



APIDIS

Autonomous Production of Images based on Distributed and Intelligent Sensing

STREP Project, 1st FP7-216023

D5.2 Event detection algorithms

Due date of deliverable: 31-12-2009

Actual submission date: 18-03-2010

Start date of project: 1st January, 2008

Duration: 36 months

Lead contractor for this deliverable: UCL

[Revision Final v1]

D5.2	Deliverable Name
Project Acronym :	APIDIS
Contract No :	FP7-216023
Due Date :	31/12/10
Reply To:	Damien Delannay Damien.Delannay@uclouvain.be
Actual date of delivery	18-03-2010

Executive Summary

This document discusses the high level interpretation mechanisms that have been considered by APIDIS to personalize the summarization of basket ball games. Both the recognition of players and events has been studied. The deliverable demonstrates that these two types of recognition can be reached with more than 90% of accuracy, thereby making the concept of automatic and personalized summarization realistic.

Deliverable Identification Sheet

Project ref. no.	FP7-216023
Project acronym	APIDIS
Project full title	FP7-216023
Security (distribution level)	PU (Public)
Contractual date of delivery	Month 24, Month 12, 2009
Actual date of delivery	Month 27, Month 03, 2010
Deliverable number	D5.2
Deliverable name	Events detection
Type	Report
Status & version	Submitted
Number of pages	41
WP / Task responsible	WP5 / UCL
Other contributors	ACIC
Author(s)	Damien Delannay, Christophe De Vleeschouwer, Christophe Parisot
EC Project Officer	Marzio Morina
Abstract	This document presents the high level interpretation mechanisms considered to personalize the summarization of basket ball games
Keywords	Events detection
Sent to peer reviewer	ACIC
Peer review completed	Yes
Circulated to partners	Yes
Read by partners	Yes
Mgt. Board approval	Pending

Table of contents

EXECUTIVE SUMMARY.....	2
1 INTRODUCTION.....	5
2 PLAYERS RECOGNITION.....	6
2.1 SHIRT COLOR & TEAM RECOGNITION.....	7
2.1.1 Silhouette extraction.....	7
2.1.2 Color characterization.....	7
A. Single dominant color.	7
B. Multiple dominant colors.....	11
2.1.3 Exploiting multi-cameras views.....	13
2.2 PLAYER IDENTIFICATION BY MEANS OF DIGIT RECOGNITION.....	15
2.2.1 Selection of candidate digit regions.....	16
2.2.2 Number recognition.....	17
A. Thumbnail normalization and projection.....	17
B. Method based on Local Binary Pattens (LBP).....	18
C. Method based on Matching Pursuit expansion.....	20
D. Preliminary evaluation of the digit recognition methods.....	22
2.3 PLAYERS RECOGNITION CONCLUSION.....	24
3 CLOCK-EVENTS CLASSIFICATION.....	26
3.1 TREE-BASED ORGANIZATION OF BASKETBALL ACTIONS.....	26
3.2 EVENTS CHARACTERISTICS.....	29
3.3 DETECTION MODULES	30
A. Throw-in and throw detector.....	30
B. Free throw and time-out detector.....	32
C. Violations and fouls.....	33
3.4 RESULTS.....	34
3.5 EVENTS CLASSIFICATION CONCLUSION.....	35
4 CONCLUSION.....	36
5 REFERENCES.....	37
ANNEX 1: FEATURES PER EVENT.....	38
A. Field goal.....	38
B. Violation.....	38
C. Foul.....	38
D. Ball out-of-bounds.....	39
E. Free-Throw	39
F. Throw-in (Ball back to court).....	40
G. Throw & Rebound.....	40
H. Lost ball.....	41
I. Substitution.....	41
J. Time-out.....	41

1 Introduction

This deliverable introduces the high level interpretation mechanisms that have been considered by APIDIS to personalize the summarization of basket ball games.

Two main objectives are considered:

- The first one attempts to recognize the individuals (players/referees) moving along the game on the ground plane of the basketball court. Players detection and tracking have been considered in Deliverable 7.1 and 4.2 respectively. Here, we are more interested in players recognition, through recognition of teams and digits on the players shirts. By getting the identity and position of the players during each action, and assuming the ball position is known, we become able to identify the key players involved in each action, e.g. the one that shoots. This allows APIDIS to generate a summary that focuses on the actions involving a preferred player.
- The second one aims at recognizing events of interest along the game. As explained in D3.2, we are mainly interested in clock-events, i.e. in events that have caused a stop, a start, or a reset of the 24'' clock. Those events include most actions that take place along the match (shot, rebound, violation, interception, ...). Hence, their recognition is sufficient to personalize the summary according to the semantics of the game. In practice, this is done by turning the clock signals and the metrics defining the position of the ball and players into a sequence of action labels. For this purpose, we have designed a tree-based classifier that relies both on deterministic and stochastic inputs to take a decision. Typically, the deterministic inputs are provided by the monitoring of the reliable 24'' clock signal, while the stochastic metrics result from ad-hoc detectors, inferring the occurrence of a specific action, e.g. a free throw, based on the appearance of specific ball and players positions patterns.

Finally, the deliverable demonstrates that those two objectives can be reached with more than 90% of accuracy, thereby making the concept of automatic and personalized summarization realistic.

2 Players Recognition

The inputs of this recognition process are the bounding box generated by the player detection module described in deliverable 4.2. The outputs are for each bounding box the player team and number, which are useful for the automatic summary generation, but are also used to increase the robustness of the tracking module.

On each detected individual on the basketball court (cfr player detection in deliverables 4.2 and 7.1), we will conduct an analysis to provide an accurate characterization of its appearance. In the most favorable situations, this process will provide accurate team identification and player identity (number). In general, the appearance characterization will inform about likelihoods to belong to each of the teams and to correspond to a specific player.

Note that the analysis described in this section is performed on each time instant independently. The output of this process can serve two different purposes: On one hand, it provides a fast characterization of the detected players that can be used to select optimal rendering parameters in a live (low-latency) production process. On the other hand, these appearance signatures can be exploited by the player tracking process as discriminant features to help making the correct associations between independent players detections. Eventually, in the light of the tracking results, appearance characterization of individual detections can be gathered to make reliable decision about player identity (number) and/or team belonging.

Player recognition consists thus in two different tasks : team recognition and player number recognition, as illustrated below in Figure 1.

For the number recognition path, the first step consists in segmenting the number on the player shirt from a thumbnail of the player. The team recognition process is directly applied on the player thumbnail.

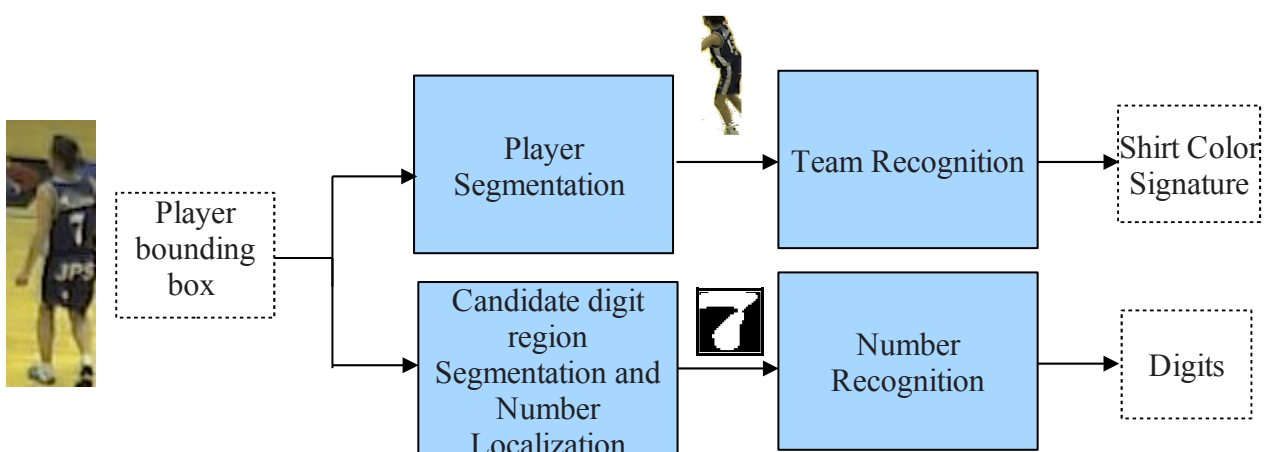


Figure 1: Player recognition consists in team and number recognition

2.1 Shirt color & team recognition

An obvious appearance characteristic for non rigid objects is the color composition. In a team sport event, it is even more the case since each team needs to wear shirts that are easily distinguishable.

In our application, color characterization is a two step process. The first step consists in computing a feature vector composed of discriminant color measures. The second step tries to make a decision or just provides a likelihood measure to belong to each of the teams.

2.1.1 Silhouette extraction

At each player position, a rectangular region is considered, based on the average player height and width dimensions. Within this rectangular region only the pixel belonging to the foreground detection mask are considered. The computation of the foreground mask is the initial step of the player detection algorithm and was already described in another workpackage.

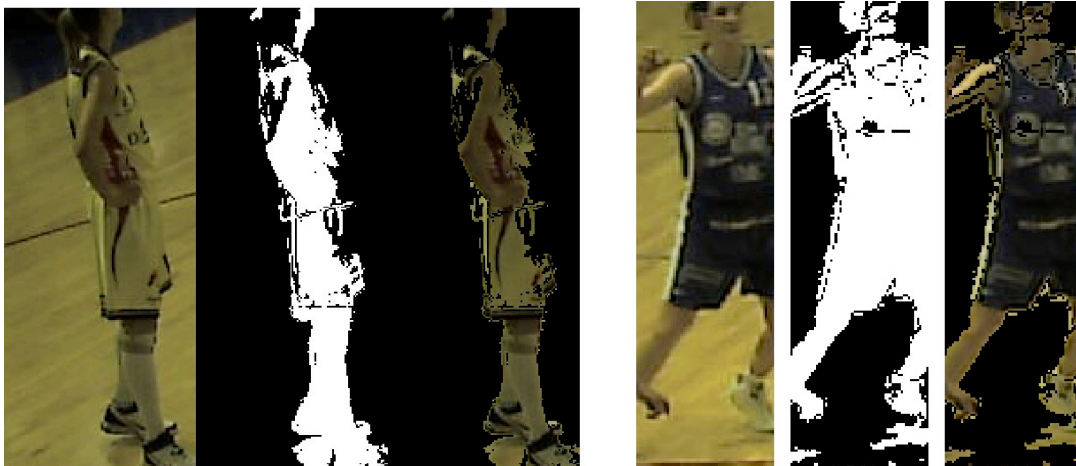


Figure 2: Player silhouette extraction

Since the head and feet of the player usually exhibit limited discriminant color information, those regions are discarded in the color measurement.

2.1.2 Color characterization

A. Single dominant color.

If the shirts have a single dominant color, the characterization of the color can be reduced to a unique color (expressed as a three component vector in an appropriate color space).

