and attempts more long-range shots than an interior one), and they build an *anytime* prediction system, as one same play may change due to defensive strategy. Their results (expressed with *top-1 accuracy*) seem to be promising, but the system is thought for a particular team in a specific season, so it is not automatically tuned to any kind of team.

Miller and Bornn [26] made another relevant contribution. They organize a large set of plays by grouping structural similarities, as they observed that there is not an efficient scouting method for professional basketball teams. Their goal is achieved through a segmentation of short plays to shorter manageable segments (modelled with Bezier curves), a possession modelling by adapting topic models and a bag-of-words structure. Finally, having clustered data with nearest-neighbours algorithms, different types of analysis are done. Although the attached videos show promising results, no numerical evidence is displayed. This work is an improvement of a previous contribution of the same authors (Miller *et al.* [25]), where the actions occurring in a basketball court were analysed by a point process factorization based on intensities.

2.2 Metrics Quantification

As it has been mentioned, different kinds of metrics can be quantified once tracking data is obtained: offensive, defensive or more concrete ones, like rebounding metrics.

2.2.1 Offensive Metrics

Name	**Goal**
CourtVision: New Visual and Spatial Analytics for the NBA	Include spatial information in NBA statistics to define what does the concept ”good shooter” really mean.
POINTWISE : Predicting Points and Valuing, Decisions in Real Time with NBA Optical Tracking Data	Try to mathematically model the decision-making process of NBA players during the course of possessions.
“Body Shots”: Analyzing Shooting Styles in the NBA using Body Pose	Find correlations between missed/tough shots and the body position of the shooter in order to complement tracking data.

Table 2.3: Included articles in the *Offensive Metrics* section.

One of the first interesting offensive metrics was introduced by Kirk Goldsberry [16], who presented new visual and spatial analytics to determine who is the best shooter in the NBA. The problem he tries to solve is that the league leader in field goals percentage tends to be a center who takes no mid/long-range shots; therefore, the goal is to define a metric to determine who is the player that shoots better from as many court spots as possible. His system builds a composite shot-map for all the shots attempted in 5 different seasons (2006-2011), finding out 1284 unique shooting cells. Then, spread parameters are defined and weighted by its distance to the basket (number of cells with acceptable accuracy). This metric definitely penalizes those centers that do not take risky shots, and provide a robust knowledge on how well a player shoots. Obtained rankings prove to be precise, as those coincide with the opinion of basketball journalists when talking about the top-5 shooters in the league.

Cervone *et al.* [7] presented a new way to mathematically model how good is the decision-making process of players during a game possession in real-time, as they realized that some teams want players with high *Basketball IQ* to create team-benefits. In order to take this project into account, they define the *Expected Possession Value* (EPV) as a number with the expected points to be scored at any moment; then, having the position of the player driving the ball, they model his/her behaviour by dividing the attitude into macrotransitions (shoot/pass/turnover) or microtransitions (basic movement). With EPV metrics, two applications are shown: (a) a ranking of the NBA players that make better decisions and (b) an equation to measure the shot satisfaction, which can help to identify selfish attitudes, such as a player that attempts a long-range shot being guarded by 3 players when he/she has two open teammates in comfortable shooting positions. Both applications show adequate results, and many more applications could be thought; however, the article misses some details in the EPV computation.

Another article about advanced statistics was presented by Felsen and Lucey [13]. In their study, the goal was to find correlations between different types of shots and the body position of the shooter. Their motivation was to complement the Sports VU data, because taking only coordinates into account, some relevant information may be missed: for example, if a player receives the ball in an open position but the pass goes directly to his/her feet, he/she will attempt a tough shot. Their method includes a quantification of the involved anatomy in a three-point-shot and a machine learning module (using Support Vector Machines [19]), where a model is trained to identify open/tough shots and to attribute correlations by comparing open shots to contested ones, and made shots to missed ones. Furthermore, the authors also perform a deep analysis of the shooting parameters of the best NBA shooter (Stephen Curry), and find out that, although there are many biometric correlated factors in open/tough shots, those cannot be generalized into a single model, as Curry has a notable percentage from long-range, but he attempts more tough shots than the vast majority of players. They prove that integrating tracking coordinates and biometric factors would provide a more realistic and precise model.

2.2.2 Defensive Metrics

Name	Goal
The Dwight Effect: A New Ensemble of Interior Defense Analytics for the NBA	Introduce a new metric that could indicate the ability of an interior defender to reduce shooter's behaviour and efficiency.
Counterpoints : Advanced Defensive Metrics for NBA Basketball	Quantify defensive metrics of NBA players to indicate who is the best (exterior) defender of the league.
Recognizing and Analyzing Ball Screen Defense in the NBA Learning to Classify Defensive Schemes	Detect different kinds of ball screen defensive plays to enable novel analysis of defensive strategies through tracking data.

Table 2.4: Included articles in the *Defensive Metrics* section.

Goldsberry and Weiss [17] wanted to quantify defensive metrics of NBA basketball games. The motivation emerged from the isolation of defensive concepts in NBA box-scores, where only defensive rebounds, steals and blocks are currently being annotated. Their contribution was called *the Dwight Effect*, and they wanted to prove that the leader of the league in blocks might not be the best defender, but the player who changes the shooter's behaviour and efficiency more often. In this article, and using Sports VU data once again, they first separate frequencies and effectiveness of different kinds of shots of every player in the NBA; then, they computer the *basket proximity*, which is the balance between the percentage in field goals and the number of avoided shots when a certain interior player contests the shot. Afterwards, *shot proximity* is estimated by checking how often is an interior player close to a shot attempt. Their results are meaningful from the point of view of a basketball coach, as a single metric summarizes several factors regarding the rim protection. However, this quantification is restricted to interior players. In order to complement this work, Franks *et al.* [14] presented new defensive metrics for exterior players, including the *Volume Score*, which contains the magnitude of shot attempts in front of a certain defensive player, the *Disruption Score* expressing the effectiveness of those shots and *Counterpoints*, which indicates who is responsible for contesting a certain shot. This analysis is based on modelling the evolution of defensive matchups (different swaps when defending a team) over the course of possession as a Markov Model, and the computation of the mentioned metrics using logistic regression plus predicting the *a priori* efficiency of a shot. Results are expressed in rankings, which are pretty accurate: the players that journalists consider the best defenders appear in top positions.

Another interesting quantifiable defensive metric was introduced by McIntyre *et al.* [24], who analysed how NBA teams defend ball screen situations considering 4 different options (over, under, trap or switch). Their goal was to quantify not only which is the most repeated strategy but also the most efficient one. This contribution enables novel analysis of defensive strategies using Sports VU data. Their method has a validation set, that comprises manual annotations of ball screen situations of 6 different basketball games (a total of

199 instances). Then, using an algorithm based on pairwise distances within players, 270853 ball screen situations are tested, obtaining 69% accuracy on three classes (*traps* could not be included because of a small number of samples); besides, the attitudes of the teams are shown, which provide interesting metrics to identify the most aggressive teams in the NBA. If the validation set had been larger, greater accuracy would have been obtained, which could lead to a robust system to be used in professional games.

2.2.3 Rebounding Metrics

Name	**Goal**
Deconstructing the Rebound with Optical Tracking Data	Check all factors that might influence in rebounding actions. Provide a metric that can indicate the kind of rebounds a player grabs.
The Three Dimensions of Rebounding	
To Crash or Not To Crash : A quantitative look at the relationship between offensive rebounding and transition defense in the NBA	Analyse the trade-off between attacking the offensive rebound or retreating back in a defensive position. Evaluate the risk of both strategies.

Table 2.5: Included articles in the *Rebounding Metrics* section.

Besides, other metrics were also introduced to contextualize rebounds with the purpose of numerically identifying if a player captures a rebound all alone or grabs it after hustling with three players. Maheswaran *et al.* [22] deconstructed the rebound by checking the factors that influence in this type of actions. First, they filtered Sports VU data to end up only with rebound observations and build a heat-map with all these locations (around 11000 instances). Right after, rebound location probabilities are checked given the shot position (distance and angle); from these regions, another heat-map is built, containing the coordinates where the ball decreased from 8 feet, which indicates the potential rebound location. Given the position of all players, a Machine Learning module is included in order to predict who has more chances to catch the rebound as the action goes forward. Their results show that in mid-range shots, the probabilities of grabbing an offensive rebound are low, and that there is not a significant directional bias depending on the shot location. Their drawback is that the paper does not summarize everything up into one metric, which would be more consistent and easy to include in statistic websites.
The same authors [23] extended their contribution by analytically decomposing the rebound into three concrete factors. *Positioning* (modelled with a Voronoi region) is used to see the position of a player when: (a) there is a shot and (b) few seconds after it. These coordinates help to indicate the player's intention: he/she can either try to capture an offensive rebound (also known as *crashing*) or retreat to a defensive position. The second factor is *Hustle*, which tells if a player is able to create a rebound opportunity despite not being at the best

initial spot. Finally *Conversion* parameterizes if a certain player allows others to grab rebounds when he/she has the best positioning; that is, if a player, captures easy rebounds or not. Once again, their results are shown in different rankings, and coincide with the experts' opinions. However, this same experts can argue that *Positioning* might not be a skill, but a matter of luck or other factors.

Furthermore, Wiens *et al.* [37] conducted more concrete research to analyse only offensive rebounds, trying to see the trade-off between two strategies: attacking the offensive rebound (*crashing*) and retreating a defensive position. Having filtered Sports VU dataset and gathered only offensive rebound situations after mid/long-range jumpshots, a reaction time is established. Specific metrics are defined: *odds ratio* (probability of a good event to occur) and *net gain*, which indicates the possibility of scoring having grabbed the offensive rebound combined with the possibility of preventing the other team to score having retreated on the defensive end. Once modelled *threat neutralization* (how effective the defensive transition is in terms of pairwise distances between players), results show that *crashing* is a risky strategy, and an early *threat neutralization* limits the negative impact of transitions. Anyway, this article should be tested again with the inclusion of more data, as it only had the strategies of 12 teams (and few observations were obtained for some of them).

2.3 Tracking

<table>
<tr><th>Name</th><th>Goal</th></tr>
<tr><td>A Template-Based Multi-Player Action Recognition of the Basketball Game</td><td rowspan="2">Track all players and the ball with a 2-camera configuration in the ceiling of the arena. Recognize game phases and patterns in order to provide tactical information to coaches.</td></tr>
<tr><td>An Analysis of Basketball Players' Movements in the Slovenian Basketball League Play-Offs Using the SAGIT Tracking System</td></tr>
<tr><td>Occupancy Analysis of Sports Arenas Using Thermal Imaging</td><td>Optimization of the use of a sports arena by tracking players and comparing the occupancy regions of different sports.</td></tr>
<tr><td>Player Tracking and Analysis of Basketball Plays</td><td>Manage to track all players and the ball to map the actions onto a 2D representation.</td></tr>
<tr><td>Detecting Events and Key Actors in Multi-Person Videos</td><td>Combine a multi-tracker with the "focus of attention" of games in different basketball situations.</td></tr>
</table>

Table 2.6: Included articles in the *Tracking* section.

An approach to track basketball players through video processing and perform data analysis was thought by Perse *et al.* [29, 11]. With a 2-camera configuration setup in the ceiling of the arena, a method could be designed in order to help planning training sessions based on players' movements. Their method creates a play-designer module, which contains a playbook of stored templates with different plays. Then, the phase of the game (offensive / defensive / time-out) is found by clustering the distribution of players on court with a Gaussian Mixture Model [34]. Afterwards, the small-scale parts of the game are found: the court is divided into 9 sections and basic events are used in order to define the player motion on the court. Finally, recognition is done by using the stored templates in the play-designer. Although their dataset was not huge, their results are consistent; nevertheless, there is no ball information and the algorithm does not have the possibility of learning new plays on its own.

Other techniques have been used in order to detect the number of players in a sports court, such as in the paper written by Gade *et al.* [15]. By making use of thermal cameras (thus avoiding legal policies), the occupancy of these courts is analysed, with the purpose of creating a method that could optimize the utilisation of an arena. The basis of this project is to record the different activities that take place in a sports arena; then, once four points of the court are manually annotated and the boundaries of it are found, an homography can be estimated in order to map the position of players in the recorded images into real court positions. Another interesting part of the project is the post-processing stage that removes reflections or splits tall/wide regions. Their results seem promising, and different patterns are shown depending on the sport that is taking place. However, the article goal is not to talk about the tactical sport details.

Chesire, *et al.* [8] wrote a paper-proposal based on the analysis of basketball plays once having tracked the 10 players on court; their purpose is to do further analysis once tracking data is projected onto a 2D court. Their OpenCV [4] implementation combines several techniques: first, the court lines are detected by applying a Canny Edge Detection [6] and finding straight lines with the Hough transform [5]. Then, pedestrians are detected using the Histogram of Oriented Gradients technique [9] and classified with Support Vector Machines [19]. Right after, a semi-automatic approach based on the HSV colorspace is applied to identify both teams; actually, the basis of the algorithm is to re-detect pedestrians at each frame, so there is no need for a tracking algorithm. Their results show 2D projections of their detections, and it is a reproducible paper; however, it has some drawbacks: there is no ball information, the colour filter must be tuned manually, there is no tracker (but some algorithms for *dropping* and *adding* cases) and there is no further analysis of the obtained data.

Finally, Ramanathan *et al.* [30] published a method to recognize event and key actors in multi-person videos by detecting the focus of attention of different basketball plays. The goal of this research was to amend the lack of a universal method to emphasize attention or include key actors in sport sequences. In order to carry out this project, they manually labelled sets of plays of Youtube basketball games using Amazon mechanical Turk. Then, for every class, they extracted features including both scene and particular player information; right after, a deep learning framework is used to classify. To properly track the players, the Lucas-Kanade tracker [3] is implemented in combination with a bipartite graph, which is used for matching. Their event detection method is done through a sliding window technique that displays attention with a heat-map. Results outperform some state of the art methods, and their dataset can be found on-line. However, the number of classes is simplified to few similar plays (*i.e* 2-points shot success/failure, 3-point shot success/failure), and their tracking system is based on positions in the screen, and not real coordinates in the basketball court, so 3D vision has to be applied in order to estimate the real position.

2.4 Overview

Although all summarized articles had different goals, a common characteristic can be found: all research is based on tracking data, which is an emerging type of information being gathered in many sports games. The general approach of all contributions is to design algorithms based on tracking data to have a high-level understanding of the game and to define/extract different types of advanced statistics. This approach is the basis of the presented project: by recognizing patterns on tracking data, plays can be classified, thus providing new metrics that cannot be found in nowadays box scores. Moreover, several conclusions can be extracted:

1. Multi-tracking basketball algorithms are difficult to design, as occlusions can only be avoided with complex multi-camera configurations. It can be said that basketball courts are challenging scenarios.

2. Tracking data is a powerful tool to define new metrics. Moreover, metrics can be generated for any single aspect of the game.

3. There is a real need of defining new metrics, as the information in box scores is not enough to explain the details of the game. As shown, the best shooter of the league might not be the player with higher field goal percentage, or the best blocker may not be the block league leader.

4. The vast majority of analysed articles are quite new (from 2012), so the tendency of quantifying game aspects is boosting.

5. Some articles related to the classification of plays have been published; however, all these contributions used deep learning models and did not use a manual feature extraction process based on meaningful basketball information.

Chapter 3

Final Problem Formulation

In this Chapter, the definition of the ideal system is described, together with the limitations that reduced the scope of the project.
From an *academic computer vision* point of view, the most interesting system would involve the design of an accurate multi-tracking algorithm in order to have players' and balls' position at a high frame rate (25 fps at least); this position should then be converted into court coordinates using 3D vision, thus making the system robust to panning or zooming. Besides, in basketball sequences, many cameras should be used in order to ensure that all targets are being detected, as multiple occlusions are present in sport sequences; camera synchronization is then a concern as well. This system should track all players (distinguishing teams), and each player should have an individual unique ID, which would be used for extracting statistics (numerical and visual displays). Likewise, obtained data after tracking should be analysed to accurately recognize patterns occurring inside the court, such as basic (screens) or wide (plays) concepts, by training a *machine learning* model; the set of plays to be detected should contain between 10 and 15 classic basketball actions (and subclasses of those). Potentially, if a lot of data is available, deep learning techniques could be implemented. This system should be able to detect actions and extract advanced statistics in real-time. With this type of tool, coaches would have the chance of adapting their strategies during the game. All these features are visually displayed in the block diagram shown in Figure 3.1:

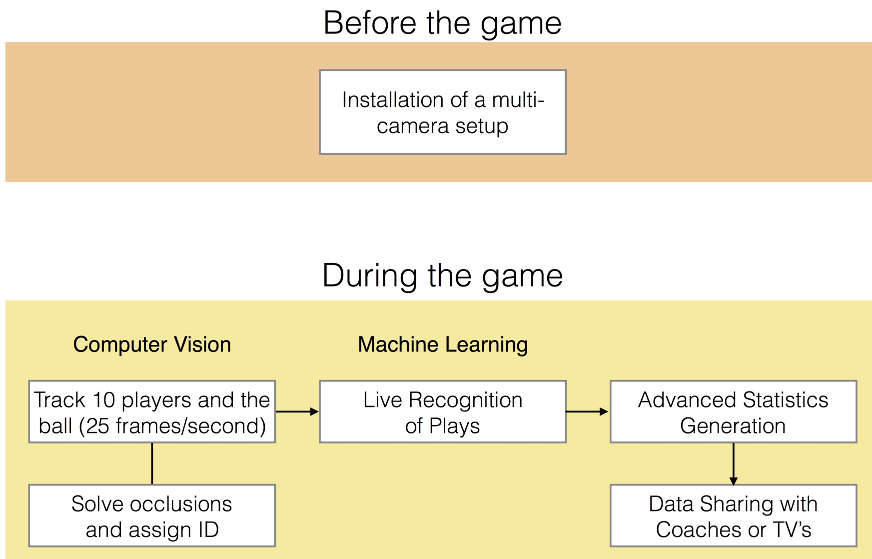

Figure 3.1: Block diagram with the required steps of an ideal system.

