Machine Vision and Applications
DOI 10.1007/s00138-012-0471-z

ORIGINAL PAPER

Vision system for tracking handball players using fuzzy color processing

Catarina B. Santiago · Armando Sousa · Luis Paulo Reis

Received: 23 February 2012 / Revised: 21 July 2012 / Accepted: 21 November 2012

Abstract The sports community needs technological aid to extract accurate statistics and performance data from both practice sessions and games. To obtain such information, players must be tracked over time and their movements processed so that individual actions and team plays are simultaneously analyzed. In order to perform this analysis in an automated, formal and accurate way, the authors developed a cost conscientious processing system fed by two overhead cameras (roughly one video stream for each half-field). Players are detected by vest colors, and Fuzzy Logic is used to allow for a given color to be shared by different teams. Color models for the background and the teams are dynamic over time to make up for changes in natural lighting conditions and consequent color changes. Player tracking is further enhanced using Kalman Filtering. Some examples of the analysis, made possible by the proposed system, are shown. Results are based on videos collected during the Portuguese Handball SuperCup competition for the year 2011.

Keywords Image processing · Video processing · Player detection · Player tracking · Fuzzy Logic · Player statistics · Game analysis

C. B. Santiago (✉) · A. Sousa
INESC TEC (formerly INESC Porto) and Faculty of Engineering, University of Porto, Porto, Portugal
e-mail: catarina.santiago@fe.up.pt

A. Sousa
e-mail: asousa@fe.up.pt

L. P. Reis
School of Engineering, University of Minho, Braga, Portugal
e-mail: lpreis@dsi.uminho.pt

L. P. Reis
LIACC, Faculty of Engineering, University of Porto, Porto, Portugal

1 Introduction

Sports play an important role on nowadays society and there is an increasing interest by the sports' community on having mechanisms that allow them to better understand the dynamics of players and teams. Often, this information is manually extracted by operators that, after the game, visualize game recordings (frequently TV footage) and perform hand annotation, which is a time consuming and error prone task.

The effort and time involved in such tasks are huge and the results are subjective to the person performing it, as mentioned in [6,31,32]. From these works it is possible to notice that recording a video of the entire field is already very useful for sports' experts since they are able to visualize all players and ball movements. This allows them to identify weaknesses, define tactics and new training directions to improve the teams' global behavior, reduce the teams' weaknesses and explore the opponents' weak points. However, the next step, automatic detection and tracking enables a systematic, objective, accurate and consistent analysis. The process will also be much more time efficient.

In this paper, we present a non-invasive, automatic visual system for detecting and tracking handball players based on a Fuzzy inspired methodology [35].

The document structure is as follows: the next section presents relevant information on image segmentation methodologies and automatic visual systems for detecting and tracking players. Section 3 discusses the proposed architecture principles, providing an overview of the methodology used. Section 4 presents the results achieved, including a detailed sensitivity analysis and, finally, Sect. 5 concludes this paper with the main conclusions and further investigation directions.

Published online: 30 December 2012

2 Fundamentals and related research

The field of automatic player/team detection and tracking represents a very challenging problem due to the complexity of sports analysis itself, as there are several similar fast moving targets that are frequently changing direction and contacting with each other. Although there are two main categories of technologies used for automatic detection and tracking: intrusive (where special tags or sensors are placed on the targets) and non-intrusive (where there are no extra objects in the game environment). This paper only addresses the second category because one of the major problems of intrusive systems is that usually regulations do not allow their usage on official games, where the most interesting information for game analysis is provided.

Non-intrusive systems have vision as the main sensory source and, therefore, devote a great effort on developing image processing methodologies that allow a good video/image segmentation.

The two following subsections provide a good insight into the existing video/image segmentation methodologies as well as into systems devoted to the player detection and tracking problem.

2.1 Video segmentation

Video segmentation is the first step, and probably the most critical, in any video (image) processing system because the quality of the final result is highly dependent on a good segmentation. There are two main categories of video segmentation:

– Temporal segmentation: segments the video into meaningful temporal sequences. It is usually used as the first step of video annotation and segments the video taking into account similarities/dissimilarities between successive frames [19].
– Spatial segmentation: aims to divide the content of each frame into homogeneous regions that correspond to independent objects.

The focus of this work is on spatial segmentation; hence, for a detailed survey on temporal video segmentation please refer to [19].

Spatial video segmentation inherits many of the methodologies used for image segmentation and can also make use of the temporal characteristics inherent to video.

Image segmentation methodologies can be subdivided into the following categories [8]:

– *Histogram thresholding* by determining the peaks or modes of the uni/multi-dimensional histogram of the image.
– *Feature space clustering* by grouping the image feature space into a set of meaningful groups or classes based on intensity, color or texture characteristics of pixels.
– *Region based* which includes region growing, Watershed transform and region split and merge. These methods attempt to divide the image domain based on the fact that adjacent pixels in a same region have similar visual features (color, intensity, texture or motion).
– *Edge detection* that segments the image by finding the edges of each region using an edge detector.
– *Fuzzy methods* allow classes and regions to have a slight uncertainty and ambiguity, which is generally the case of image processing.
– *Neural networks* allow parallel processing and adding non-linearities. They can be used either to pattern recognition, classification or clustering.