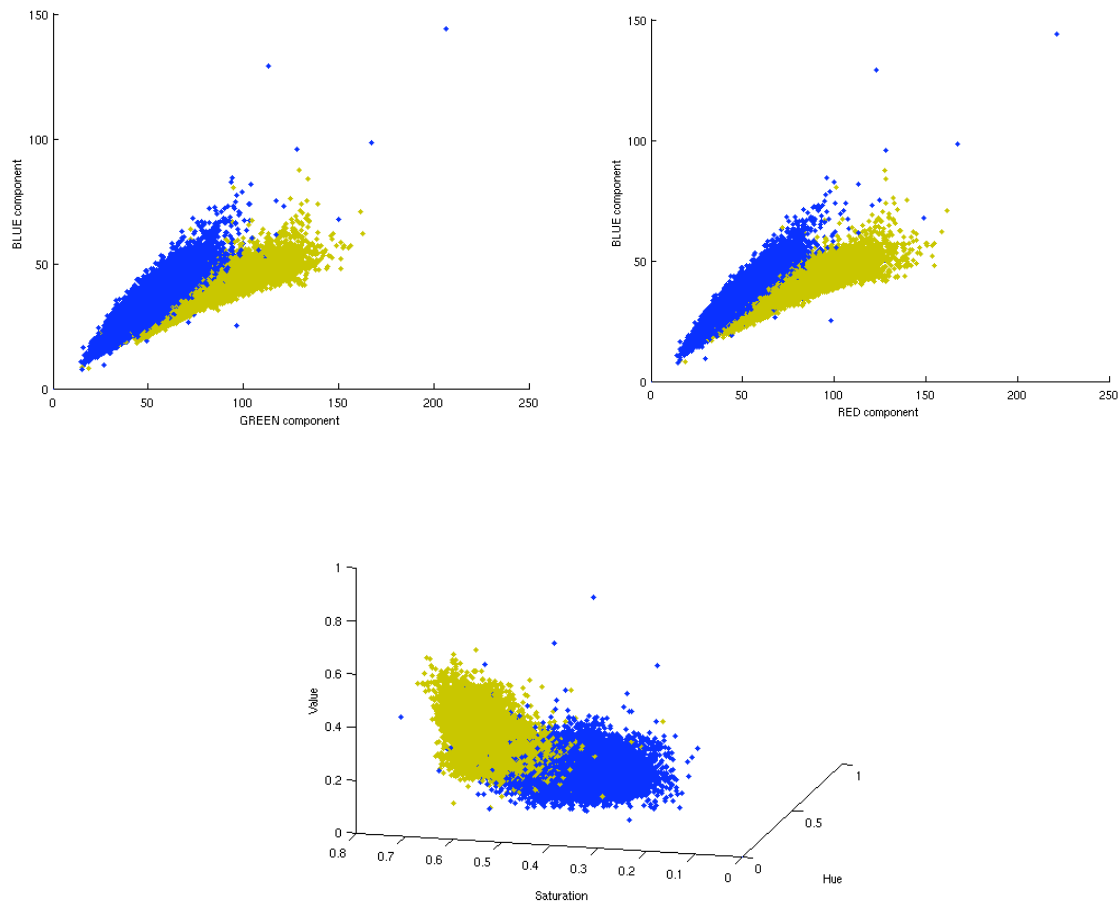


Figure 3: RGB and HSV average value for each player silhouette for the yellow and blue teams

In Figure 3 above, we plotted the RGB & HSV average value for each player silhouette within our groundtruth (~28000 detections). Since the color value results from the averaging over all the pixels of the silhouette, the presence of multiple dominant colors cannot be observed. This approach is sufficient to distinguish between teams if only one color is dominant. The color difference between two players can be computed as a weighted euclidean distance. This approach is however highly sensitive to noise in the observation.

When a priori information is available about the samples distribution, more specific metrics can be designed. In our scenario, we know that the appearance of players (almost) only differs by the color of the shirts; the samples can therefore be classified into two subsets. We consider that skin or hair color differences cannot be observed reliably. An appropriate metric would measure only the discriminant part of the color information between the two teams.

In order to design this metric, a learning step is required. A simple SVM could be implemented but it requires a training set with known labels. In order to have an automatic

process, the learning process should instead perform autonomous clustering of the color features.

One way to achieve automatic clustering of the data samples is to perform a principal component analysis (PCA) on the distribution. In the following figure we projected the samples averaged color components on the principal direction of the distribution. The achieved separation is better in the HSV color domain (optimal threshold yields approx. 4% mismatch between classes). This is due to the fact that the distribution of each class in the RGB domain is far from being isotropic in the color components.

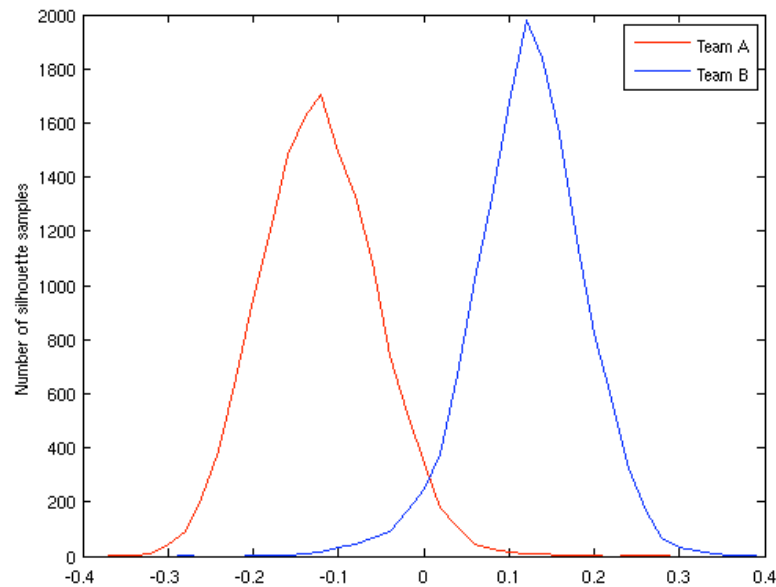


Figure 4: Projection of the HSV feature vector on the principal axis of the distribution

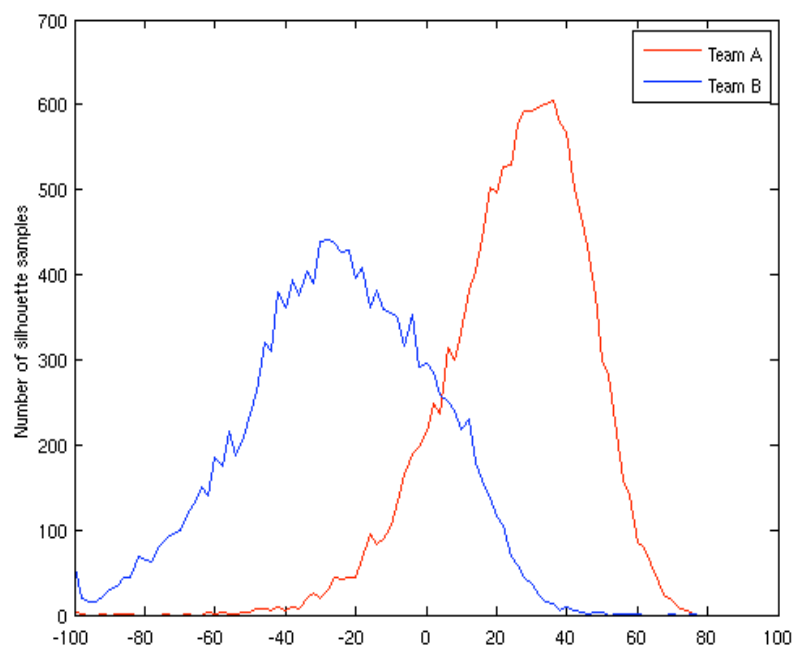


Figure 5: Projection of the RGB feature vector on the principal axis of the distribution

An intuitive metric can be designed for the RGB color domain. Because team colors are blue and yellow in the first APIDIS dataset, a metric computing the ratio of blue component with respect to the sum of red and green component provides good separation. This is illustrated in Figure 6.

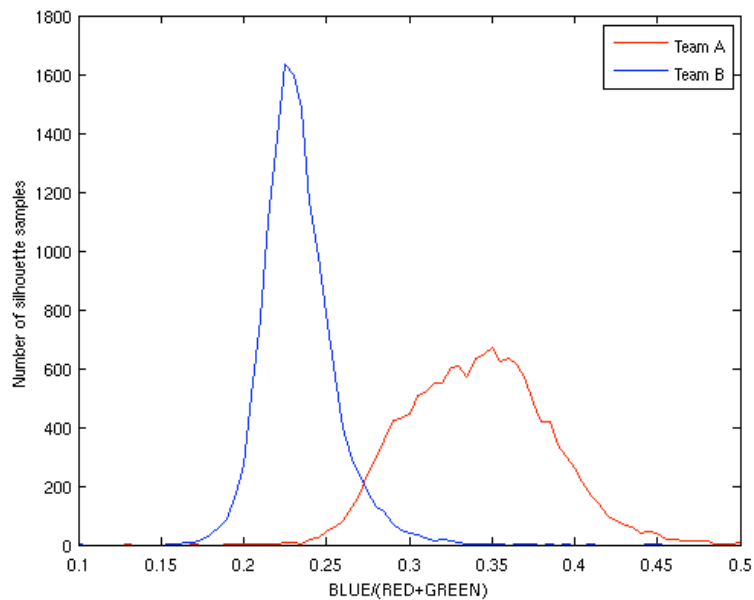


Figure 6: Distribution of $BLUE / (RED + GREEN)$ metric for each team

Figure 7 below illustrates the evolution of the $B/(R+G)$ metric for 5 different players along the game.

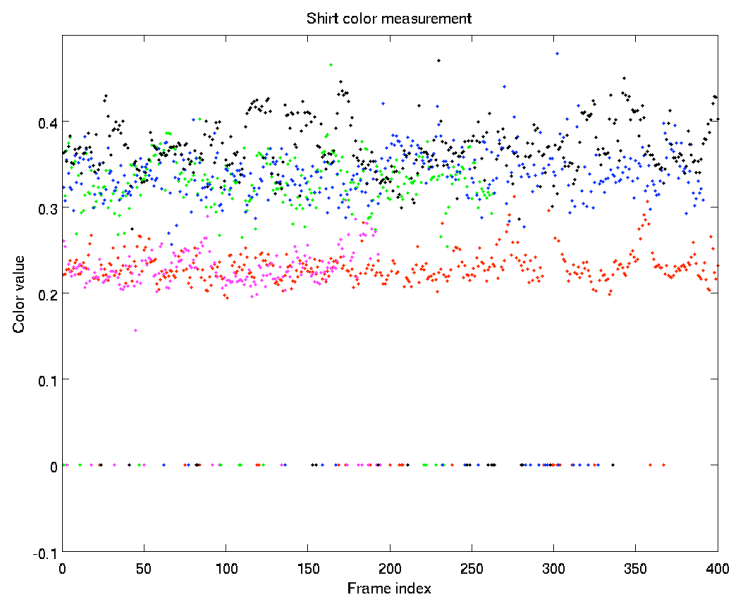


Figure 7: Evolution of the $B/(R+G)$ color metric for 5 different players along the second quarter of the game

B. Multiple dominant colors.

A more general approach considers that the shirts can have several dominant colors, the color feature vector needs to keep track of the different color components. The module is then based on the study of the histogram computed from a player silhouette. The algorithm proceeds exactly the same way as in the case of a single dominant color, except that it works on a histogram instead of the 3 averaged color component values. The process is illustrated on the same player silhouette dataset (first APIDIS dataset).

The histogram is generated by counting the number of pixels in the silhouette (Y axis) for each 3-d interval of color components (bin, X axis). The figure below represents the average histograms (8000 bins) for both teams simultaneously (Figure 8) and for each team independently (Figures 9 and 10).