3.1 Delimitation

As this project is a Master Thesis of 6 months of duration, its scope had to be delimited, as the ideal system does not even exist in United States (they do not provide data in real-time). In Section 1.2, it has been mentioned that the design of a multi-tracking algorithm was left behind for the purposes of this project (using positioning sensors instead), thus focusing on the Machine Learning part. The content of this module was narrowed too: instead of detecting actions in real-time, the presented system should work in a semi-automatic way: knowing that an action happened during a period of time, the algorithm must classify this action. Moreover, as it might be too ambitious to start the project with the goal of classifying actions among more than 10 different classes, the initial idea is to successfully distinguish between a small set of 5 plays. Note that, given tracking data, the extraction of advanced is implicit in this scope. From the block diagram of an ideal system (shown in Figure 3.1), a delimited diagram is shown in Figure 3.2, where dark boxes correspond to the project's focus.

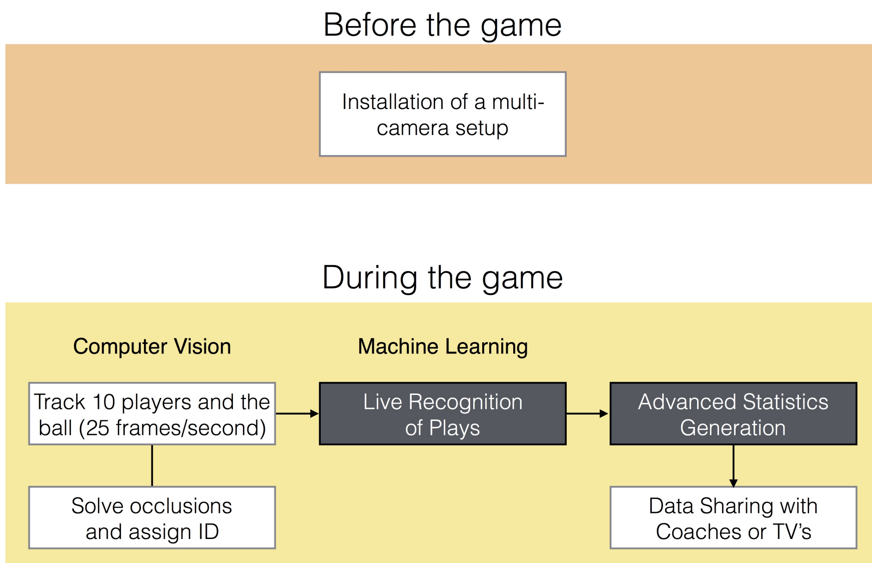

Figure 3.2: Block diagram; dark boxes correspond to the scope of this project.

3.2 Success Criteria

Before explaining the core of the project and presenting results, prior expectations have to be detailed, which will help to evaluate *a posteriori* if the goals of the project have been accomplished:

1. Gathering an acceptable experimental dataset to work with. This dataset should include at least 10 observations of 4 different plays and the tracking of 5 players minimum.

2. Finding a *fast* way to manually track the ball. For a basketball sequence of T duration, tracking the ball in all frames should not take more than $1.5 \times T$.

3. Managing to parse all data from sensors to obtain a 2D representation of data without occlusions; by visualizing this kind of representation, a basketball coach must be able to identify what is going on on court.

4. Extracting at least 6 meaningful types of advanced statistics and create different visual displays.

5. Designing a (semi)automatic method to pick only the involved players of an action.

6. After a feature extraction process, training a Machine Learning model that could classify plays with *notable accuracy*. Although it is difficult to define numerically what *notable accuracy* means, the model is expected to classify the whole dataset better than a non-basketball-expert.

Nonetheless, this success criteria is thought from a totally academic point-of-view. Having a working prototype, other objectives should be defined in order to measure how successful the program is for real sport situations, such as (a) increasing the number of wins or (b) selling more tickets due to a new playing style.

Chapter 4

Design

As mentioned, the goal of this project is to classify basketball plays within a closed-set of actions. In order to accomplish this objective, some steps (shown in the block diagram in Figure 4.1) are required:

1. Having the appropriate data to work with, an analysis of tracking is performed. This process includes the manual tagging of ball events, the parsing and synchronization of all tracking signals and the generation of a 2D representation of the game.

2. Detecting those 3 players with a relevant role in the action being played.

3. Extracting a feature vector (containing meaningful basketball features) for each action.

4. Training a model with all feature vectors.

In this chapter, the proposed system is presented, containing the following parts: first, the experimental gathered dataset is detailed in Section 4.1, and a brief discussion on pros and cons about accelerometric sensors can also be found. Afterwards, in Section 4.2, a manual approach to tag ball events using hotkeys is presented, which could be used to track the ball almost in real-time. Then, the parsing process used to organize all gathered information is described in Section 4.3, and right after, the extraction of some advanced statistics (such as numerical data or heat-maps) is introduced in Section 4.4. Afterwards, new approaches to select three involved players in each action are detailed in Section 4.5. Finally, in Section 4.6 the feature extraction process is explained.

Implementation

Besides, the technical specifications of the presented work are the following ones:

- The used computer to perform all tests is a MacBook Pro with a 2.4 GHz Intel Core i5 processor and 8 GB of RAM memory (16000 MHz).

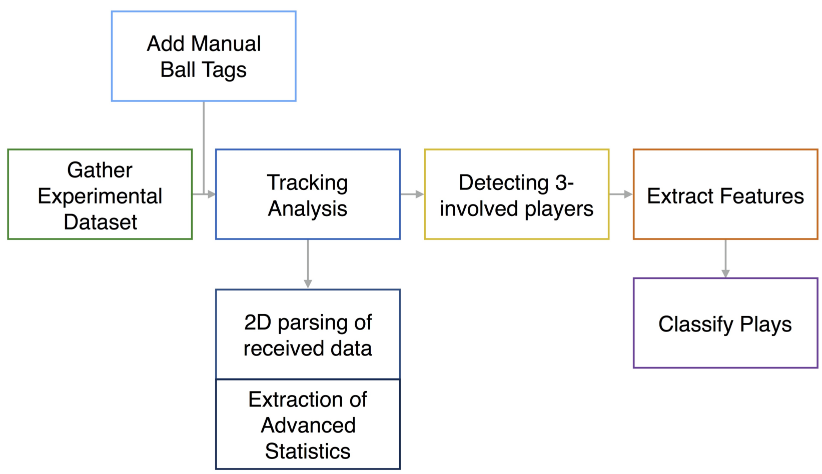

Figure 4.1: Basic flow to classify plays from sensor data.

- All code was written in Matlab.

4.1 Experimental Dataset

As seen in Section 1.2, the limitations of the on-line existing annotated datasets are evident (coordinates relative to the screen, short video sequences…), so a new dataset was created from scratch. As mentioned, the idea is to build a low-cost system, so positioning sensors (borrowed from the company NBN23) were used instead of a multi-camera configuration setup. The dataset contains 30 minutes of a whole practice of the Under-21 Team of *Valencia Basket Club* (Spanish team, in Valencia), with 8 of the 10 players in the practice wearing sensors. More concretely, the most relevant content of this practice can be divided (by a basketball expert) into 96 observations or drills:

- 22 repetitions of a 3-on-0 (3 offensive players, 0 defensive) exercise to practice *floppy offense* motion. This is a half-court exercise.
- 22 repetitions of a 4-on-0 exercise starting with a pick and roll. This is a half-court exercise.
- 14 repetitions of a 3-on-3 press break exercise to overcome defensive pressure. This is a full-court exercise.
- 21 repetitions of a 3-on-2 post-up exercise. This is a half-court exercise.
- 17 repetitions of a 2-on-2 fast-break exercise. This is a full-court exercise.

NOTE 1: all these plays are visually explained in Section 1.3.5.
NOTE 2: press-break and post-up exercises were done at the same time but on the different ends of the court (splitting the group into exterior and interior players respectively).
Despite working with sensor data, the practice was also recorded with a single static camera (neither with panning nor zooming). The reason for doing so will be described in detail in Section 4.2. This dataset (video plus tracking data) cannot be found online, as it belongs to NBN23.
Players appearing in the recordings can be divided into exterior/interior given their ID:

- Exterior: *Players 1, 2, 5, 7, 8, 9.*
- Interior *Players 3, 4, 6, 10.*

4.1.1 Accelerometric Wereable Sensors

Accelerometric NBN23 sensors are not bigger than a coin, and emit amplitude Bluetooth signals at a frame rate of 25 fps, which are then captured by 3 receptors placed in the court at pre-established spatial locations. Real-time receptors send the captured information to a server, and a script creates an individual *.csv* file for each player containing all his/her corresponding data. By triangulating the signals (as seen in Figure 4.2), the emission can be decoded in order to obtain the following information: Timestamp, ID and X and Y position in the court (measured in meters).

On the one hand, the two main drawbacks of working with sensors are evident: (a) data can only be extracted in those teams that use sensors, so the option of scouting another team is *a priori* discarded, and (b) the ball also needs to have an integrated sensor; otherwise, it must be tracked somehow. On the other hand, sensors are an easy and cheap technology to be used in team practices and do not require *extra-employees* in court; when using cameras, people in charge of recording and monitoring *audio-visual* devices are needed.

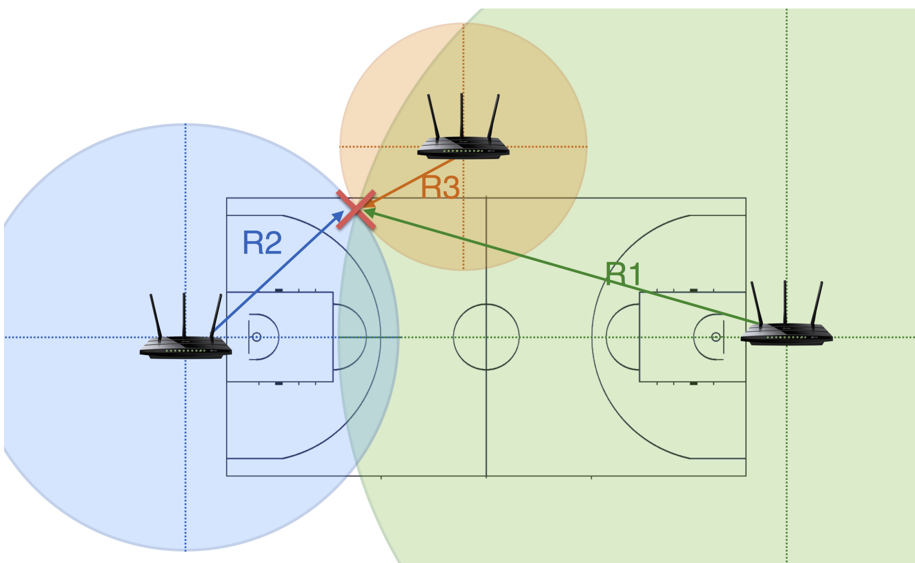

Figure 4.2: Signal triangulation through amplitude signals; with three receptors, the exact position of a player inside the court (red cross) is obtained.

4.1.2 Manual Tracking Annotations

Unfortunately, when gathering the experimental dataset, only 8 sensors were available, so the corresponding tracking of the remaining players (2 out of 10) had to be manually done. Although different video tracking methods could be applied, the conversion from screen coordinates to court coordinates was not trivial, so a simpler approach was used. Having converted the video into frames, the Bounding Box Annotation tool created by the *Visual Analysis of People* research group of Aalborg University was used [28]. In this program, two images are shown next to each other: at the left/right side, the actual frame, and at the right/left side, an empty picture of the court. A screenshot of this program is displayed in Figure 4.3. The goal is to estimate the player position in the court by drawing a bounding box as accurately as possible. The only difference when comparing different tracking trajectories is the noise these signals contain. The real ones, captured from the sensors, are much more precise, and it is really difficult to find two consecutive frames where a player stays completely static. Moreover, in the *post-up situation* exercise, one of the coaches was manually tracked as well, because he participated in the exercise passing the ball.

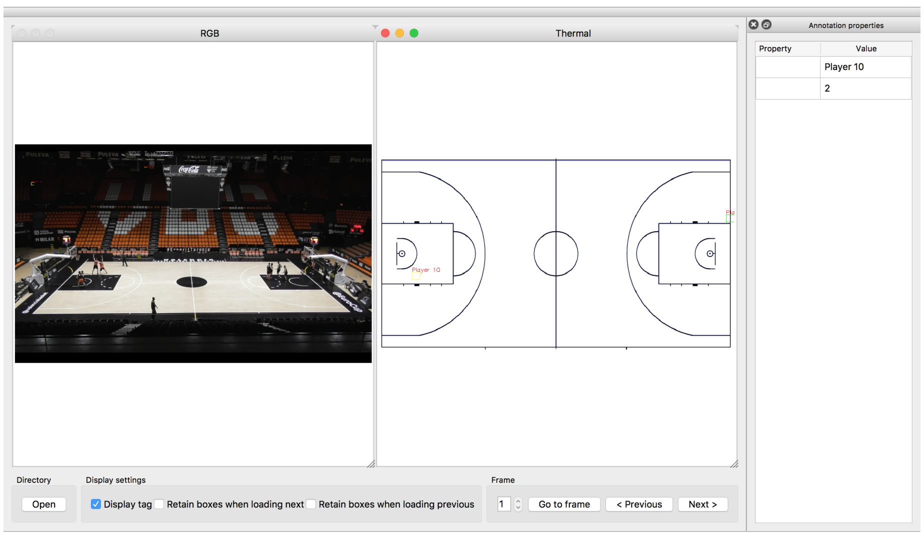

Figure 4.3: Bounding box annotation of a single frame, where the court position of *Player 9* and *Player 10* is estimated from a real frame.

4.2 Tracking the Ball using Manual Tags

In the presented experiment, it was not possible to integrate a chip to all the balls the Valencian club had, so an alternative had to be found. The solution was based on the following principle: *you can estimate the ball position even if you do not know the exact coordinates; you just need to know which player has it.* For this reason, video tracking techniques are discarded and a simpler procedure is chosen. Having recorded the game/practice, a program with hotkeys is designed to create another *.csv* file, containing annotations with the current frame and a tag indicating the type of action, which could be one of the following:

- A player gets the ball (receives from another player / grabs a rebound / steals it). This action is labelled with the following label: *team-number-IN* (*i.e.* *a9IN* would mean that the player 9 of the *a* team gets the ball).

- A player releases the ball (passes to another player / attempts a shot / looses it). This action is labelled with the following label: *team-number-OUT* (*i.e. b3OUT* would mean that the player 3 of the *b* team releases the ball).
- A player is substituted (starts playing / goes to the bench). This action is labelled with the following label: *X-team-number-IN/OUT* (*i.e. X.a1OUT* would mean that the player 1 of the *a* team goes to the bench).
- The ball touches the rim. This action is labelled with a simple tag: *BKT*.
- The game is paused/resumed. These actions are labelled with simple tags: *STOP / RESUME*.

Besides, from the lecture of these tags, statistical features can be extracted: *e.g* if a tag *N* says that *player A* releases the ball and the tag *N+1* indicates that the ball touches the rim, it is obvious that *player A* attempted a shot; otherwise, if the *N+1* tag indicates that another player receives the ball, that action was a pass. Moreover, advanced parameters such as the speed of pass can be estimated too.
An example of few lines of the resulting *.csv* file containing ball annotations could be the following one:

```
Cam0.png,Xa2IN
Cam0.png,Xa1IN
Cam0.png,Xa4IN
Cam2.png,a2IN
Cam7.png,a2OUT
Cam14.png,a1IN
Cam23.png,a1OUT
Cam30.png,a4IN
Cam38.png,a4OUT
Cam42.png,BKT
Cam48.png,Xa2OUT
Cam49.png,Xa8IN
Cam50.png,Xa1OUT
Cam51.png,Xa5IN
Cam52.png,Xa4OUT
Cam53.png,Xa6IN
Cam62.png,a8IN
Cam87.png,a8OUT
Cam94.png,a5IN
Cam102.png,a5OUT
```

Furthermore, by tagging the ball with these kinds of events, different repetitions of the exercise are automatically separated as well. The limits to separate plays are the *BKT* or *STOP* tags; this technique proved to be useful when extracting features of each play, as it will be seen in Sections 4.6. After analysing

the resulting *.csv* file, two vectors corresponding to the frames indicating the beginning and the end of the action are obtained, such as:
Start = [4.7 15.8 24.6 32.5 41.6 50.3 57.8 66.0 73.2 82.3 93.4 99.7 107.1 115.0 124.4 133.2 141.6 152.8 166.1 173.3 180.5 199.6 206.6 215.5];
End = [12.0 21.3 30.0 38.8 47.4 54.8 64.4 70.8 79.6 89.7 97.0 104.9 111.4 121.2 130.6 139.0 147.7 156.0 170.2 177.9 186.3 205.5 213.9 221.8];
Having these couple of vectors, actions can be split with ease: for instance, it is obvious that the third play starts at second 24.6 and ends at second 30.0.