Nowadays, the tendency is to aggregate techniques from different categories to achieve better results. A typical example of this is the JSEG algorithm [10] that initially clusters colors into several representative classes, afterwards replaces each pixel with its corresponding color class label and only then applies a region growing process directly to the class map.

As stated previously, videos can also be segmented based on temporal properties, namely on motion along time. In order to perform this task there are two main approaches: Background Subtraction and Optical Flow. Background Subtraction methods model background as a simple static image without objects; more complex variations propose methods such as moving average [14], median filters or mixture of Gaussians [13]. Optical Flow strategies [2] are based on the motion of brightness/color of parts of the image associated with objects—the method assumes a linearization of the objects' trajectories and, therefore, primarily applies to small displacements.

Following the tendency, the proposed method for the video segmentation step is a methodology that combines temporal information through dynamic background subtraction with physical and color information through a Fuzzy color calibration based on region growing and pixel labelling.

2.2 Player tracking using vision systems

There are two main streams of research in this area depending on how video footage is obtained: television broadcasting or dedicated camera systems. In both cases, advanced image and video processing techniques must be used due to the complexity of the problem.

Usually, these systems involve three steps: image segmentation, player detection, and finally, player tracking.

2.2.1 *Television broadcasting*

Nowadays, most games (especially soccer) are filmed by television networks and are available for everyone; therefore, this information is not hard to obtain and to create instruments to perform player detection and tracking.

However, most of the systems use a single broadcast camera that does not provide an entire view of the playing area and only gives useful information on wide angle images.

The first step (image segmentation), in the majority of the literature, consists of modelling the background which can be based on color histograms [33], color intensities [26], clustering [18] or more robust and complex methodologies, such as Mixture of Gaussians (MOG) [4,16,36]. The usage of dynamic methodologies such as dynamic clustering and MOG allows having a more robust solution that can better adapt to colour and luminance changes which is crucial in outdoor applications. Nevertheless, MOG methodologies may not be effective in highly static matches, because players that stay still for too long may be absorbed into the background model.

Afterwards, the remaining regions are filtered to identify players, using simple clues such as physical constraints [18,26], boost cascade detector of Haar features [33], clustering algorithms [16], colour template matching [12] or support vector classification [36]. The work [22] adopted a slightly different methodology and performs player detection without background subtraction using two classifiers that are previously trained with hand selected samples. It is important to notice that the usage of classifiers, template matching and boost cascade detectors, requires prior training. Therefore, before analyzing each game, a set of representative samples must be collected. The usage of background subtraction methodologies usually speeds up the processing time, since only a few regions need to be analyzed with more complex algorithms.

Player tracking is achieved with weighted graphs [26], Kalman Filters [22], fast level contours [18], CamShift [16] or probabilistic models such as Markov Chain Monte Carlo [33] and Particle Filters [15,36]. The choice of the tracking methodology plays an important role when dealing with occlusion and merging situations. For example, CamShift cannot deal well with merging, even if it is for short periods of time. On the other hand, probabilistic models are well suited for the non-linear movement often made by the players on the field. However, the computational effort is high.

Once the players' trajectories have been detected, some authors still convert the players' coordinates into real-world coordinates [5,16,33], which is extremely important if ball tracking [5,18,36] or high-level game analysis [5,16,36] is to be performed.

Despite these efforts, it is possible to verify that the detection and tracking accuracy is never 100 % because, although temporary occlusion may be handled, more persistent situations, like overcrowded scenes, serious video blur or abrupt camera motion, may lead to miss-tracked players.

2.2.2 *Dedicated cameras*

Dedicated camera systems are mostly used in indoor environments because the smaller playing area makes it possible to use a single camera [25]. However, authors tend to use two [3,24,27] or even more cameras [1,9,11,28] to cover the entire field. Benefits of using multiple cameras include higher resolutions, overlapped regions, which allow minimizing occlusion problems and open the possibility of 3D localization [1].

Dedicated systems follow the same processing flow as broadcast camera-based systems. So, the first stage usually consists of eliminating the play field area (that has no useful information), either by static background modelling [25] or dynamic methods, with more or less complex methodologies (ranging from median filters to MOG, or mixture of both) that take into account light variations [3,9,11,24,27,28].

The results of the previous step usually contain noisy regions, but only some of them really correspond to players. These regions are detected using morphological filtering [11] aided, more recently, by an Adaboost classifier [3], color histograms [27,28], templates [24] that are updated along the game or occupancy maps [1,9] generated from multiple views of the field.