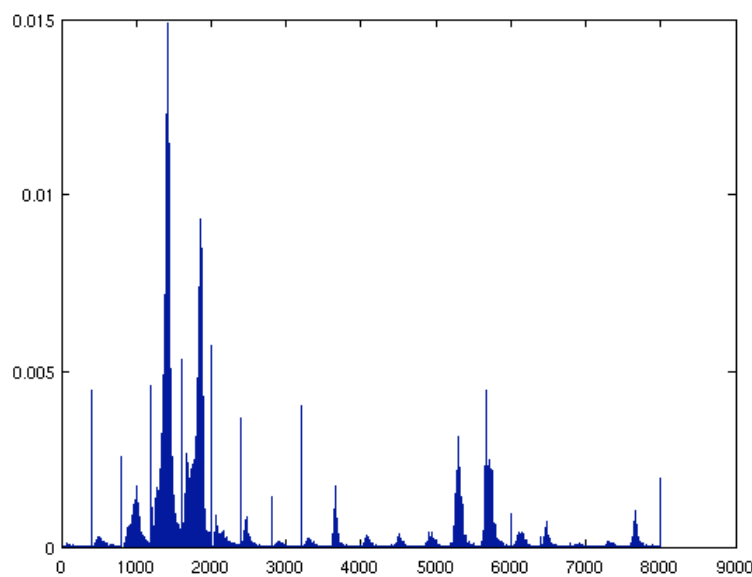


Figure 8: Players (both team 1 & 2) HSV average histogram (8000 bins)

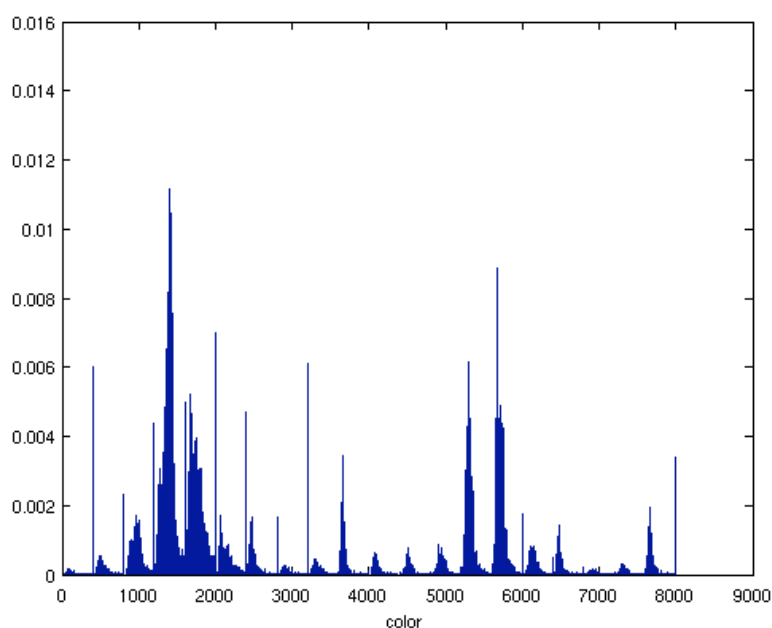


Figure 9: Team 1 HSV average histogram (8000 bins)

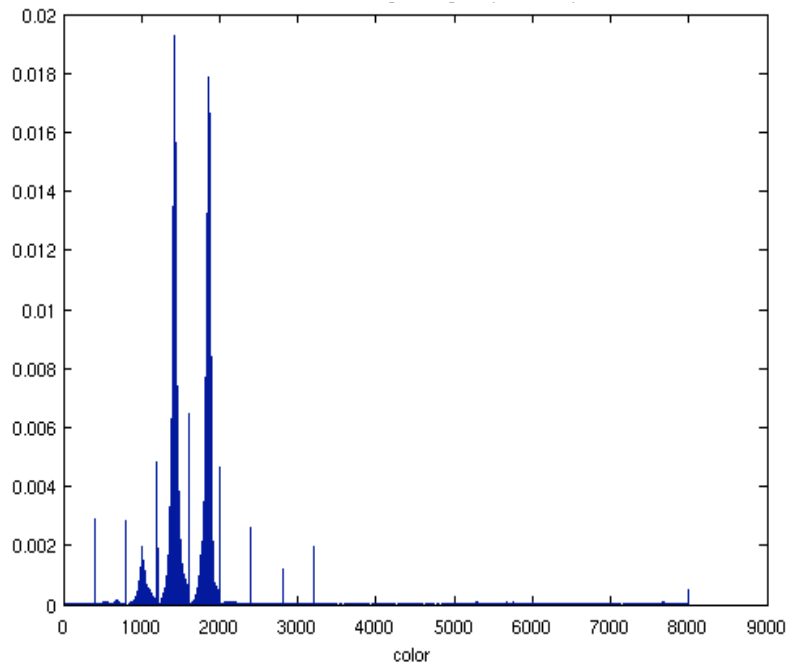


Figure 10: Team 2 HSV average histogram (8000 bins)

The Figures 11 and 12 below illustrate the separation that can be achieved between the teams after projection on the principal axis of the histogram distribution. In this case both RGB and HSV color domain yield good results (4.5% mis-classification).

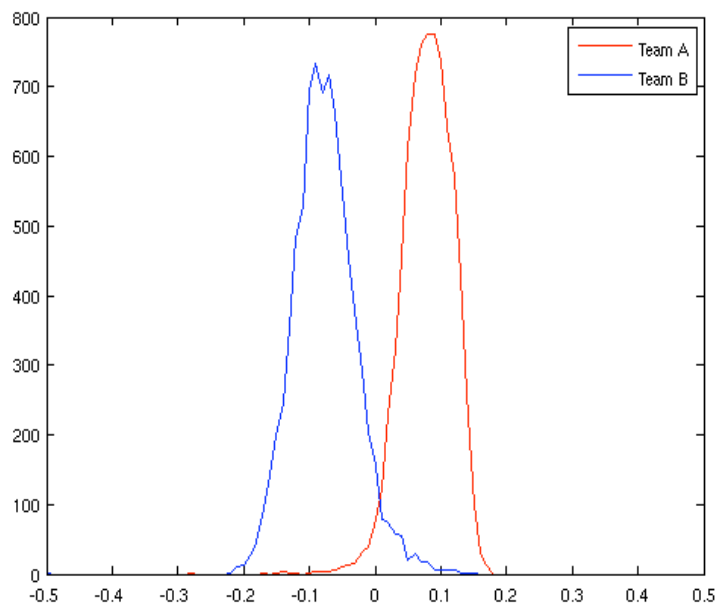


Figure 11: Projection of the 1000 bin HSV histogram feature vector on the principal axis of the distribution

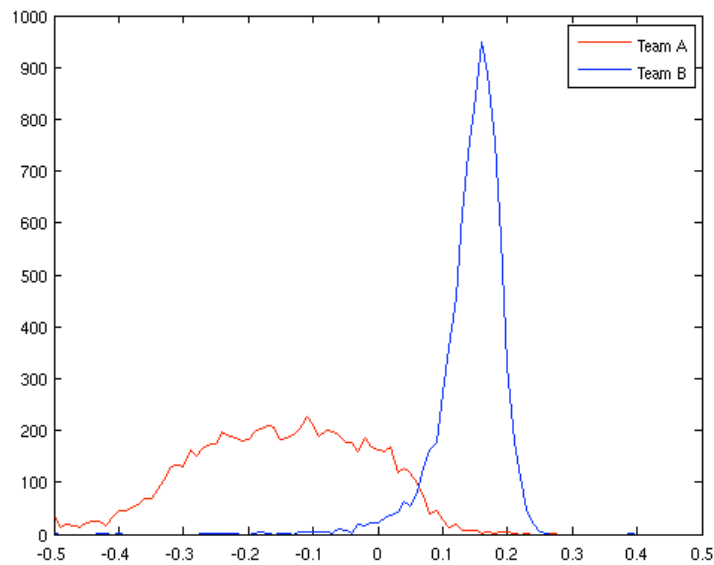


Figure 12: Projection of a 1000 bin RGB histogram feature vector on the principal axis of the distribution

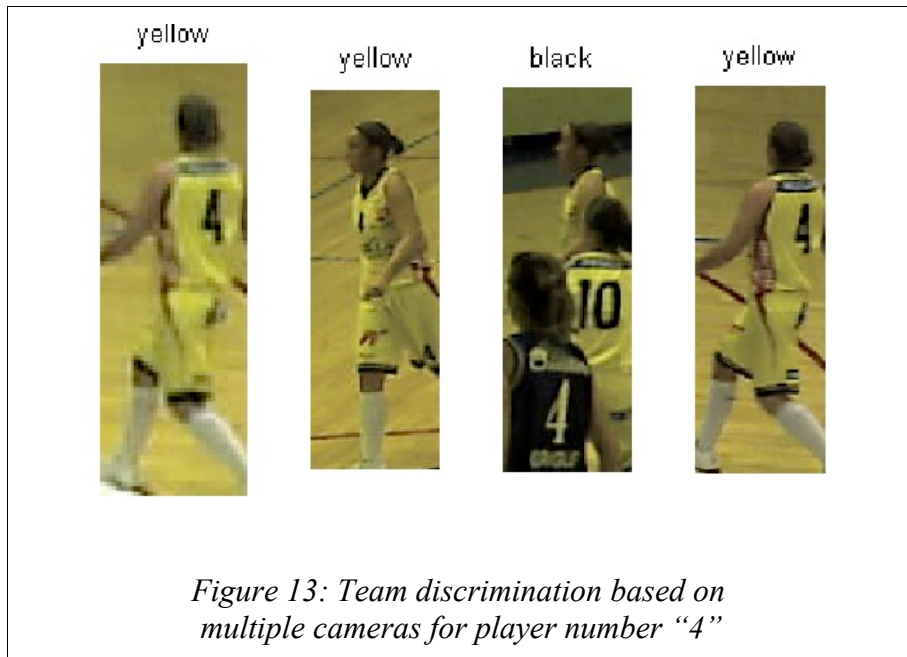
Another approach could consist in identifying the discriminant histogram peaks. An automatic identification of the peak associated to each team can be achieved using an iterative peak selection technique. The method must select peaks which clearly (not too sensitive to threshold value) split the training set into two subsets which have an important number of representative. A peak corresponding to a color which belongs to both of the teams will not exhibit such a property. This latter method has not been implemented yet.

In order to be less sensitive to the variation of illumination across different position on the court, a normalization of the luminance based on each camera background could be performed.

2.1.3 Exploiting multi-cameras views

Each player silhouette captured by a camera is subject to noise that can take different forms. The biggest source of error is related to occlusions between players of different teams. Silhouettes which exhibit occlusions that are detected by the player detection algorithm are discarded from the shirt color analysis process. However some undetected occlusions interfere with the shirt color estimation. Another source of error is due to errors in the foreground mask estimation. Due to shadow on the ground, part of the background is included into the silhouette.

In order to increase the system robustness, multiple views of the same player at the same moment are exploited, as illustrated below in Figure 13 for player “4”.



The color features are first measured for each available angle of view. Metrics are then combined together to yield a more robust measure. Below, Figure 14 below illustrates the improvement brought by this approach. Classification errors drop down to 1.5% on our dataset.

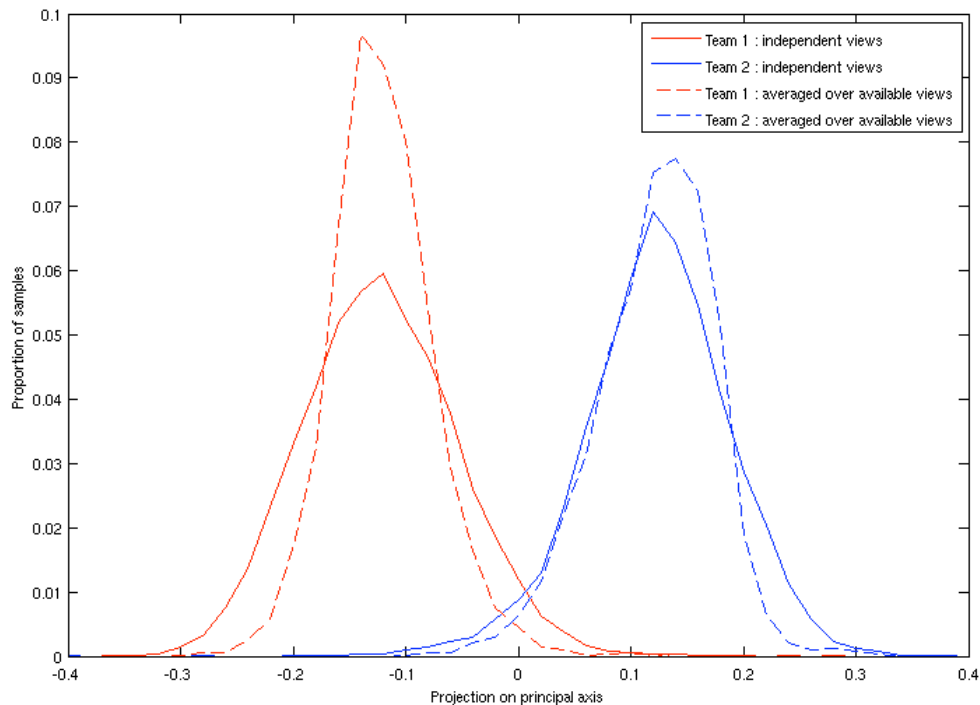
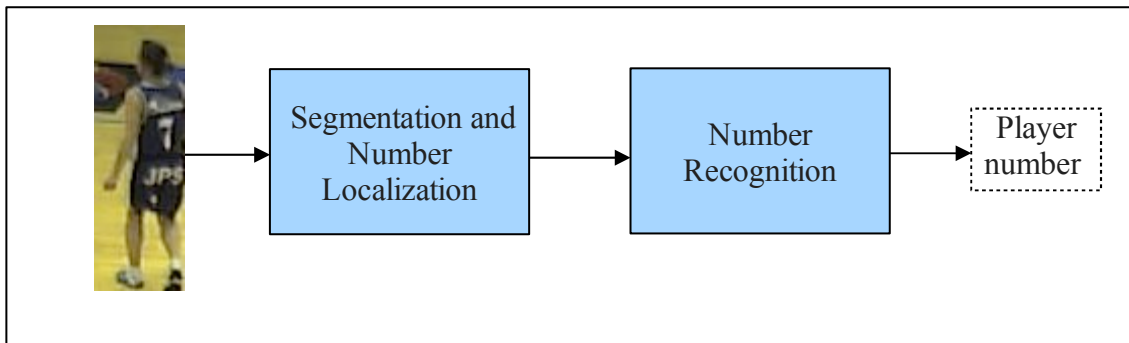


Figure 14: Exploiting multiple views to recognize the player team

2.2 *Player identification by means of digit recognition*

Player identification can be achieved by recognizing the digits on the shirts of the players when they are visible. This task consists in two steps. First, candidate digits are localized in the player silhouette. A second process tries to identify digits in those candidate regions. Since the player number can only be read when it faces towards the camera, this process must be combined with a temporal tracking to increase confidence and propagate the identity of a player along its trajectory. Tracking is addressed in workpackage 4.