4.2.1 ERIC Sports Button-Pad

The first attempt of manually tagging the ball was to use video-analysis software, like ERIC Sports [1], produced in the company where I did my internship in the 9th semester. In this kind of programs, you can create your own button-pad with as many tags as desired. However, as it can be seen in Figure 4.4, this approach is not ideal from the *user-experience point-of-view*, as the user must click on different spots of the window, thus wasting a lot of time moving the mouse pointer up and down and pausing the video many times. It could be a good solution if there were less people playing, but with *i.e.* 10 players and 4 possible tags for each one, the user has to look for one concrete button in a pad containing 40 options.

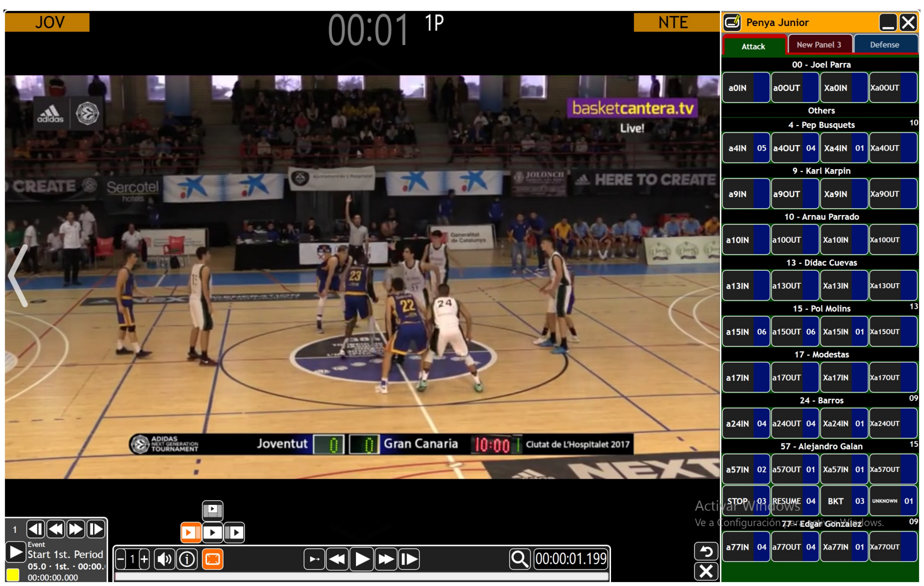

Figure 4.4: Designed ERIC Sports button-pad, used to tag all ball events.

	Receive / Release the Ball (key)	Starts playing / is Substituted (key)
Starting Point Guard	*1*	q
Starting Shooting Guard	*2*	w
Starting Small Forward	*3*	e
Starting Power Forward	*4*	r
Starting Center	*5*	t
Back-up Point Guard	*6*	y
Back-up Shooting Guard	*7*	u
Back-up Small Forward	*8*	i
Back-up Power Forward	*9*	o
Back-up Center	*0*	p

Table 4.1: Key-player associations.

4.2.2 Customized Hot-Keys Program

An alternative to button-pads was to create a homemade program that could be used without having to move the pointer. The basis of the thought program is to associate each player with a couple of close keys in the keyboard, which are shortcuts to indicate if a player receives/releases the ball (binary state), or if he/she starts playing or is substituted. In this case, the row of numbers (1,2,3,...,0) and the first row of letters (q,w,e,...p) are used. The *player-key* associations (shown in Table 4.1) were done with the following logic: basketball positions are usually expressed with numbers from 1 (point guard, small player) to 5 (center, big player), so the *1-q* keys will correspond to the actions of the starting point guard, and *5-t* to the starting center. For bench players, the same logic is applied but starting with number 6 (backup point-guard).
The final hotkey configuration used in this experiment can be seen in Figure 4.5. In terms of speed, annotations can almost be generated in real-time; different tests were performed and it was estimated that the time to tag a video of duration T is $1.15 \times T$.

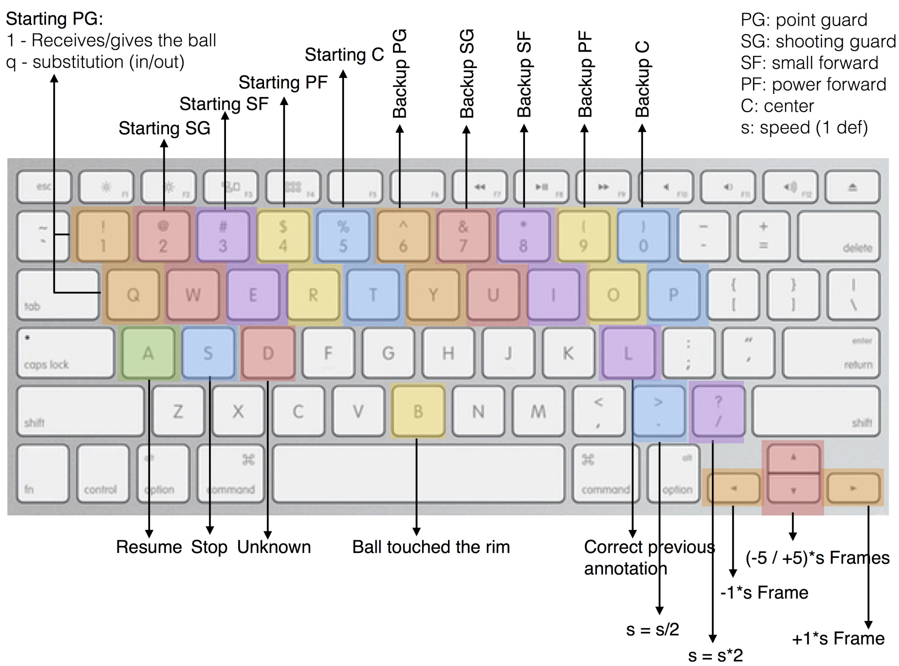

Figure 4.5: Final hotkey configuration for labelling ball events. Note that PG, SG, SF, PF and C are the different basketball positions.

4.3 Parsing into a 2D Representation

Once the information of both ball and players was obtained in different files, data had to be synchronized and merged together in a single matrix sorting by timestamp values. It has to be taken into account that sensors start emitting when the player activates them, so there is not a universal beginning for all signals. Inside this matrix, all the samples that can be comprised into time intervals of 40 milliseconds will correspond to the same frame.

Having the tracking data of the whole sequence, an animated 2D pictorial representation can be generated over a court image in order to have a visual support of the practice/game without occlusions; as it can be observed in Figure 4.6, every square represents a player and the sensor ID determines its colour. The only thing to be considered is that the decodification of the sensor signal takes as a reference the centre of the court and it is horizontally flipped with respect to the camera point of view, so a conversion has to be applied in order to obtain the player position in pixels. Having the following variables:

$$\text{im}_{\text{size}} = (\text{im}_{\text{width}}, \text{im}_{\text{height}}) \quad [\,pixels\,]$$

$$\text{court}_{\text{size}} = (28, 15) \quad [\,\text{m e t e r s}\,]$$

$$\text{im}_{\text{center}} = (\frac{\text{im}_{\text{width}}}{2}, \frac{\text{im}_{\text{height}}}{2}) = (h_x, h_y)$$

$$f_W = \frac{\text{im}_{\text{width}}}{28}; f_H = \frac{\text{im}_{\text{height}}}{15}$$

given a point (x, y) in meters representing the player location inside the real court, the mapped point (X, Y) in the image expressed in pixels is:

$$(X, Y) = \begin{cases} (h_x - (x * f_W), h_y + (x * f_H)) & \text{if } (x \leq 0) \& (y \leq 0) \\ (h_x - (x * f_W), h_y + (x * f_H)) & \text{if } (x \leq 0) \& (y > 0) \\ (h_x + (x * f_W), h_y + (x * f_H)) & \text{if } (x > 0) \& (y \leq 0) \\ (h_x + (x * f_W), h_y - (x * f_H)) & \text{if } (x > 0) \& (y > 0) \end{cases}$$

In these 2D frames, there is the possibility of better understanding the game by:

- Drawing lines to represent the trace of the players over the last *N frames* or even the whole sequence.
- Animating the ball using a basic linear motion model based on its tags. Knowing the exact frame where a player releases/receives it, the ball trajectory is easy to map.
- Changing the width of the squares of those players on court/doing an exercise (in order to distinguish which are the ones playing and the ones in the bench).
- Drawing a cross in those positions where a shot has been attempted; note that this cross can stay in the remaining frames or disappear after a certain number of seconds.

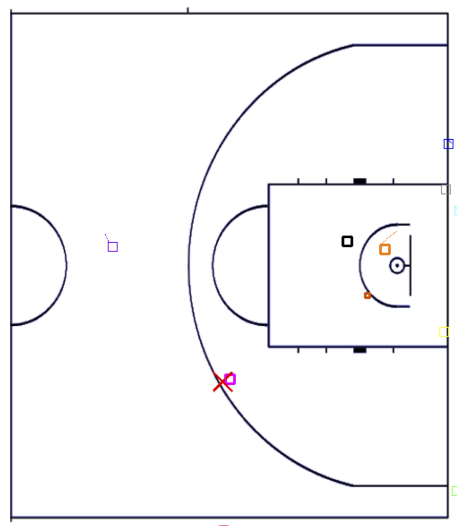

Figure 4.6: 2D Representation of a *floppy offense* situation, executed by three players (pink, black and orange). As it can be seen, the player that corresponds to the pink square has attempted a long-range shot and the ball (small orange square) is about the touch the rim.

It has to be mentioned that, in order to obtain the 2D representation, some drawing functions were created to speed the process up. It turned out that calling the *build-in* Matlab functions to draw circles, rectangles or lines took a lot of time; actually, it took around 20 seconds to process one frame. In order to change this slow approach, new functions were created with the purpose of directly accessing and changing the concrete RGB pixel values. By using the following functions, the processing time of a frame was reduced to 0.14 seconds (140 times faster):

- *drawLine(initial position (x,y), final position (x,y), color, width)*
- *drawRectangle(top left corner (x,y), bottom right corner (x,y), color, width)*

4.4 Extracting Advanced Statistics from Tags

As mentioned in Section 1, statistics are crucial for coaches in order to prepare practices and games, but parameters that can be found on-line just show tangible parts of the game, such as the number of points a player scored. By reading ball annotations, several advanced statistics can be obtained with ease, such as:

1. The number of passes a player receives.
2. The number of passes a player gives.
3. The number of times a player enters the court.
4. The number of times a player is substituted.
5. The mean and maximum amount of seconds a player keeps the ball while playing.
6. The number of attempted shots from mid/long-range distances.
7. The number of attempted layups (short-range shots) from the left/right side of the court.
8. The number of open and contested shots the player attempts, which indicates if the player usually shoots alone (open) or in a tough situation in front of a defender (contested). The worldwide established threshold to check if a shot is contested or not is 1.8288 meters with respect to the closest defensive player (equivalent to 6 feet).
9. The number of seconds a player stays standing-still.
10. The total displacement of a player.
11. The speed of the player.
12. The ball speed in given passes.

Moreover, some other interesting metrics can be obtained by combining different advanced statistics. An example could be analysing if a player tends to execute *catch-and-shoot* situations; in this type of plays, the goal of the team is to force a player to receive the ball all alone and take an open shot in less than a second. Dividing the number of field goals of mid/long-range shots by the number of seconds a player holds the ball would provide a meaningful metric.
Besides, it is also possible to filter these statistics by a time period in order to have a better understanding of the game. For instance, coaches might want to compare the statistics of players between the first and the last 5 minutes of a game, which could differ because of many factors, such as pressure or tiredness. An example of a box-score containing some of the statistics (extracted from 8 *Floppy Offense* situations) is shown in Table 4.2.

	A	B	C	D	E	F	G	H	I	J	K
Starting PG	2	3	2.20	2.80	1	0	0	0	1	0	1.40
Starting SG	3	4	1.85	1.90	0	1	0	0	1	0	1.12
Starting SF	1	1	0.80	0.80	0	0	0	0	0	0	0.80
Starting PF	1	3	0.46	0.60	1	0	1	0	1	1	1
Starting C	2	3	2.10	2.90	0	0	1	0	0	1	1.20
Back-up PG	2	3	0.43	0.60	1	0	0	0	1	0	0.90
Back-up SG	3	3	5.43	10.30	0	0	0	0	0	0	1.12
Back-up SF	3	3	2.20	3.10	0	0	0	0	0	0	1.26
Back-up PF	3	3	1.96	2.10	0	0	0	0	0	0	1.12
Back-up C	0	1	0.60	0.60	0	0	1	0	0	1	0

Table 4.2: Box-score with several advanced statistics: (from left to right) (A) number of given passes, (B) number of received passes, (C) mean ball possession (in seconds), (D) maximum ball possession (in seconds), (E) mid-range attempted shots, (F) long-range attempted shots, (G) left-side attempted layups, (H) right-side attempted layups, (I) open shots, (J) contested shots, (K) mean pass speed (in meters/second).

4.4.1 Advanced Visual Statistics

Apart from the above-mentioned advanced statistics, coaches are also interested in analysing data visually, as it is a much faster way to extract conclusions and it does not require numerical analysis. The generation of two types of maps is explained in this section: heat-maps and shot charts.

Heat-Maps

A widely used graphical way to represent the regions of the court where players are more influential are heat-maps. This display is based on a simple colour-codification applied to tracking data, which represents the frequencies of players in each part of the court and might lead to detect tendencies or tiredness. In the case of sport sequences, the general procedure to build a heat-map having with the tracking of a player is the following one:

1. Building an empty matrix with the same court size (in pixels).
2. Checking the whole tracking trajectory, adding up observations in those concrete positions where the player stepped in.
3. Converting observations into probabilities by normalizing the matrix.
4. Windowing the matrix with a box in order to turn single spots into small pixel neighbourhoods. This step also helps smoothing the representation. Two possible boxes are shown in Tables 4.4.1 and 4.4.1, where values are weighted in a Gaussian way in 3x3 and 5x5 boxes respectively.

5. Applying a colour codification and overlapping the heat-map over the real court.

1/16	2/16	1/16
2/16	4/16	2/16
1/16	2/16	1/16

Table 4.3: 3x3 Gaussian Box

1/256	4/256	6/256	4/256	1/256
4/256	16/256	24/256	16/256	4/256
6/256	24/256	36/256	24/256	6/256
4/256	16/256	24/256	16/256	4/256
1/256	4/256	6/256	4/256	1/256

Table 4.4: 5x5 Gaussian Box

An example of different types of heat-maps is shown in Figure 4.7, where not only the separate trajectories of three players are displayed individually, but also a single heat-map of all of them.

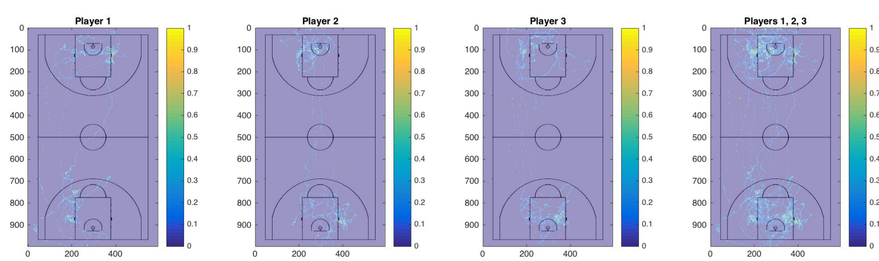

Figure 4.7: Individual and collective heat maps obtained with a 3x3 Gaussian Box; the color codification indicates the frequency of a player standing in a particular region of the court over a sequence of time.

Shot Charts

Another interesting figure that can be found in some leagues box scores is called *shot chart* and represents the position of attempted shots. Nowadays, this graphic is manually done by estimating at first glance the position where the player attempted the shot and clicking on a pixel in the screen, but it can be automatically generated by reading ball events (Section 4.2). For example, if there are two consecutive annotations that are (*Cam38.png, a4OUT*) and (*Cam42.png, BKT*), it is obvious that *Player 4* of the team *A* attempted a shot at a certain frame (38); obtaining the position of the involved player in that frame would provide the shooting position, and a cross can be drawn in that same spot over the court. All the shots attempted while doing the *Floppy Offense* exercise are shown in Figure 4.8.

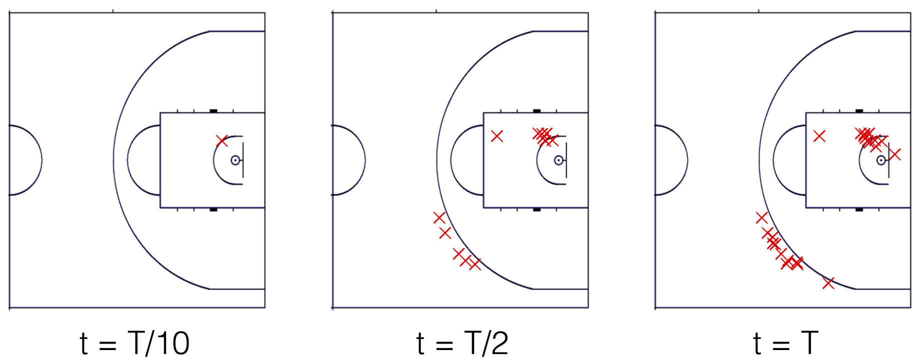

Figure 4.8: Shot chart with all attempted shots during the *floppy offense* exercise; as it can be seen, the image is progressively updated every time there is a shot.

Besides, other approaches can be thought to include more interesting data in these displays. For example, adding temporal information next to the cross, such as the exact moment where a shot was attempted. An example of this temporal inclusion is shown in Figure 4.9. However, in order to be totally understandable, the graphic should be interactive, and show this type of information when moving the pointer over the cross; otherwise, when there are a lot of shots, texts would overlap each other.

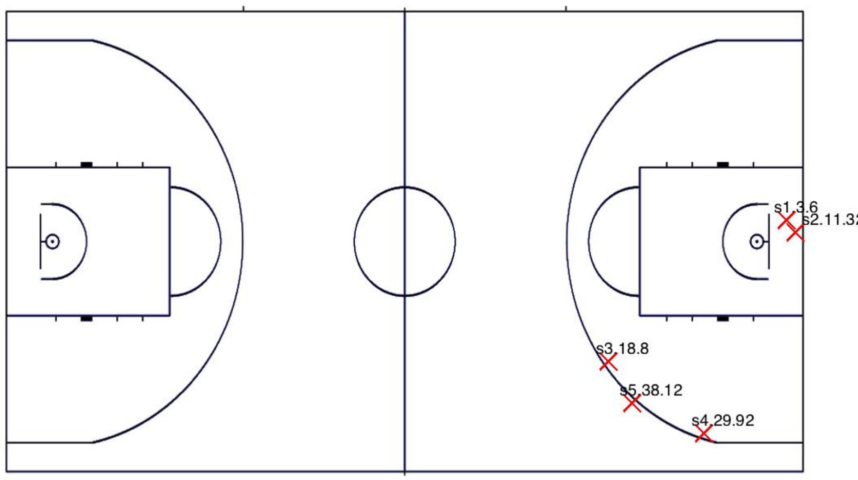

Figure 4.9: Shot chart with five attempted shots during the *floppy offense* exercise; as it can be seen, the timestamp is annotated next to every shot.

Moreover, there is some relevant information missing as well: the failure/-success of the attempted shot. As coaches might want to know the region of the court where his/her players are more effective, this metadata should be included. A simple way of entering this information is applying a colour codification when drawing the cross: painting it green if the player scored or red otherwise. Furthermore, the approach to be followed when trying to guess if a shot was successful or not is to track what happens right after:

1. If a player of the same team captures the rebound and does not go outside the court, an offensive rebound is faced, so the attempted shot was not successful.

2. If a player of the other team grabs the ball, and then drives/passes it whilst moving to the other end of the court, a defensive rebound is faced, so the attempted shot was not successful.

3. If a player of the other team grabs the ball, then goes out-of-bounds, and then he/she passes it, the attempted show was successful.

Note that this colour codification could have been done if the dataset included game data; however, having only repetitions of different drills, the mentioned patterns are not being followed, as the goal of the exercise is to practice a certain movement instead of playing a real match situation.

4.5 Three-Player Selection

Although there are thousands of different basketball plays, there is a common pattern in some of them: even if the 5 players on court move during the action, only the movement of 3 of them is relevant for its outcome. For this reason, while extracting features of different plays, only data from 3 players is actually processed. It can be argued that typical plays like the *UCLA cut* or *Flex* are evident exceptions, but at the same time, there have been prestigious coaches (*e.g* Phil Jackson) who based their strategies in the *Triangle Offense*, a tactic first introduced by Sam Barry based on *three-player* concepts [39].
As explained in Section 4.2, the players on court could be known by reading event tags, and these have to be filtered in case there are more than 3 doing an exercise/playing the game. A simple assignation is done by matching the ID's of the involved players to the following variables: *Player1, Player2* and *Player3*. Besides, another key reason to only select three players was the limitation of the dataset: in training sessions like the recorded one, concepts are practiced in small groups. In the experimental dataset (explained in Section 4.1), the total number of players doing the exercise is the following one:

1. *Floppy Offense*: 3 players; as they repeat the exercise several times without defense, the selection is straightforward.

2. Pick and roll: 4 players. From these 4 players, 1 has to be discarded.

3. Press break: 3 players. In this exercise, there are 6 players divided in two teams (3-3), so all three defenders are discarded.

4. Post-up situation: 2 players + 1 coach. As explained in Section 4.1.2, in this exercise there are not only 4 players doing the exercise (2 on offense and 2 on defense) but also a coach who passes the ball when necessary.

5. Fast-break: 2 running players + 1 *static* player. In this exercise, the actual fast-break is executed by just two players, but another one is also involved passing the ball from one side of the court.

Having explained these limitations, it is obvious that a 5-player gold standard cannot be tested, but a first prototype based on 3 players can be built. Furthermore, players also have to be sorted by *positions*, which can be estimated with the players' coordinates at the beginning of the play; otherwise, patterns cannot be found in data, *e.g* small-fast players' features must be compared to other small-fast players and not to heavy-slow ones. This association is explained in the following list, where the left and the right side of the court are always based on offensive team direction:

1. *Floppy Offense*: *Player1* - Point Guard who starts the play driving the ball outside the 3-point line, *Player2* - Shooting Guard who initiates the action placed at the right side of the baseline, *Player3* - Center who initiates the action placed at the left side of the baseline.

2. Pick and roll: *Player1* - Point Guard who starts the play driving the ball outside the 3-point line in a central position, *Player2* - Shooting Guard who initiates the action placed below the rim, *Player3* - Center who initiates the action placed at the left side of the court. The center that initiates the action placed at the right side of the court is the one left behind.

3. Press break: *Player1* - Point Guard who starts an out-of-bounds situation behind the baseline, *Player2* - Shooting Guard / Small Forward who initiates the action placed at the right side of the court, *Player3* - Small Forward / Shooting Guard who initiates the action placed at the left side of the court.

4. Post-up situation: *Player1* - Coach, always placed outside the 3-point line (either at the right or left side of the court), *Player2* - Power Forward / Center who initiates the action placed at the left side of the baseline, *Player3* - Center / Power Forward who initiates the action placed at the right side of the baseline,

5. Fast-break: *Player1* - player placed at the left side of the defensive court outside the 3-point line, *Player2* - player placed at the right side of the defensive court outside the 3-point line, *Player3* - passing player placed at the right side of the offensive court.

Several methods were tested in order to find the best one regarding the *automation-generalization-accuracy* trade-off: using a Manual Input, or finding out patterns or maximum correlation values between trajectories.

4.5.1 Manual Selection: Graphical Input

Introducing a graphical input is a simple-manual task: before processing the sequence and starting the feature extraction process, the first frame is shown, where all the squares representing players are visible. Then, the user clicks three times in the appropriate order as close as possible to those players; having stored in variables the coordinates of the click, pairwise distances are computed. The association is performed by finding the closest player to each click.

Although the *Graphical Input* approach may provide high confidence to the user, it is a totally manual process; besides, it might be misleading as well, as the first frame can be confusing (a player that reacts slow and starts far behind from the *a priori* initial position). Therefore, automatic (or semi-automatic) methods had to be found.

4.5.2 Automatic Selection: Speed

The first attempt to select the involved players automatically is based on comparing their speed during the action; normally, those 3 players that move faster during the action are the ones doing the exercise, especially in practices (instead of those standing still waiting for their turn). This method is completely automatic and does not depend on the kind of exercise; its performance will be described in Section 5.1.1.

4.5.3 Automatic Selection: Correlation

Another interesting method to detect the three-involved players is to compare trajectory signals in order to find the most similar ones in terms of correlation. Two methods were tested, based on semi-automatic (Pattern Correlation) and fully automatic (Blind Correlation) approaches. The general procedure of both methods is to compute the Pearson correlation coefficients between signals, as can be seen in Equation 4.1, where *P1* and *P2* correspond to trajectories of two different players.

$$\rho(P_1, P_2) = \frac{1}{N-1} \sum_{n=1}^{N} (\frac{P1_i - \mu_{P_1}}{\sigma_{P_1}})(\frac{P2_i - \mu_{P_2}}{\sigma_{P_2}}) \tag{4.1}$$

Note that Pearson coefficients go from 0 to 1, where values close to 1 correspond to highly correlated signals. Moreover, it has to be explained that, for those exercises where there is both an offensive and defensive team, a basic pre-filtering is done, discarding the defensive players from the very beginning. The reason for doing this is simply that defensive players move together with the players they are guarding, resulting in almost identical trajectories, as the goal is not to leave those alone. If defensive players would not be filtered, correlations between each pair of players (offensive-defensive) would potentially be higher than any other.

Furthermore, an additional step is required before computing Pearson coefficients between two trajectories: the size of both signals has to be the same, so basic resizing with linear interpolation is applied. An example of this kind of resize if shown below:

```
A = [0 2 4 6 8 10]; (size 1x5)
B = [11 27 81 23 46 213 65 119 1023 2]; (size 1x10)
```

As B is larger than A:

```
A' = [0 1 2 3 4 5 6 7 8 9 10]; (size 1x10)
B = [11 27 81 23 46 213 65 119 1023 2]; (size 1x10)
```

Pattern Correlation

In this semi-automatic method, some prior information is given: the patterns containing the type of movements that have to be found. For instance, given the trajectories of a pick and roll sequence, if the type of movement the *Point Guard* is going to do is known, computing correlation coefficients between the existing template and the rest of players would potentially indicate which one has the most similar movement. The whole procedure is explained in the following snippets of pseudocode and Tables 4.5 and 4.6:

Being N the number of stored templates (3), Np the total number of players (10), $P_1, ..., P_{10}$ the trajectories of all players and $Corr$ a matrix of N rows and Np columns:

```
for i= 1 to N {
    for j = 1 to Np {
        Corr(i,j) = ρ(P_i, P_j)
    }
}
```

	P1	**P2**	**P3**	**P4**	**P5**	**P6**	**P7**	**P8**	**P9**	**P10**
PG	0.9389	0.9783	0.9628	0.9712	0.9272	0.9712	0.9737	0.9482	0.8892	0.9712
SG	0.9642	0.9771	0.9588	0.9487	0.9709	0.9771	0.9636	0.9784	0.7740	0.9771
C	0.9718	0.9976	0.9880	0.9856	0.9698	0.9976	0.9963	0.9793	0.8764	0.9976

Table 4.5: Example of selecting 3-involved players using a *pattern correlation* approach (a). This matrix indicates the relation between the stored templates (movement of a Point Guard, Shooting Guard and Center) and the players on court, expressed as Pearson correlation coefficients.

Then, in order to know which the involved players are:

```
for i= 1 to N {
    Player_i = find(max(Corr(i,:)))
}
```

	P1	**P2**	**P3**	**P4**	**P5**	**P6**	**P7**	**P8**	**P9**	**P10**
PG	0.9389	**0.9783**	0.9628	0.9712	0.9272	0.9712	0.9737	0.9482	0.8892	0.9712
SG	0.9642	0.9771	0.9588	0.9487	0.9709	0.9771	0.9636	**0.9784**	0.7740	0.9771
C	0.9718	0.9976	0.9880	0.9856	0.9698	0.9976	0.9963	0.9793	0.8764	**0.9976**

Table 4.6: Example of selecting 3-involved players using a *pattern correlation* approach (b). The maximum Pearson correlation coefficients in each row correspond to the involved players; in this example, Point Guard corresponds to *Player 2*, Shooting Guard to *Player 8* and Center to *Player 10*.

For this specific case, the patterns of all 3-involved players are stored for the first play of each exercise, and then compared with the rest of examples. As mentioned, this method is semi-automatic and depends on the exercise, as one set of templates has to be generated for each action.

Blind Correlation

The last tested method is based on finding similar movement patterns between all players during a sequence of time. In the vast majority of actions, players have to move together in a coordinate way, so computing pairwise correlation factors between all players would provide high values in those synchronized ones. However, some filtering has to be done: if there are two players who are not taking part in an exercise and they are standing still at any place of the court, their correlation will be higher than the other ones, as their movement is null but identical. To solve this issue, the standard deviation of the players' trajectory over the time is computed and then thresholded, filtering those candidates that are definitely not moving. The whole procedure is explained in the following snippets of pseudocode and examples:

Being Np the total number of players (10), $P_1, ..., P_{10}$ the trajectories of all players and *Corr* a matrix of Np rows and Np columns:

```
for i= 1 to Np {
    for j = 1 to Np {
        if (i != j)
        {
            Corr(i,j) = ρ(P_i, P_j)
        }
        else
        {
            Corr(i,j) = 0;
        }
    }
}
```

For instance, Table 4.7 shows all pairwise Pearson coefficients (computed with Equation 4.1) over a random *floppy offense* sequence (obtained *Corr* matrix). In this example, as it can be seen, *Player 2*, *Player 5* and *Player 8* are discarded due to small standard deviation.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	1	0	0.362	0.292	0	0.346	0.888	0	0.829	0.297
2	0	0	0	0	0	0	0	0	0	0
3	0.362	0	1	0.1484	0	0.172	0.438	0	0.408	0.149
4	0.292	0	0.1484	1	0	0.144	0.362	0	0.332	0.1329
5	0	0	0	0	0	0	0	0	0	0
6	0.346	0	0.172	0.144	0	1	0.422	0	0.390	0.145
7	0.888	0	0.438	0.362	0	0.422	1	0	0.91	0.367
8	0	0	0	0	0	0	0	0	0	0
9	0.829	0	0.408	0.332	0	0.390	0.91	0	1	0.337
10	0.297	0	0.149	0.1329	0	0.145	0.367	0	0.337	1

Table 4.7: Example of detecting 3-involved players using the blind correlation method (a). In this Table, the Pearson correlation coefficients between all pairwise trajectories are computed.

Having diagonalized the matrix, the coefficients of each row and column are added together in order to see in which one similarities are higher.

```
for i= 1 to Np {
      Tot_Rows(1,i) = sum(Corr(:,i));
      Tot_Columns(i,1) = sum(Corr(i,:));
}
if (max(Tot_Rows)) > (max(Tot_Columns)) {
      Involved_Row = find(max(Tot_Rows));}
else {
      Involved_Column = find(max(Tot_Column));
}
```

This step can be seen in Table 4.8.

	1	**2**	**3**	**4**	***5***	**6**	**7**	***8***	**9**	**10**	**sum(Rows)**
1	**0**	**0**	**0.362**	**0.292**	***0***	**0.346**	**0.888**	***0***	**0.829**	**0.297**	**3.014**
2	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*
3	0	0	0	0.148	*0*	0.172	0.438	*0*	0.408	0.149	1.315
4	0	0	0	0	*0*	0.144	0.362	*0*	0.332	0.133	0.971
5	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*
6	0	0	0	0	*0*	0	0.422	*0*	0.390	0.145	0.957
7	0	0	0	0	*0*	0	0	*0*	0.91	0.367	1.277
8	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*
9	0	0	0	0	*0*	0	0	*0*	0	0.337	0.336
10	0	0	0	0	*0*	0	0	*0*	0	0	*0*
sum(Cols)	0	0	0.362	0.449	0	0.662	2.110	*0*	2.860	1.428	

Table 4.8: Example of detecting 3-involved players using the blind correlation method (b). In this Table, the sum of all Pearson correlation coefficients in each row and column is performed.

Then, taking only the *involved row/column* into account, it is clear that one of the involved players will be the one corresponding to that row/column. Besides, the two highest coefficients' positions will correspond to the other two players. This last operation can be seen in Table 4.9.

```
if (max(Tot_Rows)) > (max(Tot_Columns)) {
    Player_1 = Involved_Row
    for i= 1 to 2 {
        Player_i+1 = find(max(Corr(Involved_Row ,:)));
        Corr(find(max(Corr(Involved_Row ,:))),:) = 0;
    }
}
}
else {
    Player_1 = Involved_Column
     for i= 1 to 2 {
        Player_i+1 = find(max(Corr(:,Involved_Column )));
        Corr(find(max(Corr(:,Involved_Row ))),:) = 0;
    }
}
```

	1	2	3	4	*5*	6	**7**	*8*	**9**	10
1	0	0	0.362	0.292	*0*	0.346	**0.888**	*0*	**0.829**	0.297

Table 4.9: Example of detecting 3-involved players using the blind correlation method (c). In this Table, the Pearson correlation coefficients of *Player 1* and the remaining players are shown. As it can be seen, *Players 1, 7* and *9* are the ones selected in this example.

This approach is totally automatic and does not depend on the type of exercise.