The number recognition method developed during the second year of the project has been presented in details in a publication [1]. In this section, we present an overview of this initial approach, point out the limitations and describe two new approaches that were not part of the publication.

Figure 15 below depicts the successive steps that compose the recognition algorithm. The first half of the process which consists in the localization of candidate digit regions is common to all approaches. It is described in Section 2.2.1 below. Section 2.2.2 presents the three different approaches that we developed for digit recognition. They use different normalization steps and compute different features to characterize the candidate regions. All methods implement a SVM to achieve an optimal classification of the digit samples.

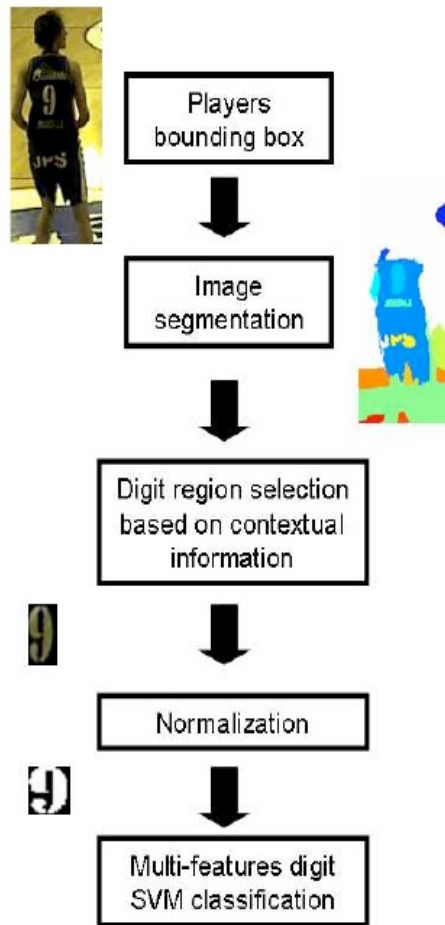


Figure 15: Recognition of digits printed on players' shirts through segmentation, selection, and classification of regions that are likely to represent digits.

2.2.1 Selection of candidate digit regions

This step consists in selecting the regions among the ones defined within the players silhouette that might correspond to digits. The image thumbnail where a player was detected is first segmented in uniform color regions. The regions provided by mean-shift segmentation are sorted out based on contextual features, e.g. relative size and position, to select the ones that, individually or grouped, are likely to correspond to a digit.

An issue with the actual approach is the following: The color segmentation step is characterized by a high complexity, mainly due to the mean-shift implementation. In an exploitation framework, this step should have a lower complexity. A reduction of the image segmentation complexity has not yet been studied in the context of this project and will be examined in future work. A solution could be to exploit the knowledge about the shirt color (cfr. previous section) to identify color clusters and speed up the segmentation.

2.2.2 Number recognition

Several approaches have been considered to design the recognition module. This module is in charge of deciding whether a candidate region is a digit or not, and of recognizing the value of the digit when the region appears to be relevant.

Note that all methods rely on a supervised learning step. This means that we have at our disposal a large set of digit samples with associated labels. If the training set is large and diversified, the method should be robust to the variety of digit fonts that we can encounter.

In the following, we refer to the concept of *Classification accuracy* defined as $\frac{TP+TN}{TP+TN+FP+FN}$ and *Balanced Classification Rate* (BCR) defined as $0.5 \cdot (\frac{TP}{TP+FN} + \frac{TN}{TN+FP})$ where T stands for True, F for False, P for Positive and N for Negative.

A. Thumbnail normalization and projection

This approach has been described in the ICDSC09 publication [1], and is depicted in Figure 15. Given a candidate region, it involves two steps. The first one normalizes the region. The main principal axis is vertically aligned, and the candidate region is stretched both vertically and horizontally, to use the whole range of a 16x16 pixels square. The second step then consists in extracting a set of features (1st and 2nd order moments) from the the horizontal and vertical projections of the binary region, to feed a SVM classifier, in charge of the digit recognition.

However, the experiments conducted in [1] have revealed that this approach suffers from one major limitation: the alignment associated to the normalization step appears to lack robustness. The weakness come from the difficulty to identify the main principal directions for some of the digit patterns (some of them have a kind of isotropic distribution).

To circumvent the normalization issue encountered by the initial OCR-based method, we have investigated two solutions that rely on features that are invariant to rotation. Hence, for those methods, only a normalization in size is required, without a rotation.

Figure 16 below illustrates the results of the method by plotting the number detected during the first quarter of a game for player “9”. We observe that while the number 9 is mostly correctly detected, both invalid and null detections still occur.

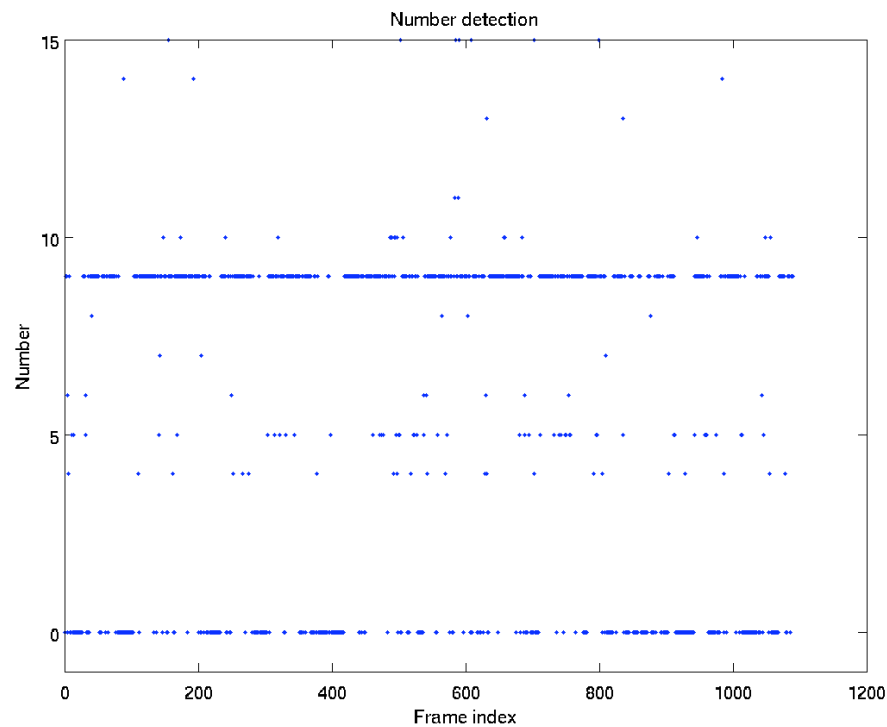


Figure 16: Number detected for player “9” during the first quarter of a game

B. Method based on Local Binary Pattens (LBP)

The first approach is based on the concept of ‘Local Binary Pattern’. The LBP operator [2] is one of the best performing texture descriptors and it has been widely used in various applications. It has proven to be highly discriminative and its key advantages, namely its invariance to monotonic gray level changes and computational efficiency, make it suitable for demanding image analysis tasks [3].

The LBP operator was originally designed for texture description. The operator did assign a label to every pixel of an image by thresholding the 3x3-neighborhood of each pixel with the center pixel value and considering the result as a binary number. Then the histogram of the labels could be used as a texture descriptor.

In our work, we have extended the notion of LBP to handle and characterize binary shapes. This has been done by:

- Defining the local neighborhood of a pixel as a set of sampling points evenly spaced on a circle centered at the pixel to be labeled. This allows any radius r and number n of sampling points. Bilinear interpolation is used when a sampling point does not fall in the center of a pixel.
- Labeling each pixel by the number of transitions observed along the circle centered in the pixel to be labeled. Obviously, the number of labels is equal to $N/2 + 1$.

We add one bit of information to the label, to indicate whether the pixel of interest is black or white.

The histogram of the labels is then representative of the binary shape at hand. As an example, Figure 17 and 18 presents the histograms obtained for three classes of digits, with different (r,n) parameters. We observe that the histograms are different for distinct digits, and that the discriminant power of bins is hardly predictable. Therefore, we have used all bins of the histogram as input features to our SVM classification engine.

Table 1 presents the balanced classification rate obtained with different parameters n and r . The experimental protocol and the training set considered in this experiment are further described below. Here, we just observe that a large circle and a large number of points increase the performance of the classifier.

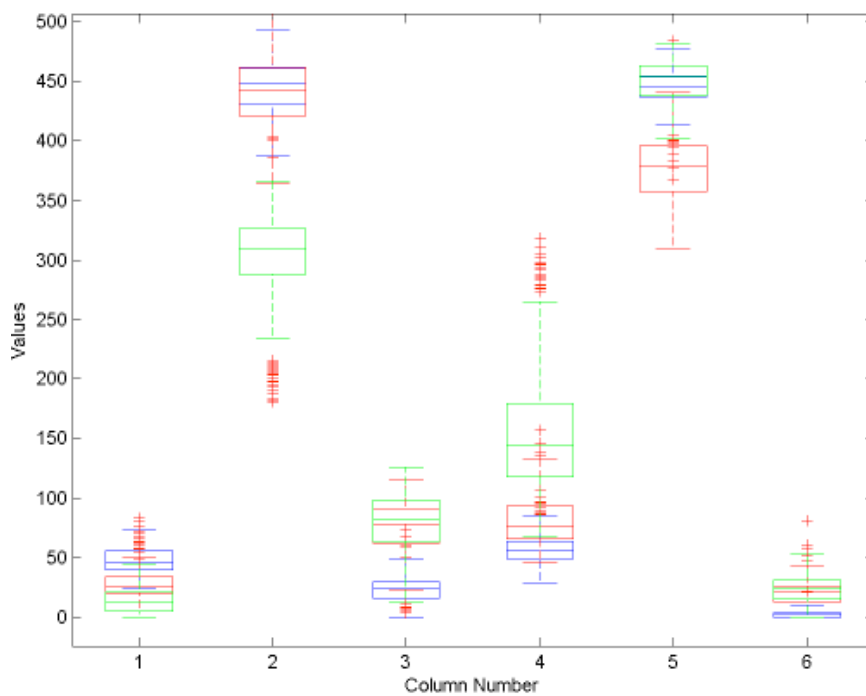


Figure 17: LBP histogram obtained with $r=8$ and $n=4$, for three different classes of digit. Green: class '7'; Red: class '9'; Blue: class '4'