4.6 Feature Extraction

As the experimental dataset was limited in the number of observations, deep learning models such as Convolutional Neural Networks [31] could not be applied. Therefore, features had to be manually extracted and carefully selected by an expert considering the game factors that allow distinguishing between two different plays. The basis of the feature extraction process is to have a single feature vector for each play containing both spatial and temporal information of the players. Note that this vector does not include any of the manually introduced ball information. Additionally, actions are divided into two segments in order to extract independent features from both the first and second half of

the play, which usually contain non-correlated information. An example could be a *pick and roll* sequence of duration T, where *Player1* calls the play at $t = 0$, receives the screen (see Section 1.3.5) at $t = \frac{T}{2}$ and then drives to the basket. *Player1* was almost static in the first segment of the action $[0, \frac{T}{2}]$ but moved fast in the second one $[\frac{T}{2}, T]$; dividing plays into segments can help detecting these kind of behaviours.
For an action of duration T, each feature vector has a total of 51 features, including:

- The distance in meters between the basket and each player when $t = 0, t = \frac{T}{2}$ and $t = T$. The initial position is chosen because the player who calls the play must be sure that everybody is on the correct position before executing it. Furthermore, it is also important to explain that the distance is always measured with respect to the rim the player is attacking to. These items correspond to a 1x9 vector as shown in Equation 4.2.

$$\begin{aligned}&[d_(p_1, rim, t = 0), d_(p_2, rim, t = 0), d_(p_3, rim, t = 0),\\ &\quad d_(p_1, rim, t = T/2), d_(p_2, rim, t = T/2), d_(p_3, rim, t = T/2),\\ &\qquad d_(p_1, rim, t = T), d_(p_2, rim, t = T), d_(p_3, rim, t = T)]\end{aligned} \tag{4.2}$$

- The angle in degrees between the baseline and the line that goes from the basket to each player when $t = 0, t = \frac{T}{2}$ and $t = T$. Besides, the absolute angle is also computed in the same temporal conditions by calculating the angle between a parallel line to the sideline placed in the centre of the court and the line that goes from the basket to each player. The reason for including both types of angles is that many plays can be executed on both sides of the court, so the absolute angle adds robustness in this case. These items correspond to a 1x18 vector as shown in Equation 4.3.

$$\begin{gathered}[\alpha(p_1, rim, t = 0), \alpha(p_2, rim, t = 0), \alpha(p_3, rim, t = 0),\\ \alpha(p_1, rim, t = T/2), \alpha(p_2, rim, t = T/2), \alpha(p_3, rim, t = T/2),\\ \alpha(p_1, rim, t = T), \alpha(p_2, rim, t = T), \alpha(p_3, rim, t = T),\\ abs(\alpha(p_1, rim, t = 0)), abs(\alpha(p_2, rim, t = 0)), abs(\alpha(p_3, rim, t = 0)),\\ abs(\alpha(p_1, rim, t = T/2)), abs(\alpha(p_2, rim, t = T/2)), abs(\alpha(p_3, rim, t = T/2)),\\ abs(\alpha(p_1, rim, t = T)), abs(\alpha(p_2, rim, t = T)), abs(\alpha(p_3, rim, t = T))]\end{gathered} \tag{4.3}$$

- The total displacement of each player in both $[0, \frac{T}{2}]$ and $[\frac{T}{2}, T]$ segments, which indicates if the player is standing still or not. These items correspond to a 1x6 vector as shown in Equation 4.4.

$$\begin{aligned}&[Disp(p_1, [t = 0, t = T/2]), Disp(p_2, [t = 0, t = T/2]),\\ &\quad Disp(p_3, [t = 0, t = T/2]), Disp(p_1, [t = T/2, t = T]),\\ &\qquad Disp(p_2, [t = T/2, t = T]), Disp(p_3, [t = T/2, t = T])]\end{aligned} \tag{4.4}$$

- The speed (in m/s) of every player in both $[0, \frac{T}{2}]$ and $[\frac{T}{2}, T]$ segments. This feature introduces temporal information to the vector and contextualizes the total displacement: if the displacement is high but the feature vector corresponds to a long play, speed shows that the movement is long but slow. These items correspond to a 1x6 vector as shown in Equation 4.5.

$$\begin{aligned}[&Speed(p_1, [t=0, t=T/2]), Speed(p_2, [t=0, t=T/2]),\\ &Speed(p_3, [t=0, t=T/2]), Speed(p_1, [t=T/2, t=T]),\\ &Speed(p_2, [t=T/2, t=T]), Speed(p_3, [t=T/2, t=T])]\end{aligned} \tag{4.5}$$

- The maximum distance in meters with respect to the basket of each player in both $[0, \frac{T}{2}]$ and $[\frac{T}{2}, T]$ segments. This feature is thought for detecting patterns in big players, which use to play close to the basket and do not take long-range shots; if the maximum distance of these kind of players is large, it is probably due to a screen they have set. These items correspond to a 1x6 vector as shown in Equation 4.6.

$$\begin{aligned}[&MaxD(p_1, [t=0, t=T/2]), MaxD(p_2, [t=0, t=T/2]),\\ &MaxD(p_3, [t=0, t=T/2]), MaxD(p_1, [t=T/2, t=T]),\\ &MaxD(p_2, [t=T/2, t=T]), MaxD(p_3, [t=T/2, t=T])]\end{aligned} \tag{4.6}$$

- The minimum distance between each pair of players in both $[0, \frac{T}{2}]$ and $[\frac{T}{2}, T]$ segments. Once again, this feature can help to identify if screens have been set during the play; moreover, it can also indicate the pair of involved players. These items form a 1x6 vector, displayed in Equation 4.7.

$$\begin{aligned}[&MinD(p_1, p_2, [t=0, t=T/2]), MinD(p_1, p_3, [t=0, t=T/2]),\\ &MinD(p_2, p_3, [t=0, t=T/2]), MinD(p_1, p_2, [t=T/2, t=T]),\\ &MinD(p_1, p_3, [t=T/2, t=T]) MinD(p_2, p_3, [t=T/2, t=T])]\end{aligned} \tag{4.7}$$

A visual representation of the included geometrical features can be found in Figure 4.10.

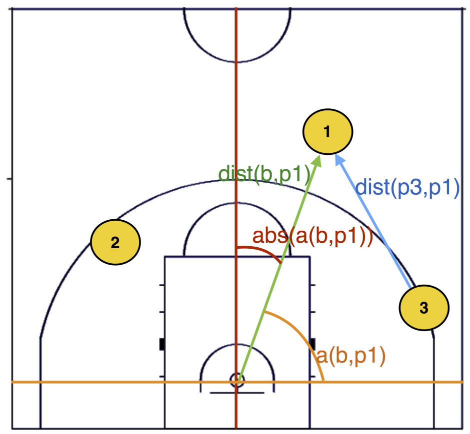

Figure 4.10: Visual explanation of four features, being *dist(b,p1)* the distance from the basket to *Player1*, *dist(p1,p3)* the distance from *Player3* to *Player1*, and *a(b,p1)* and *abs(a(b,p1))* the angle and absolute angle respectively between the basket and *Player1*.

Chapter 5

Evaluation

In this Section, different kind of results are shown and discussed:

- On the one hand, the precision when selecting players with the different explained techniques is detailed.
- On the other hand, the obtained accuracy when trying to classify plays is shown as well, comparing the gold standard method (picking players manually) with the (semi)automatic ones.
- Finally, a general recap is done based on the proposed Success Criteria of Section 3.2, which will indicate if the project's goals have been accomplished or not.

5.1 Three-Player Selection

When trying to automatically detect three-players, different results are obtained depending on the criteria that is being followed. As mentioned, three approaches were tested: classifying based on the speed, on the correlation given a template and on *blind* correlation. In all three cases, accuracy is an obtained percentage that checks if detected players coincide with the ones currently doing the exercise. The average accuracy over all the exercises is a weighted mean of the separate accuracy values, as the number of repetitions of each exercise is not the same.

$$Acc_{average} = \frac{22}{96} \times Acc_{E1} + \frac{22}{96} \times Acc_{E2} + \frac{14}{96} \times Acc_{E3} + \frac{21}{96} \times Acc_{E4} + \frac{17}{96} \times Acc_{E5}$$

5.1.1 Speed

The obtained accuracy for each exercise when picking players based on their velocity during the play is shown in Table 5.1.

	E1	E2	E3	E4	E5	Average
Speed	70.83	92.04	64.28	28.57	100	72.30

Table 5.1: Obtained accuracy based on speed. E1, E2, E3, E4 and E5 stand for the first (*floppy offense*), second (pick and roll), third (press break), fourth (post-up situation) and fifth (fast break) exercise respectively.

In order to have deeper understanding of accuracy metrics, the confusion matrix of all classifications is shown in Table 5.2.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	23	1	0	0	0	2	0	0	0	0
2	0	22	1	0	3	1	0	0	0	0
3	2	2	22	0	3	1	2	3	0	0
4	0	3	0	20	2	0	0	4	0	0
5	0	0	0	0	25	0	0	0	0	0
6	1	2	0	0	4	19	3	2	0	0
7	0	0	1	1	1	0	23	1	0	0
8	0	1	1	0	2	0	0	20	0	0
9	0	0	4	2	1	3	0	1	15	0
10	2	3	0	1	4	0	2	4	0	12

Table 5.2: Confusion matrix when detecting 3 players based on their speed.

As it can be observed, this method works fine for pick and roll and fast break sequences, but the accuracy of post-up situations is low. The reason is related to the comment in Section 4.1; during post-up sequences, data is gathered from both sides of the court, including exterior players practicing press-break, which is a much more frenetic exercise than post-up, thus boosting the speed of small-fast players. In Table 5.2, it can be seen that a lot of instances are incorrectly classified as *Player 5* and *Player 8* and that the algorithm has some difficulties when dealing with instances belonging to *Player 9* (11/26 misclassified instances) and *Player 10* (16/28). The logic behind this second drawback is that this couple of players is the manually tracked one. As mentioned in Section 4.1.2, manual tracking trajectories are much more smooth and less variant than the ones obtained with sensors, as players accumulate small distances every 40 milliseconds. Although tiny-short displacements could seem negligible, its accumulation is meaningful if the comparison is performed over the whole sequence in terms of speed, especially when dealing with interior-heavy players (that explains why the true positive rate of *Player 9* is higher than *Player 10*).

5.1.2 Pattern Correlation

Results obtained after computing Pearson correlation coefficients between templates and different signals are displayed in Table 5.3.

	E1	**E2**	**E3**	**E4**	**E5**	**Average**
Corr. Patt.	91.67	60.22	100	100	100	85.25

Table 5.3: Obtained accuracy based on a *Correlation Pattern* approach. E1, E2, E3, E4 and E5 stand for the first (*floppy offense*), second (pick and roll), third (press break), fourth (post-up situation) and fifth (fast break) exercise respectively.

A confusion matrix is displayed in Table 5.4 for further understanding of data.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	25	0	0	0	0	1	0	0	0	0
2	0	23	0	0	1	1	0	1	1	0
3	2	0	33	0	0	0	0	0	0	0
4	1	0	0	25	0	0	2	0	0	1
5	1	1	0	1	20	0	0	1	0	1
6	1	0	0	0	1	25	1	0	0	3
7	0	1	0	0	0	0	21	1	3	1
8	0	1	0	0	0	1	0	22	0	0
9	0	1	0	0	0	1	4	2	17	1
10	1	0	0	0	0	0	0	1	0	26

Table 5.4: Confusion matrix when detecting 3 players based on a *Correlation Pattern* approach.

Once again, the player with lower true positive rate is *Player 9*, the one manually tracked. Besides, it is interesting to see that *Player 9* is classified 4 times (out of 26) as *Player 7*, as both players have similar characteristics and play in the *point-guard* position. In this case, there are no problems when trying to detect *Player 10*, who is properly classified 26 out of 28 times. As it can be observed in Table 5.3, the obtained accuracy is high for all exercises except for pick and roll sequences. The main reason is that, as this action can be executed on both sides (right and left), there is not a valid template for all sequences. Therefore, another test was performed computing absolute distances in the Y-axis with respect to the middle of the court instead of the regular trajectory. Being $P1 = (x, y)$ the coordinates of a player, and (w_C, h_C) the size of the court, the absolute distance to the center of the court is:

```
if (y > (h_C/2)) {
    P1 = (x, y − (h_C/2)) }
else {
    P1 = (x, (h_C/2) − y)
}
```

Obtained accuracies can be seen in Table 5.5.

	E1	**E2**	**E3**	**E4**	**E5**	**Average**
Corr. Patt. (abs dist)	87.50	72.73	76.19	92.86	100	84.69

Table 5.5: Obtained accuracy based on a *Correlation Pattern* approach. E1, E2, E3, E4 and E5 stand for the first (*floppy offense*), second (pick and roll), third (press break), fourth (post-up situation) and fifth (fast break) exercise respectively.

Although accuracy increases (as expected) in pick and roll sequences (+12%), it decreases in press-break situations (-24 %). The reason of this drop is that the basis of fast-break sequences is to have one open player on each side of the court (right-left), so using absolute distances does not provide the appropriate data to detect it.

5.1.3 Blind Correlation

In the last type of automatic detection of the 3-involved players, *Blind Correlation* was used with no prior information. As mentioned, some signals are discarded before computing Pearson coefficients, in order not to take into account static players with low standard deviation. Using 22 as a threshold, those players that had a lower standard deviation value were filtered, resulting in the different accuracy results displayed in 5.6.

	E1	**E2**	**E3**	**E4**	**E5**	**Average**
Correlation	84.72	86.36	4.76	85.71	82.35	73.02

Table 5.6: Obtained accuracy based on a *Blind Pattern* approach. E1, E2, E3, E4 and E5 stand for the first (*floppy offense*), second (pick and roll), third (press break), fourth (post-up situation) and fifth (fast break) exercise respectively.

A confusion matrix of classification data is shown in Table 5.7.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	24	0	0	1	0	0	1	0	0	0
2	0	16	1	1	4	1	2	0	2	0
3	3	2	21	0	2	0	2	5	0	0
4	2	1	0	18	3	0	2	1	2	0
5	1	0	0	0	24	0	0	0	0	0
6	4	1	0	0	4	20	0	1	1	0
7	1	1	0	0	3	0	21	0	1	0
8	1	0	0	0	0	0	3	20	0	0
9	1	0	0	1	1	1	0	1	21	0
10	1	0	0	0	1	0	4	4	0	18

Table 5.7: Confusion matrix when detecting 3 players based on a *Blind Correlation* approach, setting a Standard Deviation threshold to 22.

As it can be seen, the precision for post-up situations is really low (once again because two different exercises were being played at the same time); nevertheless, accuracy is notable for the rest of exercises. For this reason, the Confusion Matrix shows several False Negatives in Interior Players (*Player 3, 4, 6* and *10*). To properly deal with post-up issues, some thresholding optimization was performed by:

- Creating two different standard deviation thresholds, one for exterior players (*Players 1, 2, 5, 7, 8, 9*), which tend to move all the time, and another one for interior ones (*Players 3, 4, 6, 10*), which stay rather static.
- Looking for the optimal threshold values by computing the accuracy taking all possible combinations into account (with a minimum of 1 and a maximum of 100).

The obtained results and corresponding thresholds are shown in Table 5.8:

	E1	**E2**	**E3**	**E4**	**E5**	**Average**
Correlation (optimized)	86.11	87.50	71.43	85.72	82.35	82.49
Thresh. Int.	14-19	19	1-12	1-100	11-19	(-)
Thresh. Ext.	44-45	4-5	95-100	1-7	2-6	(-)

Table 5.8: Obtained accuracy based on a *Blind Pattern* approach once the ideal thresholds are found (2nd and 3rd rows). E1, E2, E3, E4 and E5 stand for the first (*floppy offense*), second (pick and roll), third (press break), fourth (post-up situation) and fifth (fast break) exercise respectively.

Although the average accuracy is close to outperform previous approaches, using optimal thresholds has some drawbacks:

- Despite having the possibility of defining a general-optimal threshold for interior players, a generic value cannot be set for exterior players because of its variability.
- The 70% increase peak in post-up situations does not mean that the algorithm had learned the players to choose. Knowing that only interior players are performing the post-up exercise, the algorithm is just setting a really high threshold for exterior players; in this way, this type of players will not be taken into account when computing correlation coefficients. Actually, if the exterior threshold keeps increasing, there is a moment (values above 155) where accuracy equals 100 %, as neither interior defensive players nor exterior players are taken into account.

5.2 Play Classification

In terms of classifying different observations into a closed-set of plays, the *graphical input* gold standard method for player selection is used, and then compared to the *Pattern Correlation* approach.
A 96×51 matrix is obtained by extracting a 51-dimensional feature vector from each of the 96 observations. Nevertheless, considering the limited size of the dataset, the model cannot be trained as it is, because it might have non-relevant features that should be discarded to avoid over-fitting. Principal Component Analysis (PCA) [20] is applied in order to reduce data dimensionality and discard those components that are highly correlated. The eleventh first principal components are kept in order to account for 95% of the variance in the data. This procedure is visually explained in Figure 5.1.
In order to build the classification model, the *Classification Learner App* of Matlab was used. Besides, the *Principal Component Analysis* Matlab *build-in* function was used as well.

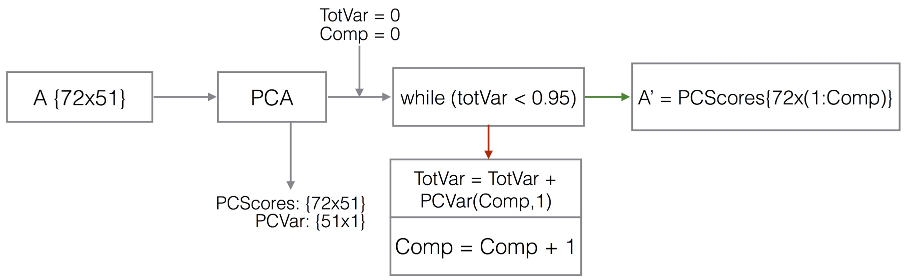

Figure 5.1: Steps that have to be followed in order to retain 95% of the dataset variance.

A 10-fold cross validation was used, obtaining **97.9 % accuracy** when using a Linear Support Vector Machine classifier [19] with a *One-vs-One* strategy to deal with multiclass classification.
The resulting Confusion Matrix can be seen in Table 5.9 and a Scatter Plot can be found in Figure 5.2.

	E1	E2	E3	E4	E5
E1	**22**	0	0	0	0
E2	2	**20**	0	0	0
E3	0	0	**14**	0	0
E4	0	0	0	**21**	0
E5	0	0	0	0	**17**

Table 5.9: Confusion Matrix using a *Graphical Input* approach: rows correspond to true classes and columns to predictions. E1, E2, E3, E4 and E5 stand for the first (*floppy offense*), second (pick and roll), third (press break), fourth (post-up situation) and fifth (fast break) exercise respectively.

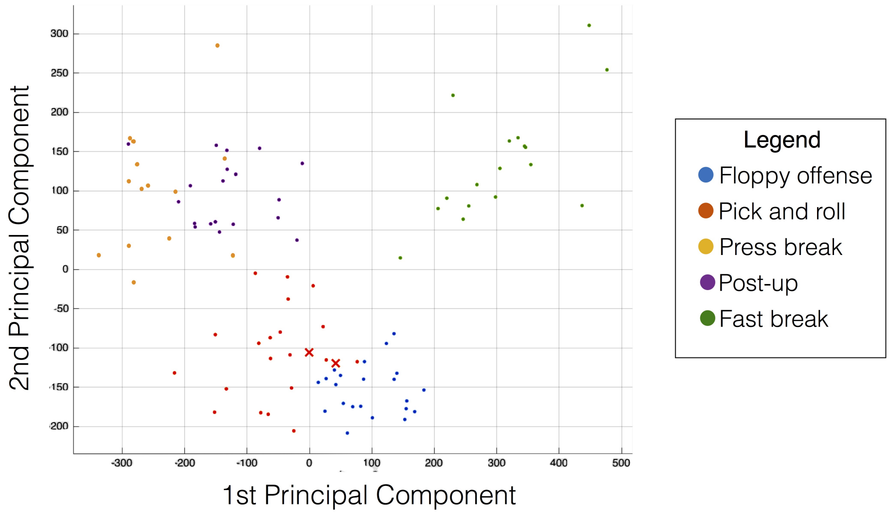

Figure 5.2: Scatter Plot obtained after picking players manually. In order to represent data visually, PCA has been applied to keep only the two first Principal Components.

Besides, other metrics are displayed in Table 5.10 in order to provide a complete picture of the classifier's performance. These metrics include:

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$\text{F1-score} = \frac{2 \times TP}{2 \times TP + FP + FN}$$

	E1	**E2**	**E3**	**E4**	**E5**	**Mean**
Precision	0.96	1	1	0.95	1	0.98
Recall	1	0.91	1	1	1	0.98
F1-Score	0.98	0.95	1	0.98	1	0.98

Table 5.10: Precision, recall and f1-score of separate classes, and weighted mean obtained after picking players manually. E1, E2, E3, E4 and E5 stand for the first (*floppy offense*), second (pick and roll), third (press break), fourth (post-up situation) and fifth (fast break) exercise respectively.

Although the obtained performance is notable (only two instances are misclassified), this performance has been obtained with a manual approach. In order to test how the classification deals with a semi-automatic approach, where mistakes are committed when selecting the three involved players, the same 96×51 matrix was obtained using the *Pattern Correlation* method (without using absolute distances). Using the same classification method (10-fold cross validation Support Vector Machines with a *One-vs-One* strategy), **89.6 % accuracy** is obtained; the resulting confusion matrix is displayed in Table 5.11, and a Scatter Plot can be observed in Figure 5.3.

	E1	E2	E3	E4	E5
E1	**20**	0	0	2	0
E2	0	**17**	0	4	1
E3	0	0	**13**	0	1
E4	0	1	0	**20**	0
E5	0	0	0	0	**17**

Table 5.11: Confusion Matrix using the *Pattern Correlation* approach: rows correspond to true classes and columns to predictions. E1, E2, E3, E4 and E5 stand for the first (*floppy offense*), second (pick and roll), third (press break), fourth (post-up situation) and fifth (fast break) exercise respectively.

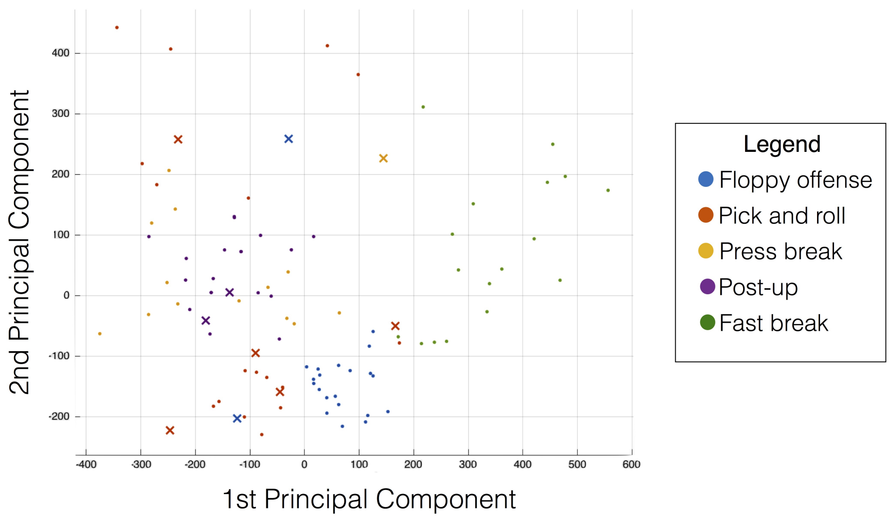

Figure 5.3: Scatter Plot obtained after picking players in a semiautomatic way. In order to represent data visually, PCA has been applied to keep only the two first Principal Components.

Once again, *Precision*, *Recall* and *F1-score* metrics are shown in Table 5.12.

	E1	**E2**	**E3**	**E4**	**E5**	**Mean**
Precision	1	0.94	1	0.77	0.89	0.92
Recall	0.91	0.77	0.93	0.95	1	0.91
F1-Score	0.95	0.85	0.96	0.85	0.94	0.91

Table 5.12: Precision, recall and f1-score of separate classes, and weighted mean obtained after picking players in a (semi)automatic way. E1, E2, E3, E4 and E5 stand for the first (*floppy offense*), second (pick and roll), third (press break), fourth (post-up situation) and fifth (fast break) exercise respectively.

As expected, the *pick and roll* situation is the one with a lower True Positive Rate (17/22); the logic behind this fact is the weak detection of the three involved players in this kind of play. As it was seen in Table 5.4, only 60 % accuracy was obtained in pick and roll sequences using a *Pattern Correlation* approach; even not selecting the appropriate players, the classification of these actions remains acceptable.

5.3 Success Criteria Validation

In this Section, the different success criteria hypothesis are validated individually.

1. *Gathering an acceptable experimental dataset to work with. This dataset should include at least 10 observations of 4 different plays and the tracking of 5 players minimum.* The gathered dataset (Section 4.1) contains more than 14 observations of 5 different plays, with tracking data of 8 players, so this goal has been accomplished.

2. *Finding a fast way to manually track the ball. For a basketball sequence of T duration, tracking the ball in all frames should not take more than $1.5 \times T$.* A homemade program based on hot keys has been presented in Section 4.2, and some user-testing tasks prove that the speed of labelling a sequence of T duration takes $1.15 \times T$. This objective has been reached.

3. *Managing to parse all data from sensors to obtain a 2D representation of data without occlusions; by visualizing this kind of representation, a basketball coach must be able to identify what is going on on court.* As shown in 4.3, the synchronization of tracking signals has been properly performed, and visual customized representations are extracted. In these types of display, not only the position of the players is shown, but also their trace, the ball movement, and shooting positions. Again, this goal has been attained.

4. *Extracting at least 6 meaningful types of advanced statistics and create different visual displays.* 12 different advanced statistics are explained in Section 4.4, and both heat maps and shot charts are shown in Section 4.4.1, thus accomplishing this purpose.

5. *Designing a (semi)automatic method to pick only the involved players of an action.* Three different ways of selecting only involved players are detailed in Section 4.5.2 (based on speed, and pattern or blind correlation), obtaining more than 80 % accuracy when picking players. Although it still can be improved, the goal has been accomplished.

6. *After a feature extraction process, training a Machine Learning model that could classify plays with notable accuracy. Although it is difficult to define numerically what notable accuracy means, the model is expected to classify the whole dataset better than a non-basketball-expert.* Obtained results show 97.9 % accuracy when classifying plays, which is good enough (94 out of 96 observations are correctly classified). Most likely, a non-basketball-expert would not attain this performance, so the objective has been fulfilled.

Chapter 6

Discussion

Having tested the three different approaches to pick the three involved players in each action, a general recap could be the following one:

- *Speed* outperformed other approaches in pick and roll sequences and worked great for fast breaks, but it did not perform that well for the other exercises.
- *Pattern Correlation* proved to work fine for all types of exercises except for pick and rolls. In order to deal with the main issue of this type of actions (execution of the exercise at both right and left sides of the court), a test was performed using absolute distances, obtaining higher accuracy for pick and roll situations but dropping in the press break exercise.
- *Blind Correlation* provided good results in all exercises aside from post-up situations, where less than 5% accuracy was obtained. In order to have a more balanced method, different ways of finding optimal thresholds were tested, but it was not possible to generalize into a single general value.

Besides, it is obvious that having data of 10 players would work much better than manually tracking two of them, especially when selecting three players with a method based on speed.
Despite the good performance of the classification model, a higher number of observations and classes would be required in order to build a professional system. In addition, combining *Play Classification* with imperfect semiautomatic methods for detecting the three involved players in each action produced a drop in overall accuracy, but the system still managed to properly deal with almost the 90 % of actions, which is encouraging. It would definitely be interesting to have more *unpredictable full-court* exercises, where the objective is not to follow certain patterns (like some of the included classes), but to accomplish a goal no matter how. Press-break could be considered as an *unpredictable full-court* exercise and it was properly classified, but without having more data, it cannot be proved that the algorithm would generalize to other situations. Moreover, it would also be interesting to have subclasses, since the offense may change their

strategy in real-time depending on their opponents defense and plays usually have second and third options (*i.e. pick and roll/pop*).
Once plays are correctly classified, advanced statistics like the ones explained in Section 4.4 can be extracted with ease: for example, the coach is able to know the pass speed of a certain player in all the *post-up situations* during a game.
Additionally, manual ball tags proved to be useful, and not only to have the ball in a 2D representation, but also to temporally segment repetitions during the exercises. It might be argued that, while the whole purpose of the project is to substitute cameras for sensors, a camera has been used in the presented experiment; although it is a valid reasoning, the purpose of having a single camera is just to support sensor data and not to perform automatic tracking. Moreover, in the case of big companies, their camera setup includes a minimum of 6 high-quality fibre-synchronized cameras and, in the presented test, a simple camera was used (even a mobile phone recording could have been helpful). The best solution is adding a positioning sensor to the ball too, which must not change its weight. As mentioned, there are companies such as Wilson that are starting to commercialize this type of basketballs [38], so it is a feasible solution.

To conclude, it can be said that a new method to automatically extract advanced statistics based on sensors data has been detailed. Even though working with sensors might have drawbacks (such as difficulties when trying to scout another team), it is a much cheaper solution than multi-camera configuration systems like the ones installed in NBA arenas, and it is an attainable way to start extracting advanced statistics in Europe. For the purposes of this project, a dataset containing both video and tracking data of 30 minutes of the *Under-21 Valencia Basket Club*'s practice was recorded using NBN23's technological resources. In these recordings, there were a total of 96 different actions of the following classes (types of basketball plays): *floppy offense*, pick and roll, press break, post-up and fast break situations.
Knowing that the basis of the automatic extraction of statistics is the identification of different basketball plays occurring on court, these steps must be followed:

1. Labelling the frames containing events related to the ball with simple tags (receive, release, substitutions...).

2. Merging all tracking information in a single matrix, taking synchronization into account by sorting timestamp values.

3. For visualization purposes, mapping the players' court coordinates into pixels.

4. (For each play) Selecting three involved players in the action with a manual approach based on graphical inputs (high confidence, but requires user interaction) or an automatic method (lower confidence, but without any manual procedure).

5. (For each play) Extracting meaningful basketball features of the involved players in order to build a 1×51 feature vector.

In the presented test, once all feature vectors have been merged into the same data matrix, PCA has been applied in order to avoid model over-fitting, keeping the 95% of the observations' variance. Using a 10-fold cross validation and a Linear Support Vector Machine algorithm, 97.9% accuracy is obtained when trying to classify the whole training data using a manual approach to select players. Another test was performed with an automatic selection of players (using the explained *Pattern Correlation* method), obtaining 89.6 % accuracy.

6.1 Future Work

In order to improve the presented work, more data has to be recorded or obtained somehow (*e.g.* video games), containing a larger variety of observations and classes. Besides, it would be interesting to track the ball with a sensor instead of manual annotations. Likewise, more sequences corresponding to 5-on-5 games must be tested, as those actions will be less predictable; in addition, it would also be desirable to include defensive strategies. Another weakness of this project is the lack of a robust method to select the 3-involved players of each action, as all the tested ones have their own drawbacks and do not achieve more than 90% accuracy. With thousands of examples, Convolutional Neural Networks would provide higher accuracy and data could be divided into training and testing sets.
Another interesting purpose could be applying the same technique to recognize patterns in other sports, especially in soccer, where European clubs have high salary caps.

Appendices

Appendix A

Entrepreneurship Lectures: Yuzz Program

While this project was being performed, I got accepted in an entrepreneurship program called Yuzz (`http://yuzz.org.es/`). This program is sponsored by *Banco Santander*, which is one of the most important banks in Spain, and the goal is to encourage university students not to be afraid of starting a business from scratch. Yuzz is carried out in 52 different universities in Spain, and a tough selection process is faced in all of them; in my case, I applied to the program at Universitat Pompeu Fabra, where I studied my bachelor degree. This course has a duration of 5 months, and 8 hours of class are taught each week; besides, two supervisors are assigned to each project based on the relation between their background and the projects aim. The content of this program is divided into 9 learning units:

1. **Opportunity**: generation and identification of opportunities based on a market analysis.
2. **Personal Autonomy**: reflection tools to face the fact of being an entrepreneur: personal vision, values and risks.
3. **Project**: structure of company resources; *hands-on*, project managements and business plan.

4. **Clients**: methods to transform an idea into an start-up model, studying the initial development stages.

5. **Leadership**: required entrepreneur skills in order to build a team able to develop his/her business: decision-making process, motivation, negotiation or conflict resolution.

6. **Operations**: legal and fiscal actions to be taken into account.

7. **Business Development**: methods to acquire clients in national and international markets.

8. **Human Talent Management**: organize people management and their actions inside the company.

9. **Action Plan**: how to build a realistic and feasible action plan, including funding and a project for investors.

The final project to be delivered was a detailed executive summary, which is encapsulated and translated in Section A.1. The software that includes the algorithms presented in this report is called **Spottern**, due to the mix of *sports* and *patterns*. Besides, whilst being a *Yuzzer*, I had the chance to attend some entrepreneurship events, such as *Mobile World Congress* or *Biz Barcelona*. In this second event, I got selected to participate in the *III Elevator Pitch* contest, where 50 university students presented their idea and business model in less than 2 minutes; I was actually awarded with an honorable mention.

A.1 Executive Summary

Need

Nowadays, the way basketball coaches prepare games and practices is tedious. According to tested hypothesis, European coaches need (at least) 7 hours to prepare a game, which include the edition of audiovisual material, the revision of these videos to annotate the most important aspects of the games, and the effective transmission of this knowledge to the court. Knowing that some European teams play 70 games a season, the coaching staff requires 490 hours to prepare matches.

Value Proposition

Spottern is a software based on tracking data of basketball players; this data is obtained using accelerometric wearable sensors. Thanks to artificial intelligence techniques, the extraction of advanced statistics can be automatized, thus reducing to 2.5 hours the required preparation of a game. This software is thought to optimize team resources in order to accomplish goals such as winning more games.

The Product

As it can be seen in Figures A.1, A.2 and A.3, Spottern contains different features the coach when preparing the game.

- 2D Representations, which might help watching the game without occlusions. Besides, it might be possible (depending on the license the user has) to integrate video, in order to see both the game and the 2D representation at the same time.
- Visual statistics such as heat-maps or shot charts, which can be filtered depending on players or temporal stages.
- Advanced customized statistics.

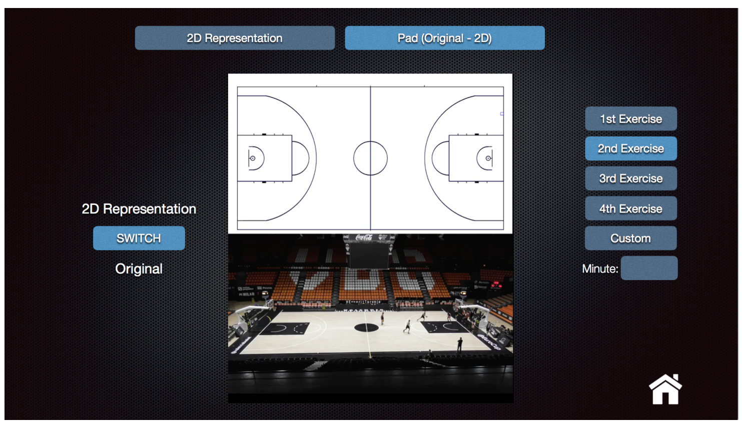

Figure A.1: 2D Representation of the game.