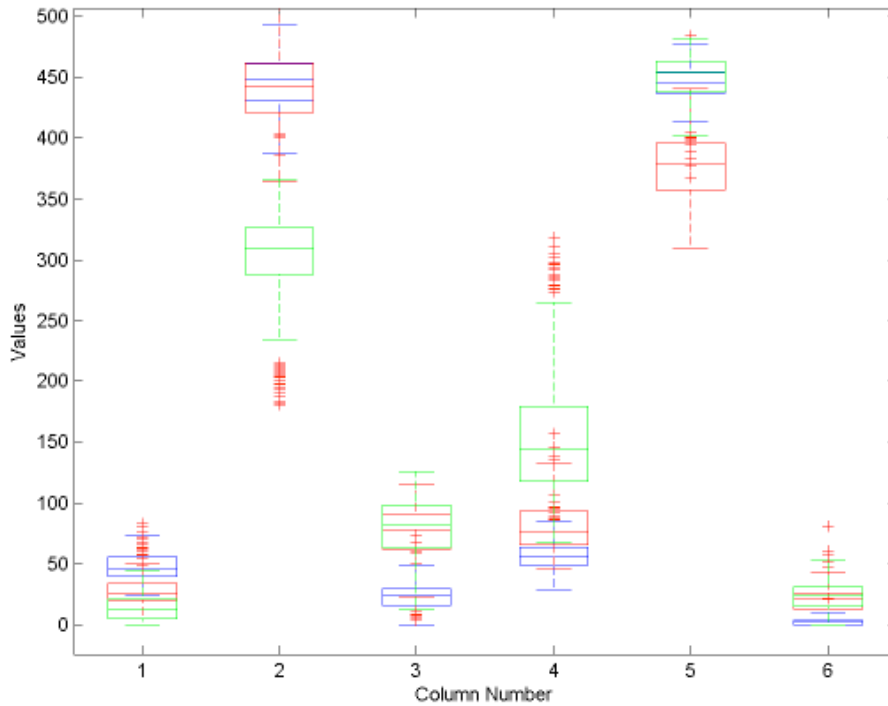


Figure 18: LBP histogram obtained with $r=8$ and $n=8$, for three different classes of digit. Green: class '7'; Red: class '9'; Blue: class '4'

Rayon	4 points	8 points
4	90,89	92,68
5	93,39	96,07
6	91,79	95,54
7	87,86	96,43
8	91,34	96,79

Table 1: LBP balanced classification rate as a function of parameters (r,n)

C. Method based on Matching Pursuit expansion

The second approach relies on a Matching Pursuit (MP) expansion process [4]. It uses an iterative greedy approach to approximate the candidate digit region by a weighted sum of a limited number of atoms (6 atoms have been considered in our experiments), selected in a large and redundant dictionary of functions. Those functions are characterized by an anisotropic ellipsoidal support, and are typically defined based on the separable product of two Gaussian functions. They have been chosen to approximate the straight line segments composing the digit. An example of digit approximation is depicted in Figure 19. The expansion can be computed efficiently by using gradient descent mechanisms at each iteration[5]. A set of discriminant features is then extracted to

characterize the relative position of those atoms around their gravity center. Those features are presented in Figure 20. They are invariant to rotation and scaling, and correspond for each ellipse to:

- The distance d of the ellipse center to the center of gravity g of the digit. This distance is normalized by the largest side of the region bounding box
- The angle α_2 between the principal axis of the ellipse, and the line connecting the center of the ellipse to the gravity center g

In addition, for a pair of ellipses, we define:

- The angle α_1 between the straight lines connecting each ellipse center to the gravity center g

Those features have to be ordered to feed the subsequent classifier. In practice, the distance d and the angle α_2 are ordered in decreasing order of atom magnitude, while the angles α_1 are computed between pairs of consecutive atoms ordered in decreasing order of magnitude.

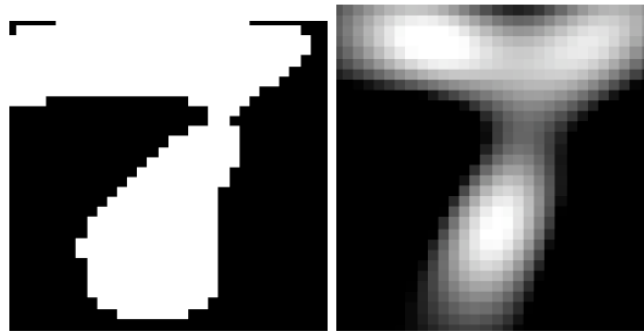


Figure 19: The candidate digit region is approximated by 3 atoms selected in a dictionary of anisotropic ellipses

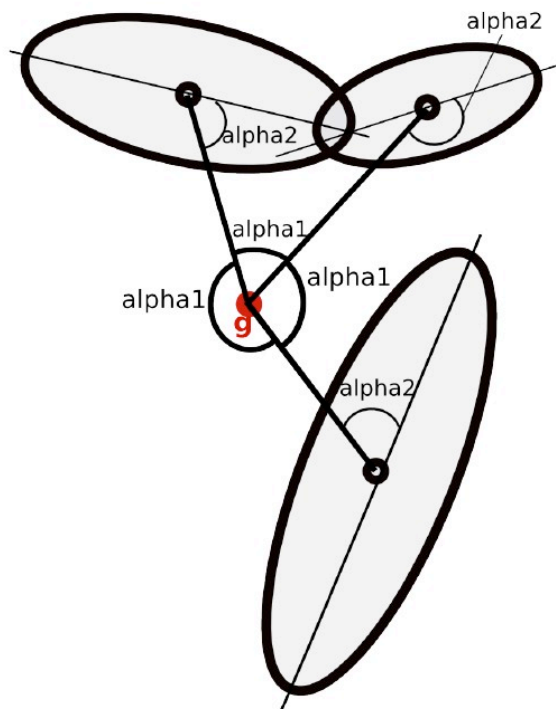


Figure 20: Three kinds of features are extracted from the set of atoms that approximates the candidate region

D. Preliminary evaluation of the digit recognition methods

This section presents the initial classification results obtained with each one of the three proposed methods. To assess and compare the classification performance of the methods, we have extracted manually a number of samples that are representative of the digits printed on players' shirts. This has been done according to the process depicted in Figure 21. Starting from regions containing numbers manually extracted from the second quarter dataset sequence, the following steps were applied:

- Step 1. RGB to grayscale conversion
- Step 2. Otsu [6] thresholding to reduce the gray level image to a binary image
- Step 3. Automatic discard of small blobs which do not correspond to digits
- Step 4. PCA based normalization of orientation

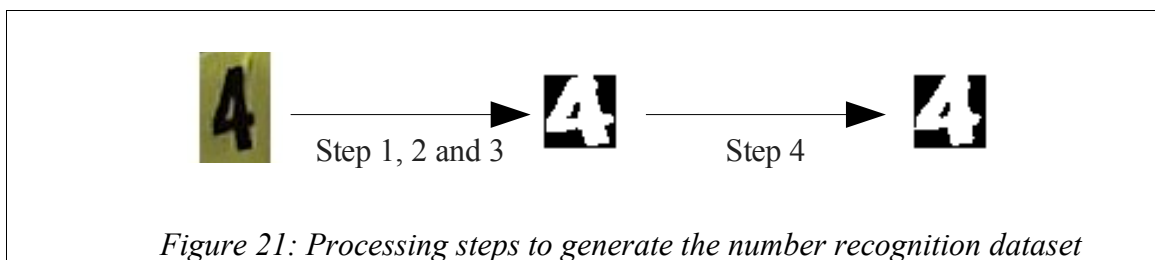


Figure 21: Processing steps to generate the number recognition dataset

A bin class has also been created to characterize samples that are not digits.

Examples of samples for the class ‘7’ and ‘14’ are presented in Figures 22 and 23, respectively. It is worth mentioning here that the dataset considered in this experiment is a normalized and rather clean dataset. In particular, samples for which the digit could not be recognized visually have been removed from the corresponding training class. The purpose is thus to compare the digit recognition capabilities of each method in ideal conditions. Real-life testing and assessment is however needed to draw definitive conclusions about each method. Such an experiment requires extensive manual annotation of data and is beyond the scope of our preliminary validation of the methods. It should however be considered before actual deployment of the APIDIS solution in a real-life environment.

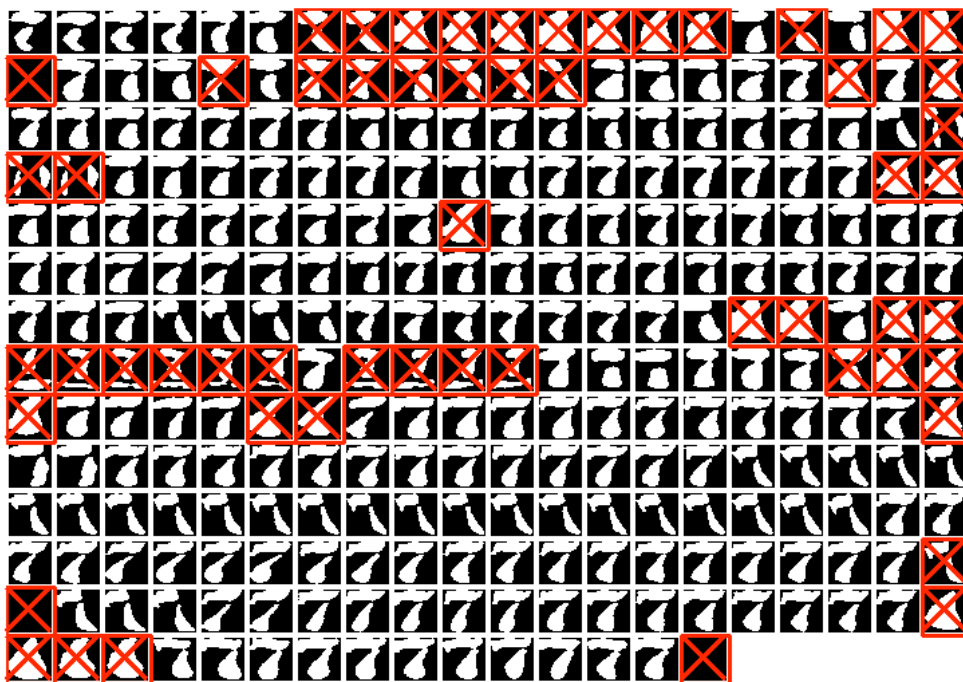


Figure 22: Class ‘7’ of the dataset that has been manually generated for assessment and comparison of the digit recognition methods. Red crosses correspond to samples that have been removed from the training set, because they could not be recognized visually.

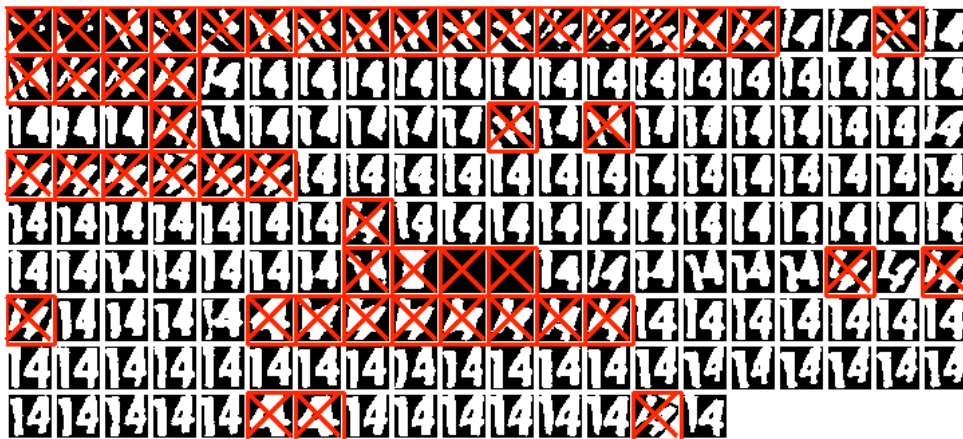


Figure 23: Class ‘14’ of the dataset that has been manually generated for assessment and comparison of the digit recognition methods. Red crosses correspond to samples that have been removed from the training set, because they could not be recognized visually.

For the three methods, SVM classification has been considered, and 10-fold cross validation has been implemented to evaluate the performance obtained with each set of features.

The continuous one-against-all strategy has been considered to extend SVM to more than two classes [7]. It involves using the continuous values of SVM decision functions rather than simply their binary signs. The class of a data point is whichever class has a decision function with highest value, regardless of sign.