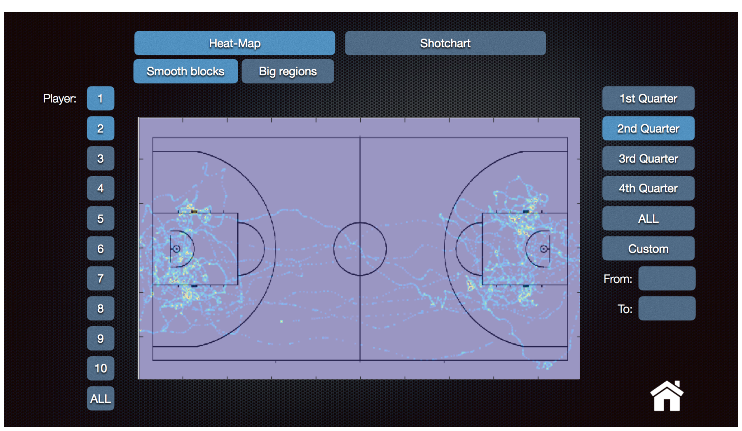

Figure A.2: Visual statistics.

Figure A.3: Combination of advanced statistics.

Besides, Artificial Intelligence models have been built (not integrated yet), in order to distinguish a closed-set of plays. Right now, the algorithm has 97.9% accuracy when trying to distinguish between 5 common basketball plays, which constitute the basis of the 70 % of total plays.

Competitors

The main competitors of Spottern are SecondSpectrum and STATS, which are American companies and official providers of statistics and tracking data to the NBA. Their systems are based on a 6-camera configuration in the ceiling of the stadium, and all cameras are synchronized with fiber wires and remotely controlled. These companies used to offer their benefits to NBA teams; in 2011 an annual STATS license cost 1.100.000 \$. However, in 2014, the NBA decided to buy their services, installing their multi-camera setup in all 29 arenas; nowadays, the NBA pays 41.000.000 \$/year to both companies.
Nevertheless, neither SecondSpectrum nor STATS have European clients, for two main reasons: (a) the size of the European stadiums is not high enough to install their camera-setup and (b) the salary caps of European teams are much lower than in the United States. The *poorest* team in the NBA is Utah Jazz, with a salary cap of 69 million dollars; the difference with (*i.e.*) Spanish clubs is evident when looking at Table A.1. Clubs with a salary cap lower than three million dollars cannot spend 1/3 of their total budget on technological resources.

Salary Cap	Teams
<3.000.000 $	7 (Fuenlabrada, Tenerife, Obradoiro, Zaragoza, Sevilla, Badalona, Manresa)
{3.000.000 - 8.000.000} $	4 (Estudiantes, Andorra, Bilbao, Murcia)
{8.000.000 - 15.000.000} $	4 (Valencia, Gran Canaria, Unicaja, Baskonia)
{20.000.000 - 30.000.000} $	2 (Barcelona, Madrid)

Table A.1: Spanish salary caps.

Partners

NBN23 is supposed to be the main partner of Spottern. This emerging Spanish start-up is trying to provide technological resources to revolutionize basketball courts. Their main product is a system based on accelerometric sensors to gather tracking data from all players without requiring a camera-system; the price of this service is 30.000 $/year. However, they are facing a problem: the data they are currently gathering can only be transformed to speed, acceleration and workloads. Without any kind of tactical content, this tool is not so appealing for coaches. Therefore, the interests of both companies to merge forces would be the following ones:

- Spottern would get the required technology to obtain tracking data and perform tests. Entering the market together with NBN23 would be advantageous in terms of prestige.
- NBN23 would enrich their current software with tactics and strategies.

Client Archetypes

The different client archetypes are **Clubs** and **Federations**, as shown in Table A.2.

Client Segment	Archetype	Do they have resources?	Do they have the need?	How to reach them?
Clubs	First Division Teams	Yes	Yes	In Situ + 3 free months
	Second and Third Division Teams	No	Yes	In Situ
	Youth Teams	Some	Not necessarily	Landing Page
Federations	Official Championships	Yes	Yes	In Situ + 1 free tournament
	Non-official Championships with Several Teams	Yes	Not necessarily	In Situ

Table A.2: Different clients to be reached

Business Model

The product to be offered to clubs or federations must be split into several software versions, including different types of specifications.
On the one hand, different annual licenses would be offered to **Clubs**, as seen in Table A.3:

	Spottern Basic	**Spottern Bronze**	**Spottern Silver**	**Spottern Gold**
Number of Included Sensors	5	5	10	15
2D Visualization	Yes	Yes	Yes	Yes
Visual Statistics	Absolute Heat-Maps	Customizable Heat-Maps (players), and Shot Charts	Customizable Heat-Maps (players-time), Shot Charts and Work Loads	Customizable Heat-Maps (players-time), Shot Charts and Work Loads
Statistics Extraction	Yes	Yes	Yes	Yes
Statistics Customization	No	Yes	Yes	Yes
Game Comprehension	No	No	7 included plays	20 included plays
Video Integration	No	No	No	Yes
Free Technical Support Included	None	10 hours / year	15 hours / year	20 hours / year
Price / year	6.000 $	12.000 $	25.000 $	40.000 $

Table A.3: Specifications of several Spottern versions, each one with an associated price.

On the other hand, the business model for **federations** organizing tournaments would be much different, and the cost would depend on the following factors:

$$\text{Price} = \text{Version} \times N_{\text{courts}} \times \text{Tournament}_{\text{duration}}$$

The Version of the software determines, once again, the enabled parts of the software the user will have, and it is a fixed variable (Basic 200, Bronze 400, Silver 1100, Gold 2250); N_{courts} indicates the number of courts where an installation of receptors is required and also determines the number of sensors to be delivered; $\text{Tournament}_{\text{duration}}$ just indicates the duration of the competition in days. As an example, let's consider the implementation of Spottern in the *TIM Andorra* tournament. In this 3-day tournament, there are 6 courts, and a total of 128 teams (1536 players); hiring Spottern's services for this type of tournament would have the costs displayed in Table A.4:

	Basic	**Bronze**	**Silver**	**Gold**
Number of Sensors	30	60	60	90
Price	3600 $	7200 $	19800 $	40500 $
Price / Player	2.34 $	4.68 $	12.89 $	26.36 $

Table A.4: Different types of prices in TIM Andorra depending on Spottern's bought version.

Action Plan and Costs

Finally, the important milestones and their cost are the following ones:

1. *September 2017* - massive data gathering to expand the dataset, either installing the system in a club or obtaining virtual data (from videogames). The associated costs can be seen in Table A.5.

A. Installation		**B. Virtual Data**	
Production of 15 sensors	375 $	*Meetings with 2K Spain*	
Production of 8 receptors	250 $	Trips to Madrid	225 $
Installation	300 $	License	500 $
Data storage in the cloud	50 $	*Recording Games (500 games of 20 minutes)*	
		Price/game	2 $
Total	1075 $		1725 $

Table A.5: Associated costs (point 1).

2. *November 2017* - Interface development, with the required characteristics of a *Minimum Viable Product* ready to be used. The associated costs are shown in Table A.6.

	Cost	**Total**
During 1 year		
Rent studio (UPF Business Shuttle, 9 squared meters)	918 $	1218 $
Other expenses	300 $	
During 2 months		
Code porting to .NET	850 $	4450 $
Back-end tasks	1800 $	
UX-engineering tasks	1800 $	

Table A.6: Associated costs (point 2).

3. *Year 2018* - trips to 3 international tournaments: ANGT Hospitalet (Euroleage), ANGT Coin (Euroleague) and TIM Andorra, with the purpose of meeting potential clients. The associated costs can be observed in Table A.7.

	$ / person / day	**Total**
Meals and transportation (2 people, 3 tournaments of 3 days)	30 $	540 $
Housing (2 people, 2 tournaments, 2 nights)	70 $	560 $
Printing business cards and flyers	100 $	
Total		1200 $

Table A.7: Associated costs (point 3).

4. *Year 2018* - trips to 4 Spanish cities where a first-division team can be found (Barcelona, Malaga, Valencia and Madrid), in order to meet potential clients and show a demo. The associated costs are shown in Table A.8.

	$ / person / day	**Total Cost**
Meals and Transportation (2 people, 4 cities, 2 days)	30 $	480 $
Housing (2 people, 3 cities, 2 nights)	70 $	840 $
Sensors & Receptors Fabrication	0 (already fabricated)	
Installation Externalization	900 $	
Total		2200 $

Table A.8: Associated costs (point 4).

5. *June 2018* - Feedback integration and development of a beta version of the software to be launched (1 month). The associated costs can be seen in Table A.9.

Hire a senior Engineer (Artificial Intelligence)	1000 $
"Small" front-end Tasks	750 $
Total	1750 $

Table A.9: Associated costs (point 5).

6. *July 2018* - Product and web launch. The associated costs are displayed in Table A.10.

Management costs (society foundation)	3500 \$
Intellectual Property Register	14 \$
Web Domain	15 \$ / 1 year
Hosting	300 \$ / 1 year
Total	3829 \$

Table A.10: Associated costs (point 6).

Costs Recap.
The required investment (from September 2017 until September 2018) intended for meeting potential costumers, testing and launching a totally-usable product would be 17477 \$. Adding the salaries of employees such as the CEO, the CTO, and a Marketing leader, these costs would increment to 53567 \$. The breakeven would potentially be reached after selling three club-Spottern licenses in September 2018 (1 Silver and 2 Bronze, most likely) plus the implementation of this software in a tournament such as ANGT Hospitalet (a Silver or a Gold license). More documents proving these facts are available on demand.

Appendix B

Diary of my Master Thesis

Week 1: January 23rd-27th 2017
Having met all the team from the LASSIE lab, this week has been a little bit chaotic from the point of view of "progress", as I have been desperately trying to find a proper annotated dataset to work with. To sum up:

- My main intention was to use the VATIC basketball dataset (`http://web.mit.edu/vondrick/vatic/`) but the link was broken; I contacted the authors and they told me that there was a huge drive crash and it was impossible to recover the files.
- I found other datasets: OSUPEL (`http://blogs.oregonstate.edu/osupel/dataset/`) and APIDIS (`http://sites.uclouvain.be/ispgroup/index.php/Softwares/APIDIS`), being the second the most interesting one; however, I contacted both research groups to obtain the whole dataset and I am still waiting for an answer.
- I found another dataset in the paper called "Detecting Events and Key Actors in Multi-Person Videos" (`http://basketballattention.appspot.com/`), which had a lot of annotated videos from Youtube; I started parsing data from the csv files they provide to check how did it look like. However, I faced two main issues: the ball was not tracked at all and the positions of the annotations were given with respect to the frame, which meant that there were no references, and that pannings and zooms were not taken into account.
- I contacted two companies that provide data to the NBA: on one hand, I called STATS LCC (`https://www.stats.com/`) several times, which told me that I cannot use the data they provide to the teams without a really expensive license (they did not even tell me which the price was); they actually sent me some sample xml files containing statistics, but without tracking data. In fact, I even contacted a Catalan data analyst working

in the NBA (76ers), but he could not send me the data I wanted. On the other hand, I am still waiting an answer also from SecondSpectrum (`https://www.secondspectrum.com/`); I found out that one of the creators of the software they use is Pascal Fua (`https://people.epfl.ch/pascal.fua/bio?lang=en`).

- I had a meeting with two engineers who were willing to help me, and they introduced me to a Valencian company they are working with called NBN23 (`https://www.nbn23.com/`), which also provide tracking data extracted from sensors. I contacted them; as it is not such a big company as STATS, I do not think they will have such complicated regulations and policies.

Week 2: January 30th - February 3rd 2017

- I started annotating data of basketball video sequences with the tool I used in the study job I had last year. I dragged rectangles over all the 10 players and the ball extracting just 1 frame/second. However, this task is extremely tedious, and after 6 hours of tagging, I only had 4 minutes of actions (the game lasts for 40 minutes). Besides, the position of the rectangles does not provide much information, because it tells in which pixel in the image a player can be found, but there is no relation with the court itself.
- I held a meeting with the Valencian company NBN23 (in Barcelona) on Wednesday, and they are willing to collaborate with me :) (great news!). They have their sensor-system installed in 4 professional basketball courts and they will provide me with data to work with (in exchange of future collaboration if my system works). I am going to Valencia to gather data next Tuesday.
- The only con of working with the data NBN23 will provide is that their system does not track the ball, so this task must be done manually. I tagged ball events using a basic codification; for example: a15IN means that the player whose number is 15 and belongs to team "a" has received the ball. Tagging the ball manually is not that hard: I tagged 10 game minutes in less than 90 minutes.

Week 3: February 7th - February 11th 2017

- As I explained last week, I went to Valencia to talk with NBN23, and the truth is that everything went great. They are really willing to collaborate with me and (hopefully) next week they will send me some videos and data. Their long-term idea is to use my algorithms in their software as a tool for coaches to have advanced statistics of the players.
- Although NBN23 told me that they will soon have a ball with sensors (that will provide automatic ball tracking) I kept working on the simple manual annotations of the ball. These annotations include:

1. Substitutions: the goal is to know which are the five players on court. These actions are tagged with: “X-team-NUMBER-IN/OUT”; for example: Xa15IN means that the player 15 of the “a” team is on court.
2. Ball movement: the goal is to track the ball by using knowing who has it. These actions are tagged with: “team-NUMBER-IN/OUT”; for example: b6IN means that the player 6 of the “b” team received the ball. Apart from that, I also tag the frames where the ball reaches the rim with “BKT”.
3. Pauses are also tagged with a “STOP” label, and “RESUME” when the game starts again.

- I also started to extract advanced statistics from this data (saved into a .mat file *a posteriori*), such as the number of times a player touched the ball, the amount of seconds a player retained the ball, the number of passes/shots/free throws or how many times a player was substituted.

Week 4: February 13th - February 17th 2017

- I started thinking about the vectors I will use to represent what is going on court in numbers, in order to distinguish different types of plays. This vector (which will be at least 1x40) will contain: the position of every player with respect to the center of the court, the distance between each pair of players and their velocities; for a play with a duration of T seconds, these values will be set for t = 0, t = T/2 and t = T.

- I wanted to speed up the process of tagging actions, so I created a button-pad with ERIC Sports (the software produced by the company I was doing the internship in the 9th semester), but a button-pad was not the best idea from the *user-experience* point-of-view. For this reason, I created a ”homemade tagging-machine” with Matlab that can be used with hotkeys, where there is no need of writing or clicking anything. This function reads frames from a certain folder, and using the arrows, you can go through them at different speeds (1, 5, 10. . .); moreover, just by pressing a key, the corresponding annotation is automatically generated. Tagging the actions of one team in 10000 frames (16,67 minutes) took me 23 minutes; it is not real-time, but close enough.

- Although I still have not received videos nor data from NBN23 (I talked to them and they had a really busy week because of the basketball King’s Cup in Spain), I created a basic linear interpolation function to animate the video sequences. This function will be used to display the movement of the players once I have the positions of them on court.

Week 5: February 20th - February 24th 2017

- I received a first *.csv* file with ”practice” data of three different players, so I build my mapping functions to convert those values into court positions.

I found out that the origin of their reference system is the middle of the court, so my first step is to identify the quadrant where the player is, and then perform the appropriate conversion (meters to pixels).

- I started displaying the player's positions (something I will need in order to validate results) on the court, and I thought about several ways of showing them: (a) basic movement of the players without lines, (b) movement with each player leaving a trace, (c) same movement with a trace updated every 33 frames, (d) movement with a line that goes from the initial position of the player to the actual position.
- As it took me a lot of time to extract the frames of the resulting video, I checked what happened with my program, and I found out that this slowness was because of the Matlab built-in *line* and *rectangle* functions. In order to solve it, I created manual drawing functions by accessing to the value of a certain pixel and changing it. The improvement is evident: before creating these functions, it took me about 2 hours to extract 350 frames; now, it takes me only 50 seconds.
- I also created functions to compute a feature vector containing not only the speed of the players but also the distance between those. This feature vector will be the one to be used in order to distinguish which type of play is going on, but I still need to work on it (my goal for next week).

Week 6: February 27th - March 3rd 2017

To be honest, I had a really busy week, because I obtained an accreditation for the Mobile World Congress, a huge event that was taking place in Barcelona. The truth is that it has been an awesome experience, but a little bit overwhelming. However, I still made some progress in my project:

- I solved the timestamp decodification issues. Basically, I did not know the units they were using, and I found out that I have a sample every 40 ms, which means a framerate of 25 samples/second/player.
- I build a first version of the feature vector. Having set the beginning and the end of a play in seconds, I divide the sequence into 2 parts: [t= 0, t=T/2], [t=T/2, t=T] and I compute some characteristics: the distance and angle of each player with respect to the basket at t=0, t=T/2 and t=T, the total displacement and the mean speed in both parts and the distances between each pair of players.