The first method, based on vertical and horizontal projections, yields to an average classification accuracy of about 95% on the normalized dataset. In comparison, during an initial evaluation performed on a subset of the training dataset, the second method has achieved 93% of accuracy, while the third method lies around 90% of accuracy. It is worth noting however that both the second and third methods are invariant to rotation. We can thus reasonably expect that those two methods perform better in real life conditions, for which normalization has appeared to be an issue [1].

2.3 Players recognition conclusion

The purpose of this Section was to recognize the the players based on their appearance. Given detected individuals (see D7.1), we have been interested in characterizing their dominant color, and in recognizing their number. We have demonstrated that high accuracy was obtained, both for color (98%) and number (95%) recognition.

Those results are important and useful for the APIDIS project because they can help in:

- Tracking the players along the time. We have explained in D4.2 how the color and the number printed on players’ shirts could help in solving tracking ambiguities;

- Assigning key players to each actions of the game. Indeed, assuming that both the ball trajectory (see D5.1) and the players' identities are known, we become able to define the player that has thrown the ball to the basket, or taken the rebound. This is fundamental to generate a really personalized summary, which for example focuses on the actions involving a preferred player.

Future works should mainly involve careful comparative assessment of our proposed digit recognition methods, and real-life validation of our recognition algorithms.

3 Clock-events classification

This section presents the method proposed to detect and classify the actions observed during a basketball game. As explained in Deliverable 2.1, our purpose is to recognize the clock-events, i.e. all the events that have caused a stop, start or reinitialization of the 24" clock, and the ones that occur during periods for which the clock is stopped.

For this purpose, we assume an accurate monitoring of the 24" clock, and of the scoreboard. Hence, the clock state and the score information are assumed to be available, and should be completed with visual hints when necessary.

Rather than basing the classification on various detectors working in parallel to detect the actions, our method exploits the analysis of the temporal structure of a basketball game. This temporal partitioning of the game enables us to reduce for each given basketball phase the set of possible actions to discriminate.

We have focused on the main events occurring during a basketball game: field goals¹, violations, fouls, balls out-of-bounds², free-throws, throw-in (ball back to court), throw & rebound and lost balls. The detection of two parallel events, substitutions and time-outs, has not been implemented. These actions are described in details at the end of this deliverable in “*Annex 1: Features per event*”.

3.1 Tree-based organization of basketball actions

Two elements of a basketball game are particularly interesting for the interpretation of a basketball game event: the score and the shot clock³ state, as explained in Section 5 of deliverable D2.1. The acquisition of this data in real-time during the game is possible with a high reliability through a physical connection to the game scoreboard and shot clock or even through the analysis of the video acquired by a camera pointing towards these boards.

Since this shot clock and score information can be considered as very accurate, we have designed a tree of actions based on the scoreboard and the three states of the shot clock:

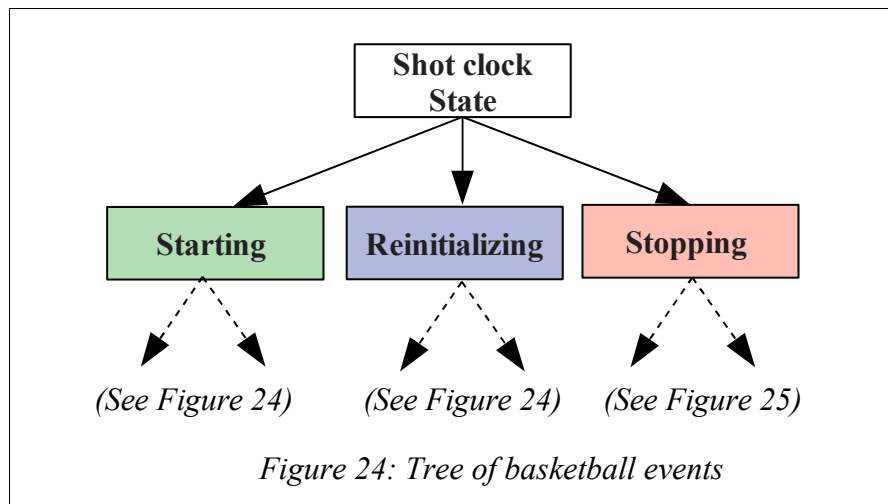
1. **Stopping:** The shot clock is stopped and the current countdown value remains visible. A reset of the countdown can occur during a stopping state.
2. **Starting:** After a stopping state, the shot clock countdown is re-started.
3. **Reinitializing:** The shot clock countdown is reinitialized to the 24 seconds and continues its countdown without stopping.

¹ A *field goal* is a basket scored on any shot other than a free throw.

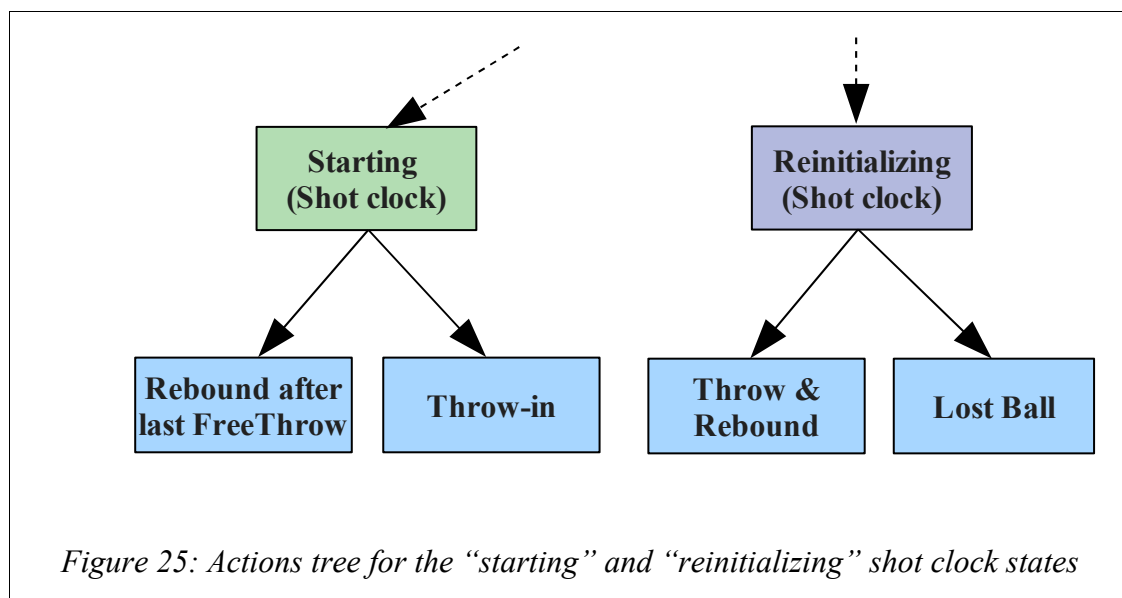
² The ball is *out-of-bounds* when it touches the floor or a person who is on, above or outside the boundary line.

³ The actions of a basketball game are regulated by a *shot clock* (also named clock-event) which is designed to increase the pace of the game. The offensive team must attempt a field goal before the shot clock expires (24 seconds), and the ball must then either touch the rim or enter the basket, or the offensive team will be assessed a violation resulting in a loss of possession.

Figures 24, 25 and 26 depict the actions tree based on the shot clock state. The score information is exploited in the “stopping” branch. In these figures, blue boxes correspond to actions and yellow boxes correspond to events characteristics used to discriminate the events.



We observe in Figure 25 that when the shot clock is in “starting state”, the only potential actions to have occurred are a rebound after the last free throw, or a throw-in. In “reinitializing state”, the only possible actions leading to this state are a throw&rebound or a lost ball. We will see below that simple detectors will help us take the correct decision for each node.



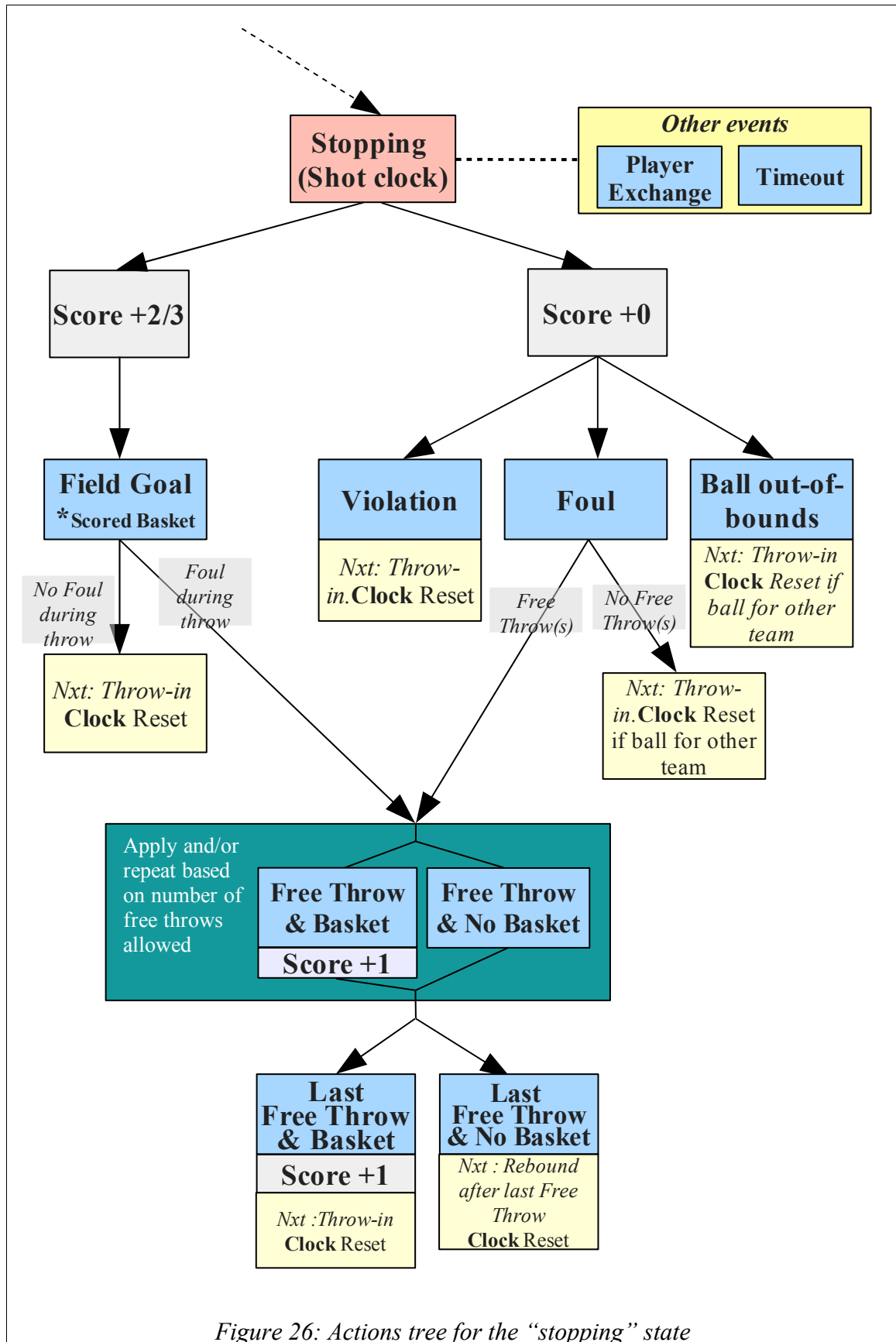


Figure 26: Actions tree for the "stopping" state

For the “stopping” branch depicted in Figure 26, the upper node is based on the evolution of the score. If the score increases by 2 or 3 points, this corresponds to a field goal. During the scoring action, fouls can be committed, leading to one or several free throws. If the score does not evolve when the shot clock is stopped, this can be due to a ball out of bound, a violation or a foul (which can lead to free throws).

As explained previously, other events (time-out and player exchange) can occur in parallel to other events during the “stopping” state. The extension of the current detection tools to these other events is planned for future work.

3.2 Events characteristics

In order to classify an event based on the tree structure described above, decisions have to be taken at each node based on a number of features.

The following table presents the characteristics of each action that can be used to discriminate the events (gray 'x' cell). By taking into account the temporal structure of the game described above, we have reduced the number of characteristics to analyze (gray cells with white square).

Characteristics	Clock state	Ball Position	Players position	Evolution of players position	Ball Possession	Previous actions	Next actions	Score
Field Goal	■	x	x	x			x	■
Violation	■				x		■	■
Foul	■						■	■
Ball out-of-bounds	■	■					■	■
Free Throw	■	■	■	x		■	x	■
Throw-in	■	x	x			■		■
Throw & Rebound	■	■	x					■
Lost Ball	■	■		x	x			■

Legend

x	Characteristic appearing in action
■	Characteristic appearing in action and required with structural analysis

With this analysis, the remaining game characteristics to observe are the clock state and score, the ball and player position and the previous and next actions. The next section details the analysis modules developed to combine these characteristics and take the decisions at each node.

3.3 *Detection modules*

The decision to take in the nodes of the tree presented in Section 3.1 sometimes involves some complex reasoning process. This is for example the case to detect free throw sequences when the clock is stopped, or to distinguish between a foul or a violation.

Managing such reasoning implies the extraction and analysis of features that are typically related to the spatio-temporal distribution of ball&players before and after the instant at which the clock state changes. In some cases, it could even require the recognition of referee gestures. Obviously, the capture and exploitation of such kind of information is delicate and subject to noise and ambiguity. Therefore, a multi-feature approach, involving some redundancy in the analysis process (e.g. by checking both the ball trajectory and the position of players around the basket) is recommended to infer a reliable decision. Handling multiple, potentially contending, features in a deterministic ‘if-then-else’ decision system is somewhat complex and might lead to the definition of sub-optimal rules by a human expert. In contrast, a classifier-based approach is expected to better cope with the intrinsic complementarity or redundancy of the multiple features. Hence, classifier-based approaches should be favored to define such detectors. However, the design and training of such classifiers are conditioned to the availability of a sufficiently large training set, which might involve a painful manual annotation procedure.

Therefore, we recommend an intermediate solution. In a first stage, simple ‘if-then’ rules are considered to automatically extract positive samples from a large video database. This extraction process would tolerate a significant amount of false positive. Those false positives would then be manually annotated, to feed the training of a classifier envisioned in the second stage.

In the following, we only consider and validate the first stage. We identify a set of features to discriminate between a set of events including free throw, time-out, throw in, violation, and foul. Simple ‘if-then’ rules are then considered to demonstrate the relevance of those features on our dataset.

The second stage, including actual training of classifiers, is not considered, due to the lack of training data. However, it is worth mentioning that the classifier could typically be a SVM. Recent works on visual event recognition [8] have also recommended the exploitation of ensemble of classifiers to improve the generalization capabilities of decision systems that have to live with a small number of training samples. In particular, ensemble of randomized trees appear to be particularly well suited to handle spatio-temporal events, based on simple tests on metrics that characterize the average spatio-temporal distribution of objects [9]. The features extracted in the first stage correspond to such spatio-temporal metrics, and are thus well suited to the design of such classifiers. This work is beyond the scope of APIDIS, but should be considered upon actual deployment of a permanent infrastructure in a sports room.

A. Throw-in and throw detector

Based on the ball position acquired during the last seconds by the ball tracking algorithm, the ball position module outputs the probability of the ball to be out-of-bounds (outside the playing field) and to be near the basket.

In order to take into account the acquisition noise for the ball position (potential loss of ball position during several frames or positioning error), and a potential delay in the shot clock (also called 24" clock) state update, the detector considers the ball position inside a temporal sliding window.

This process is illustrated in Figure 27 which depicts in black the distance of ball to the field line (d_{oob}), in green a simple out-of-bound probability P_{oob} and in blue the same probability with a temporal sliding window applied P_{oob}^{max} .

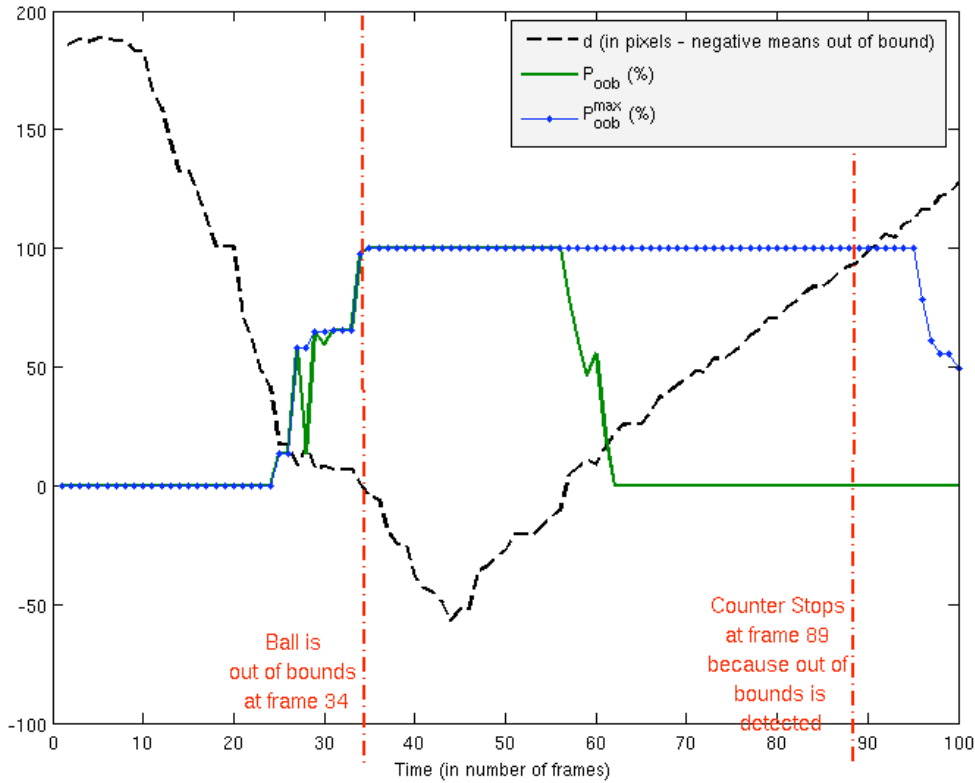


Figure 27: Illustration of detection of out-of-bounds detection

If the ball is detected outside the field, $P_{oob}=1$. When the ball is inside the field and $d \leq \alpha_{oob}$, $P_{oob} = \frac{\alpha_{oob} - d}{\alpha_{oob}}$. Finally, $P_{oob}=0$ when $d_{oob} > \alpha_{oob}$. P_{oob}^{max} corresponds to the maximum out of bound probability for the β_{oob} last frames. This sliding window enables to take into account a delay in the actualization of the shot clock. We observe in Figure 27 that this temporal filtering enables to detect the out-of-bound event, although the counter only changes state 2.5 seconds later.

The probability of the ball to be near the basket P_{bas} and P_{bas}^{max} are calculated in a similar way with parameters α_{bas} and β_{bas} .

The α_{oob} and β_{oob} parameters values are currently set to 20 and 40 respectively and α_{bas} and β_{bas} are set to 20 and 20 respectively. These values should be refined when larger dataset will be available, as currently P_{oob}^{max} is only used for the single out-

of-bound event of the period, and P_{bas}^{max} is only used in the “reinitializing” state to discriminate a throw&rebound from a lost ball (18 occurrences).

Improved robustness could also be obtained by taking the position of players into account, i.e. during a throw-in (ball back to court) action, a player goes out of the field for a short period.

B. Free throw and time-out detector

This module detects in “stopping” state if a free-throw and/or a time-out event has occurred. In order to increase the detection robustness, four features are taken into account:

1. **Duration of the “stopping” state.** This feature discriminates free throws and time-outs from shorter actions that are directly followed by a throw-in (ball back to court), such as field goals, violations and ball out-of-bounds
2. **Pattern of players in free-throw position.** We consider that for a free throw, at least one player is inside the free-throw semi-circle and several players are located near the free-throw lane (also known as key). To measure this last characteristic, we calculate the sum of the distance of the closest 4 players to the free-throw lane⁴. This summation is assumed to remain small for a significant period of time during the clock stop period. Figure 28 and 29 illustrate this pattern for a free-throw, respectively from a camera view and from the output of the tracking module.
3. **Pattern of ball movement.** The ball movement is analyzed to detect a throw from the free-throw line to the basket. The ball trajectory is considered as being the one of a free-throw if the ball goes from the free-throw semi-circle to/near the basket following a straight trajectory.
4. **Players center of gravity.** While during a free throw, players are located inside the field, both teams remain within the vicinity of their team bench area during a time-out. Hence, using a 2-mean clustering algorithm, and measuring the average distance of players to the center of their cluster provides a valuable feature to discriminate between free throws and time-out periods.

Features 2, 3 and 4 will be used to differentiate a free-throw from a time-out. However, as no time-out has occurred during the available period of the APIDIS dataset, the implementation and validation of the detection of time-out events is kept for future work.

4 The distance of point (x_0, y_0) to the line specified by two points $\mathbf{x}_1 = (x_1, y_1)$ and $\mathbf{x}_2 = (x_2, y_2)$ is given the following formula:

$$d = \frac{|(x_2 - x_1)(y_1 - y_0) - (x_1 - x_0)(y_2 - y_1)|}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}}$$



Figure 28: Typical positioning of players for a free throw event (camera view)

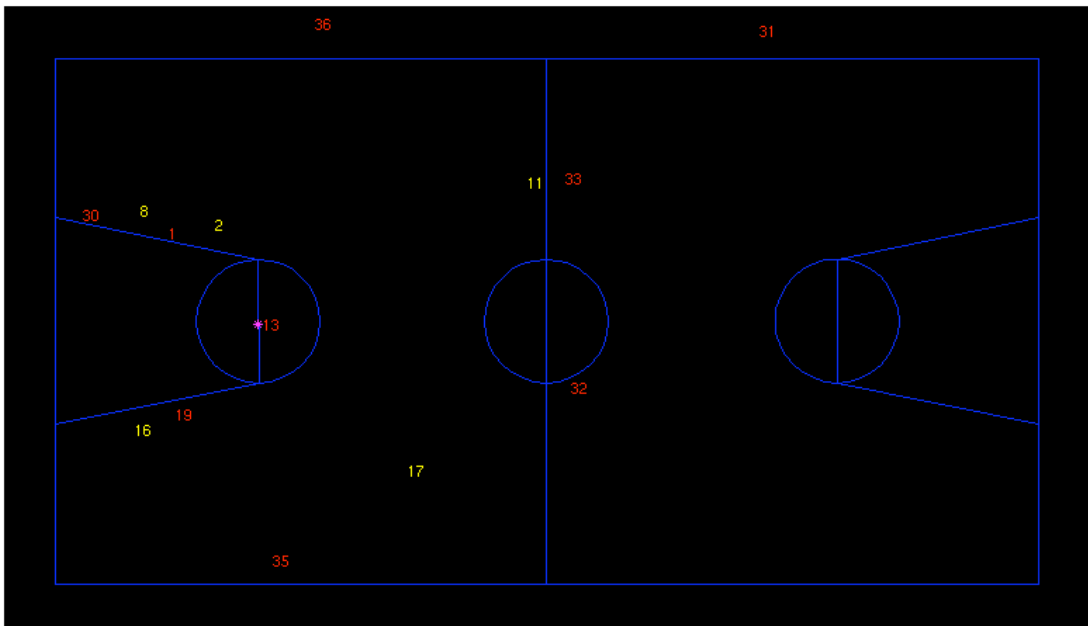


Figure 29: Typical positioning of players for a free throw event (players position calculated by tracking module)

C. Violations and fouls

A last pair of actions, namely the violation and the foul, have to be discriminated in the tree presented in Figure 26.

Knowledge of ball and player positions might give some hints to identify violations, but we do not expect to be able to achieve high accuracy when discriminating between those two actions.

Therefore, we recommend to design specific detectors to infer the probability of facing a violation (typically an offending player standing for a long time in the racket, or an offending player going back in his/her defending area with the ball), and to fill-in the XML annotation file with the two possible events, and their associated posterior probabilities. In that way, the summarization engine can decide to include those actions or not, depending on the user request and on the certainty associated to each action.

To train the ad-hoc violations detectors, the availability and annotation of a large database is required. Hence, this work is beyond the scope of APIDIS, and should be considered once a permanent infrastructure is deployed.