Week 7: March 6th - March 10th 2017

- I created a function in order to generate heat maps from the tracking data; in this function, there are two inputs: the first and the last frame where positions will be accumulated in each pixel. Besides, knowing that the court I use as a reference measures around 1000x600 pixels, it is very difficult to have accumulations in individual pixels, so the result does not look like a heat map (instead, it is the trace of each player). For this reason, I adapted my code to be able to establish bigger regions in order to generate the heat map; it is crystal clear that if the accumulation of values is done over neighbourhoods, more values/bin will be obtained.

- I included ball information in the feature vector. My decision was to add the % of time that each player kept the ball during the play (at first, I included the total number of seconds, but as it may differ from one action to another one, I discarded it and I chose the percentage instead).

- I generalised the feature vector for five players; at this point, its size is 1x90. Moreover, I started thinking about the plays that I wanted to detect and classify, as well as about specific features that would help me to discriminate.

Week 8: March 13th - March 17st 2017

- I started working with heat maps using Gaussian blur filters; more concretely, I worked with 3x3 and 5x5 neighbours, and I obtained displays with nice transitions.

- Without data, I decided to stop thinking about feature vectors, because all decisions I was making were assumptions, so I wrote important parts of the report, such as the Introduction and a part of the State-of-the-Art. On one hand, in the introduction, I described the importance of the project and the collaborations with UPC and NBN23; on the other hand, in the Previous Work section, I summarized 9-related papers (topic, need, methods, results, reproducibility, limitations and relation to my project).

Week 9: March 20th - March 24th 2017 (Data received!)

- I deeply analyzed the data they gave me; it contained four parts of a practice (of a really good team actually) with several exercises: from basic movements of 2 offensive players without any defending ones until 10 minutes of real game situation (5 on 5).

- As expected, they sent me not only the video but also the tracking data of 8 of the players in different csv files that had to be parsed and mapped into a 2D court; besides, I also had to identify which sensor corresponded to each player (I did not have this information). When I ran the functions I already created something weird happened: the movement of the "mapped players" did not make any kind of sense, it seemed like random movements that made impossible the recognition task. The reason was quite simple: there was a timestamp that I did not take into account in my first attempts and all the files were unsynchronised. I managed to introduce the timestamp feature in my parser and it seemed to work; however, it was not 100% accurate, because a video frame comprises an interval of 40 milliseconds, and different shifts between sensor emissions may cause short delays.

- Once I obtained a *draft result*, I was able to recognise all players and identify which were those two that did not wear the sensor.

Week 10: March 27th - March 31st 2017

- The video I was working on last week had an accumulated time offset because of the way I was sampling data. I corrected that, as the delay might ended being larger than 1 second, and in these kind of sport sequences, this is not negligible.

- I manually tagged (with the program I designed) the ball for the 4 sequences. It took me larger than expected mainly because there were two players that looked the same and I had no way to distinguish them (same jersey, same shoes, "same" height...). I also had to synchronise those annotations with the existing synchronisation, which was tough because many time-units had to be taken into account.

- I decided to include tracking information of the two players that did not wear the sensor, so I manually annotated them too (for every 0.1 seconds, I dragged a bounding box over the position in the court they were staying).

- Having the appropriate sampling and all players, I updated the 2D-mapping, including: (a) the ball position based on the .csv file generated with the obtained annotations, and (b) I painted a X mark in a certain position where a shot had been attempted. In the obtained results, the yellow and the gray squares correspond to the manual-introduced players; besides, the pink and the black squares correspond to the above-mentioned *similar players* (a couple of actions have to be corrected).

Week 11: April 3rd - April 7th 2017

- I corrected the wrong actions of the videos I was working on you last week (the mapping was not being performed properly with the manually-introduced players).

- In order to have a better understanding of the 2D actions, I decided to differentiate somehow the players that were actually doing the exercise. My first attempt was changing the size of the rectangles, but it was not the best idea because they were confused with the ball, so I decided to make the width of the rectangle edges thicker.

- I updated the algorithms to extract nice-to-have basketball statistics that could help me distinguishing between actions, such as the number of received/given passes, the number of seconds each player retain the ball, number of shots (actually, I created a function to distinguish between open/contested shots given the distance of all the other players), and so on.

- I started extracting features, but I decided not to take all 5 players into account: in a certain basketball play, there are not more than 3 players involved in the 90% of the cases, so having features of 5 players would add noise to my vector; in order to simplify, I decided to start again considering only 3 players/action, which reduced the size of my feature vector.

- As the timestamp values were really confusing and non-trivial (different fps and initial timestamp reference value), I designed a function to temporally segment actions in one video to be able to process an specific play instead of the whole video every time.

Week 12: April 10th - April 14th 2017

- I changed my program in order to pick three players and extract features; I used a graphical input and click three times, selecting the closest players to each click.

- I extracted the 24 feature vectors of the first exercise in the first video, which is a simple flow exercise to practice a certain play with 3 offensive players. This play is only executed on the right side of the court.

- I extracted the 22 feature vectors of the second exercise in the first video, which is another simple flow exercise to practice another different play with 4 offensive players and 0 defensive (note that one would be left behind with the 3-player strategy). This play can be executed both on right and left sides of the court. (Note: in both previous cases, the exercises were included in the first practice video; I had already included manual annotations of the players that did not wear sensors a couple of weeks ago).

- I manually tracked the players that did not wear the sensor for the 2nd video. Besides, I also tracked one of the coaches, as he participated in an exercise. I also synchronized ball annotations with the current tracking in order to automatize the separation of plays.

- I extracted the 16 feature vectors of the exercise going on at the left side of the court in the second video, which is a post-up situation (two interior players with the tracked coach passing the ball). The players try to follow a pattern, but this is not always being used. Half of the repetitions are being played on the right side and half on the left side.

- I extracted the 15 feature vectors of the exercise going on at the right side of the court in the second video, which is an offensive strategy to overcome full-court pressure: 3 offensive players and 3 defensive players. The players only want to overcome press, so no patterns are present.

- Finally, I merged all feature vectors in a matrix, and I applied PCA to see if clusters could be distinguished. More concretely, I plotted onto the 3 first principal components.

Week 13: April 24th - April 28th 2017

- I included new features in the feature vector, so instead of a 1x24 vector, I ended up with a 1x51.
- I implemented K-means over the 3D projection of my dataset in order to find out clusters; besides, I created a function to have an accuracy estimation. Nevertheless, this approach was discarded.
- Using the *Classification Learner App* of Matlab, I managed to perform a 10-fold cross validation and obtain 100% accuracy with Bagged Trees and 96.8% with a Linear SVM. Nonetheless, this 100% accuracy seemed suspicious (overfitting), as the dataset was limited in number of observations and classes; for this reason, I applied PCA once again until keeping the 95% variance of my dataset, and discard those correlated features. My data matrix turned from a 71 (observations) x 51 (features) into a 71x13. Applying a 10-fold cross validation, I obtained 91.7 % accuracy using a Linear SVM with a One-vs-One.
- At last but not least, I wrote the paper for the CVSP workshop, which took me 3 days of full-dedication.

Week 14: May 1st - May 5th 2017

- In order to have more consistent results, I decided to include a new class in the dataset, which was a *Fast break* play included in one of the videos the company sent me. To do so, once again, I had to manually track the players that did not wear the sensor over the whole sequence. Besides, the video and tracking data had to be synchronized.
- Following the same procedure as in the previous videos, I included the 17 fast break feature vectors in the dataset and obtained 92.3% accuracy performing the same 10-fold cross-validation (increase of 0.5% with another class). Besides, I decided to test how SVM performed using only 3 classes (excluding the unpredictable "press break" situation), and I obtained 98.6% accuracy.
- I thought that it could be a good idea to change the way players were selected. At that moment, I only extracted data (building feature vectors) from three of the involved players in a certain action by picking them manually (using a graphical input). My first approach was to check which were the players who moved faster during a play, but it was not as precise as desired: it usually picked properly two players, but the third one was difficult to guess.

Week 15: May 8th - May 12th 2017

- In order to automatically detect the 3 involved players in one action without a region map or a graphical input, I tested three I methods:

1. Computing the speed of the involved players in the action and checking which the top-3 was.
2. For each kind of play containing different repetitions in one same exercise, taking one action as a template (where, for instance, players 1, 2 and 3 were the involved ones). Then, for a new observation of the same play, computing the correlation between stored templates, and the rest of players. Maximum responses would indicate which players had a more similar movement.
3. In order not to require an action template, another approach would be computing the correlation among all players in one same action, in order to check how they move together; potentially, this technique would show high correlation values between those involved players.

Week 16: May 15th - May 19th 2017

- My goal was to detect the involved players in the action without showing a pattern as an example; this is, given trajectories of the 10 players on court, I have to find out which are the ones doing the exercise. In fact, all the exercises had less than 5 players (floppy offense has 3, pick and roll 4, press break 3, post-up 2 and fast break 2 as well), and the only part where I had actions with 5 involved players was a 3-minutes game, which did not contain any prepared play. The way to detect involved players is based, once again, on the correlation between different time signals.

- In this procedure, thresholding had to be applied to discard outliers. The problem statement was the following one: if there are two players standing still, the correlation between them will be really high, but they had not to be considered as players performing the action; in order to remove them, some thresholding was applied by taking standard deviation into account.

- Having obtained interesting results (but not as good as Pattern Correlation), I tried to optimize the threshold values, which should lead to better accuracy. However, these depend on the type of exercise and cannot be generalized into single values.

- Having validated that showing an example is the best way to detect the 3-involved players, I wanted to improve the accuracy of the *Pick and Roll Player-detection* (60%), so I tried different ways: computing the correlation only with the X or Y-trajectory, and doing the same with the absolute distance to the center of the court (as it is a kind of play that can be executed on both sides of the court). Although the accuracy of this particular case improved, the general accuracy decreased a little bit.

Week 17: May 22nd - May 27th 2017

- I started writing the report, as I have less than two weeks to deliver the thesis; I focused on the Related Work section, which is the one that takes more time.

- When I got the rejection for the CVSP conference, I got a little discouraged, so I shared the project with some basketball journalists, because I wanted to know if the project had any kind of sense for them. Their feedback was so positive that I got interviewed in a newspaper and a magazine. The sports newspaper is called *Mundo Deportivo* and it is the most important one in Catalunya; the magazine is called Zona 1-3-1, and is thought for a concrete target of *basketball fans*; the article will be published in July.

Bibliography

[1] 1D3A. Eric sports, 2016. http://www.ericsports.net/; Last Accessed: 2017-05-20.

[2] Mohamed Amer, Alan Fern, and Sinisa Todorovic. Probabilistic Event Logic For Interval-Based Event Detection - OSUPEL Dataset, 2011. http://blogs.oregonstate.edu/osupel/dataset/; Last Accessed: 2017-05-17.

[3] Jean-Yves Bouguet. Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm. *Intel Corporation*, 5(1-10):4, 2001.

[4] G. Bradski. Opencv library. *Dr. Dobb's Journal of Software Tools*, 2000.

[5] H.P.V. C. Method and means for recognizing complex patterns, December 18 1962. US Patent 3,069,654.

[6] John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1986.

[7] Dan Cervone, Alexander D Amour, Luke Bornn, and Kirk Goldsberry. POINTWISE : Predicting Points and Valuing Decisions in Real Time with NBA Optical Tracking Data. In *MIT Sloan, Sports Analytics Conference*, 2014.

[8] Evan Cheshire, Cibele Halasz, and Jose Krause Perin. Player Tracking and Analysis of Basketball Plays. 2015.

[9] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.

[10] Christophe De Vleeschouwer and Damien Delannay. APIDIS Basketball Dataset, 2011. http://sites.uclouvain.be/ispgroup/index.php/Softwares/APIDIS; Last Accessed: 2017-05-17.

[11] Frane Er, Brane Dežman, Goran Vu, Janez Perš, Matej Perše, and Matej Kristan. AN ANALYSIS OF BASKETBALL PLAYERS ' MOVEMENTS IN THE SLOVENIAN BASKETBALL LEAGUE PLAY-OFFS USING THE SAGIT TRACKING SYSTEM. *Physical Education and Sport*, 6(1):75–84, 2008.

[12] ESPN. NBA extends television deals with ESPN, TNT. `http://www.espn.com/nba/story/_/id/11652297/nba-extends-television-deals-espn-tnt`; Last Accessed: 2017-04-29.

[13] Panna Felsen and Patrick Lucey. " Body Shots ": Analyzing Shooting Styles in the NBA using Body Pose. In *MIT Sloan, Sports Analytics Conference*, 2017.

[14] Alexander Franks, Andrew Miller, Luke Bornn, and Kirk Goldsberry. Counterpoints : Advanced Defensive Metrics for NBA Basketball. In *MIT Sloan, Sports Analytics Conference*, 2015.

[15] Rikke Gade, Anders Jørgensen, and Thomas B. Moeslund. Occupancy Analysis of Sports Arenas Using Thermal Imaging. In *Proceedings of the International Conference on Computer Vision Theory and Applications*, pages 277– 283, 2012.

[16] Kirk Goldsberry and D Ph. CourtVision : New Visual and Spatial Analytics for the NBA. In *MIT Sloan, Sports Analytics Conference*, 2012.

[17] Kirk Goldsberry and Eric Weiss. The Dwight Effect : A New Ensemble of Interior Defense Analytics for the NBA. In *MIT Sloan, Sports Analytics Conference*, 2013.

[18] HoopsHype. NBA Salaries. `http://hoopshype.com/salaries/`; Last Accessed: 2017-04-29.

[19] Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multi-class support vector machines. *IEEE transactions on Neural Networks*, 13(2):415–425, 2002.

[20] Ian Jolliffe. *Principal component analysis.* Wiley Online Library, 2002.

[21] Patrick Lucey, Alina Bialkowski, Peter Carr, Yisong Yue, and Iain Matthews. " How to Get an Open Shot ": Analyzing Team Movement in Basketball using Tracking Data. In *MIT Sloan, Sports Analytics Conference*, 2014.

[22] Rajiv Maheswaran, Yu-han Chang, Aaron Henehan, and Samantha Danesis. Deconstructing the Rebound with Optical Tracking Data. In *MIT Sloan, Sports Analytics Conference*, 2012.

[23] Rajiv Maheswaran, Yu-han Chang, Jeff Su, Sheldon Kwok, Tal Levy, Adam Wexler, and Noel Hollingsworth. The Three Dimensions of Rebounding. In *MIT Sloan, Sports Analytics Conference*, 2014.

[24] Avery Mcintyre, Joel Brooks, John Guttag, Jenna Wiens, and Ann Arbor. Recognizing and Analyzing Ball Screen Defense in the NBA Learning to Classify Defensive Schemes. In *MIT Sloan, Sports Analytics Conference*, 2016.

[25] Andrew Miller, Luke Bornn, Ryan Adams, and Kirk Goldsberry. FACTORIZED POINT PROCESS INTENSITIES: A SPATIAL ANALYSIS OF PROFESSIONAL BASKETBALL. In *International Conference on Machine Learning*, pages 235–243, 2014.

[26] Andrew C Miller and Luke Bornn. Possession Sketches : Mapping NBA Strategies. In *MIT Sloan, Sports Analytics Conference*, 2017.

[27] NBN23. Technology to take Basketball into the future. `https://www.nbn23.com/en/`; Last Accessed: 2017-04-29.

[28] Visual Analysis of People (Aalborg University). Bounding box annotator, 2016. `https://bitbucket.org/aauvap/bounding-box-annotator`; Last Accessed: 2017-05-20.

[29] Matej Perse, Matej Kristan, Janez Perš, and Stanislav Kovacic. A Template-Based Multi-Player Action Recognition of the Basketball Game. In *CVBASE '06 - Proceedings of ECCV Workshop on Computer Vision*, pages 71–82, 2006.

[30] Vignesh Ramanathan, Jonathan Huang, Sami Abu-el haija, Alexander Gorban, Kevin Murphy, and Li Fei-fei. Detecting events and key actors in multi-person videos. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[31] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

[32] Second Spectrum. Our Work. `https://www.secondspectrum.com/ourwork.html`; Last Accessed: 2017-04-29.

[33] STATS. Basketball Data Feed — Basketball Player Tracking — SportVU. `https://www.stats.com/sportvu-basketball-media/`; Last Accessed: 2017-04-29.

[34] Chris Stauffer and W Eric L Grimson. Adaptive background mixture models for real-time tracking. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 2, pages 246–252. IEEE, 1999.

[35] Carl Vondrick, Donald Patterson, and Deva Ramanan. Efficiently scaling up crowdsourced video annotation. *International Journal of Computer Vision*, pages 1–21. 10.1007/s11263-012-0564-1.

[36] Kuan-chieh Wang and Richard Zemel. Classifying NBA Offensive Plays Using Neural Networks. In *MIT Sloan, Sports Analytics Conference*, pages 1–9, 2016.

[37] Jenna Wiens, Guha Balakrishnan, Joel Brooks, and John Guttag. To Crash or Not To Crash : A quantitative look at the relationship between offensive rebounding and transition defense in the NBA. In *MIT Sloan, Sports Analytics Conference*, 2013.

[38] Wilson. Wilson X Connected Basketball - Smart Sensor Basketball & App. `http://www.wilson.com/en-us/explore/basketball/wx`; Last Accessed: 2017-04-29.

[39] Tex Winter. The Encyclopedia of the Triangle Offense, 2006. `http://www.championshipproductions.com/cgi-bin/champ/p/Basketball/The-Encyclopedia-of-the-Triangle-Offense{_}BD-02610.html`; Last Accessed: 2017-04-29.