3.4 Results

The events detection was performed on the second quarter of the first APIDIS dataset. The players position was obtained from the tracking module while the ball position was obtained from manual annotations. The shot clock and scoreboard information was also obtained from manual annotations.

The following table shows the events detections results.

	Action	Number of instances	Correctly detected	Correct detection rate
1	Ball-back-to-court	22	22	100 %
2	Ball-out-of-bounds	1	1	100 %
3	Field Goal (2 points)	12	12	100 %
4	Field Goal (3 points)	3	3	100 %
5	Lost ball	3	3	100 %
6	Throw & Rebound	15	14	93 %
7	Foul - Violation	2	2	100 %
8	Violation	2	2	100 %
9	Foul (& Free Throw)	6	6	100 %
10	Foul (no Free Throw)	2	0 %	0 %
11	Player exchange	3	0 %	0 %
Total		71	65	92%

We observe that most events are correctly detected. One *Throw & Rebound* has not been correctly detected (line 6) because of errors in the data describing the ball position

during that shot. Two *Fouls* occurring very close to the field limits were mistaken with *out-of-bound balls* (line 10) and no *Player exchange* were detected (line 11) as no detector has been implemented for this event.

Regarding the fouls, the system is sometimes incapable of deciding whether the actions are violations or fouls. In that case, the events are labeled as “Foul – Violation”, a category (line 7) created for these ambiguous events.

In order to be able to provide a summary of the game, which is the final goal of this event detection, some actions are more important than others. Field goals and rebounds are of course the most important actions, while player exchange are of a much lower interest. The results above show that we perform very well on those important actions (97% of correct detection).

3.5 Events classification conclusion

We have presented in this chapter an event detection technique exploiting the temporal structure of a basketball game. In particular, by taking advantage of the shot clock and scoring information, the number of possible events at a given moment can be greatly reduced. Following this first step, two detection tools have been developed to fully discriminate the actions.

The current event detection system has two main limitations. First, the system is unable to discriminate violations and fouls which do not involve free-throws. This could be achieved by obtaining this information from the scoreboard when available (most professional scoreboards today display fouls). The second limitation is the impossibility to discriminate out-of-bounds and violations located near the field border. This limitation could be removed by improving the ball tracking or by analyzing the arbiter gestures.

A remaining task will consist in determining the exact localization of the event on the basketball court. This consist for example in identifying which player is the author of the shot or of the violation. A reliable information about the ball position can drastically reduce the complexity of this task.

The system has been tested on the second quarter of a basketball game and has proven its efficiency on this dataset with a correct event detection of 92%. However, a validation on larger period of time is essential to improve the system robustness. With the target of a commercial deployment in mind, this validation on larger datasets should be achieved in two phases. First, a pre-selection of ambiguous actions will be made based on the system proposed in this deliverable. In that stage, detection parameters would be relaxed, to make sure that no relevant actions is missed, while tolerating a significant amount of false positive. The second step would consist in training (ensemble of) classifiers, based on the annotation of actions selected during the first step.

4 Conclusion

This document has presented the high level interpretation mechanisms that have been considered by APIDIS to personalize the summarization of basket ball games. Both the recognition of players and events have been considered.

Regarding the players, we have shown that the two teams could be discriminated based on color attributes, while the identity of players could be inferred from the number printed on their shirts. Mean-shift segmentation has been proposed to extract the digit regions on the player's shirt. Digit recognition has then been implemented based on three different methods. A preliminary assessment, based on a manually extracted digits, has demonstrated the relevance of each recognition method. However, further experiments integrating the automatic segmentation and recognition of digits in real-life conditions are needed to quantify and compare accurately the performance and complexity of the three methods. This work is in progress and should be reported either in D5.3 or D7.2, by the end of the project.

Regarding event detection, the deliverable has focused on the recognition of the clock-events defined in D2.1, assuming the 24'' clock and score board were monitored. We have proposed a tree-based classification of the events. The decisions in the nodes of the tree are generally taken based on the reliable information collected from the clock and scoreboard. In cases for which this information was not discriminant, e.g. to decide between a free throw or a time-out action for example, dedicated if-then-else classifiers have been designed based on a set of features measuring ad-hoc metrics about players and ball position. Training of more robust classifiers (SVM or ensemble of trees) based on a large and extensive training dataset is beyond the scope of the project, but is suggested as a valuable path for future work.

Our experimental results have demonstrated that both the player and event detection can be reached with more than 90% of accuracy, thereby making the concept of automatic and personalized summarization realistic.

5 References

- [1] Damien Delannay, Nicolas Danhier and Christophe De Vleeschouwer, *Detection and recognition of sports (wo)man from multiple views*, Third ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC), 2009
- [2] T. Ojala, M. Pietikainen, and D. Harwood, “*A comparative study of texture measures with classification based on feature distributions*,” *Pattern Recognition*, vol. 29, no. 1, pp. 51–59, 1996.
- [3] T. Ahonen, A. Hadid, and M. Pietikäinen, “*Face description with local binary patterns: Application to face recognition*”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, No. 12, December 2006, pp.2037-2041.
- [4] S. Mallat and Z. Zhang, “*Matching pursuits with time-frequency dictionaries*”, *IEEE Transactions on Signal Processing*, vol.41, n°12, December 1993, pp.3397-3415.
- [5] L. Jacques and C. De Vleeschouwer, “*A geometrical study of matching pursuit parameterization*”, *IEEE Transactions on Signal Processing*, vol.56, n°7, July 2008, pp. 2835-2848.
- [6] N. Otsu, “*A threshold selection method from gray-level histograms*”, *IEEE Trans. Syst. Man Cybern.*, vol 9, pp 62-66, 1979
- [7] Vapnik, V. (1998). *Statistical Learning Theory*. Wiley-Interscience, NY.
- [8] C. Simon, J. Meessen, C. De Vleeschouwer, “*Visual Event Recognition using Decision Trees*”, *Multimedia Tools and Applications*, 2010
- [9] C. Simon, J. Meessen, D. Tzovaras, and C. De Vleeschouwer, “*Using decision trees for knowledge-assisted topologically structured data analysis*,” in 8th International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS), Santorini, Greece, June 2007.

Annex 1: Features per event

In the following, we follow the terminology of the FIBA (International Basketball Federation)⁵.

A. Field goal

This action occurs when a player throws the ball toward the basket and scores. The following features are observed when a field goal occurs:

1. **Clock**
The clock is stopped and reset.
2. **Ball position**
During this event, the ball is close to the basket and enters it.
3. **Players position**
Many players are usually in the key (restricted zone around the basket), and several players are near the ball.
4. **Evolution of players position**
After the field goal throw, most players go in the opposite direction on the field.
5. **Next actions**
After scoring, if no foul is observed, the next action is a throw-in. If a foul is observed, the next action is a free-throw.
6. **Score**
After a field goal, the score of the offensive team is increased by 2 or 3 points, depending on the position of the player who scored.

B. Violation

This action occurs when a player does not respect one of the basketball rules. The following features are observed when a violation occurs:

1. **Clock**
The clock is in "*stopping*" state and reset, except in the case that no free-throw is requested and the violation is done by the defensive team.
2. **Ball Possession**
The ball possession changes from one team to another after this action.
3. **Next actions**
A violation is followed by a throw-in.
4. **Score**
The score remains the same (exception for an interference violation).

C. Foul

A foul is an infraction of the rules concerning illegal personal contact with an opponent and/or unsportsmanlike behavior. The following features are observed when a foul occurs:

5 Official Basketball Rules 2008. <http://www.fibaeurope.com/files/%257BA1E7F7B7-BE56-4002-813A-F1BB6F1B2346%257D.pdf>

1. **Clock**

The clock is in "*stopping*" state and reset, except in the case that no free-throw is requested and the violation is done by the defensive team.

2. **Next actions**

◦ *Free Throw*

- A free-throw can be awarded when a player is victim of a personal foul while in the act of shooting. If the foul causes the player to miss the shot, the player receives **two or three** free throws depending on whether the shot was taken in front of or behind the three-point line. If, despite the foul, the player still makes the attempted shot, the number of free throws is reduced to **one**, and the basket counts.
- In the case of team foul penalty (a team commits a set number of fouls)
- In the case of an unsportsmanlike foul, disqualifying fouls or technical fouls

◦ *Throw-in*

- In the case of personal foul on a player not in the act of shooting.
- In the case of double foul (in case of field goal of at least a team had control on the ball)

◦ *Jump Ball*

- In the case of double foul when neither team had control of the ball

3. **Score**

The score remains the same.

D. Ball out-of-bounds

The ball is out-of-bounds when it touches the floor or a person who is on, above or outside the boundary line. The backboard supports, the back of the backboards or any object above the playing court are considered out-of-bounds.

The following features are observed when a ball is out-of-bounds :

1. **Clock**

The clock is in "*stopping*" state. It is reset if the last player touching the ball before the out-of-bounds is in the offensive team.

2. **Ball Position**

During this event, the ball is close to the field boundaries, and usually crosses it.

3. **Next actions**

This action is followed by a throw in achieved by the team that is not responsible for the out-of-bounds event.

4. **Score**

The score remains the same.

E. Free-Throw

A free-throw is a free shot taken from the foul line awarded to a player whose opponent committed a foul. The number of free throws awarded (1, 2 or 3) depends on the foul type.

The following features are observed during a free-throw :

1. **Clock**

The clock is stopped, and is re-started when the last free-throw does not enter the basket. If this last free-throw does enter the basket, the clock is started at the throw in following the free-throw.

2. **Ball Position**
While the player is preparing to throw the ball, the ball is located in the semi-circle located at the end of the free-throw area. The ball is then thrown toward the basket.
3. **Players position**
One player is located in the semi-circle located at the end of the free-throw area and several other players are located near the free-throw lines.
4. **Evolution of players position**
The players in the free-throw rebound places are rather static in the period previous to the throw. Right after the last free-throw, most players in the free-throw rebound places rush towards the basket.
5. **Previous Actions**
The previous actions is a foul caused by the defensive team.
6. **Next actions**
If the last free throw is scored, the game directly starts when the basket is scored. Otherwise, the next action is a throw-in.
7. **Score**
Each time the ball enters the basket during a free-throw, the score is incremented by one point.

F. Throw-in (Ball back to court)

This action consists in putting the ball in play from out-of-bounds. The following features are observed when a throw-in occurs:

1. **Clock**
The timer is in "*starting*" state (a reset does not always occur before the timer start).
2. **Ball position**
The ball is located near a boundary of the field.
3. **Players position**
The player who puts the ball in play is located out-of-bounds.
4. **Previous actions**
A throw-in occurs after the following actions :
 1. Field goal
 2. Violation
 3. Ball out-of-bounds
 4. Fouls (free throws may occur before the throw-in)
5. **Score**
The score remains the same.

G. Throw & Rebound

This action occurs when a player, regardless of his team, recovers the ball after a throw.

The following features are observed when a rebound occurs:

1. **Clock**
The clock is reset without being stopped
2. **Ball Position**
During this event, the ball remains located in the key (restricted zone around the basket), and passes by a high position phase.

3. Players Position

Several players are usually located inside the three point line, and near the basket.

4. Score

The score remains the same.

H. Lost ball

This action occurs when a player intercepts the ball from the other team.

1. Clock

The clock is reset without being stopped

2. Ball Position

During this event, the ball is usually not close to the basket. When close to the basket, the ball is not in a high position phase.

3. Evolution of players position

After a lost ball, the direction of most players is inverted. This is mainly true for offensive

4. Ball Possession

The ball possession changes from one team to another after this action.

5. Score

The score remains the same.

I. Substitution

A substitution is an interruption of the game requested by the substitute to become a player. Substitution opportunities occur in the following situations:

- For both teams, the ball becomes dead, the game clock is stopped and the official has ended his communication with the scorer's table.
- For both teams, the ball becomes dead following a successful last or only free throw.
- For the non-scoring team, a field goal is scored in the last two (2) minutes of the fourth period or the last two (2) minutes of each extra period.

J. Time-out

A time-out is a one minute interruption of the game requested by the coach or assistant coach. Time-out opportunities occur in the following situations

- For both teams, the ball becomes dead, the game clock is stopped and the official has ended his communication with the scorer's table.
- For both teams, the ball becomes dead following a successful last or only free throw.
- For the non-scoring team, a field goal is scored.