Tracking is performed via weighted graphs [3,11], probabilistic methodologies [25,27], Kalman filters [28] or more simply methodologies based on velocity constraints [1] or fixed area around the players [9]. An interesting aspect—"Closed world assumptions" —used by [20] and followed by [24] defines several heuristics that can be used on semi-controlled environments and include the partition of the world into Voronoi cells that may be occupied by a single player and, hence, improve the tracking.

Dedicated systems place the cameras at specific locations and, therefore, most authors compute the cameras homographies and translate the players' positions into world coordinates [1,3,11,24,25,27,28]. Additionally, [9] scans the player's regions for digits to determine the player's number, [28] is able to track the 3D position of the ball and [27] performs high-level game analysis.

3 Proposed vision and processing system

The challenges of defining a vision system able to identify and track the game elements in an invasion team game are huge due to the dynamic and spatial characteristics of the game itself. In fact, handball is played in an area of 20 by 40 meters and it is quite a dynamic game, with high physical

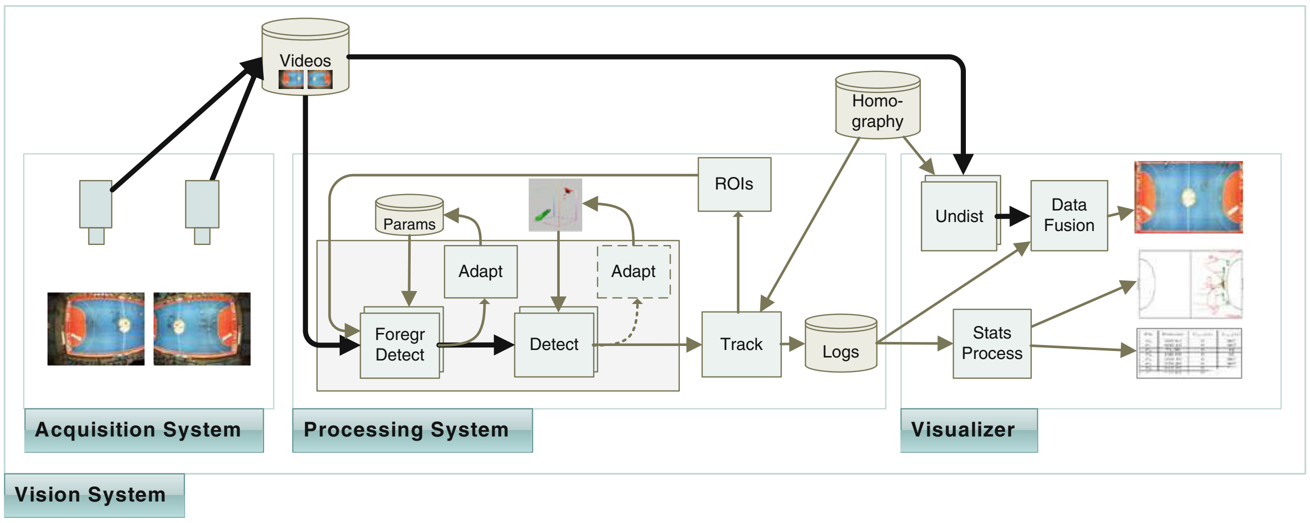


Fig. 1 System's architecture

contact among players and rapid movements (a player can achieve velocities higher than 5 m/s). These characteristics impose a careful choice on the system's architecture, which includes choosing not only the cameras and their disposition but also defining the software design.

The system must cover the entire handball field including the extra border, the image should have enough resolution to correctly detect all players and capture images with a frame rate adequate to the involved speeds. The characteristics of the problem and of the sports hall place the ceiling as the best spot to set the camera system, since there is no interference from the crowd, a single player never fills the entire field of view of the camera and a bird's-eye perspective usually means less occlusion and/or merging situations (this solution was also adopted by [21,24]).

On the other hand, placing the cameras on the ceiling generally forces the usage of multi-camera systems for additional resolution. Although this choice may result in a more complex system, it also carries the advantage that some parts of the field are covered by more than one camera, which provides two views of the same portion of the field that can be used to overcome or minimize occlusion situations.

Given the enumerated characteristics, the proposed system uses two Gigabit Ethernet cameras DFK 31BG03.H model from Imaging Source. Resolution is $1{,}024 \times 768$ pixels and the camera delivers 30 frames per second. The used lenses are Computar T2Z1816CS Vari-focal lens with focal distances ranging from 1.8 to 3.6 mm (wide angle lens).

When compared with other implementations, the presented architecture and hardware choice allow an "easy" setup that can be transferred between sports halls. The choice for industrial grade Gigabit Ethernet interfaced cameras allows reliability with digital quality, high data rate and low cost: no frame grabber, common hardware, low cable costs while still allowing large distances. The available data rate is very high and that, in turn, allows large frame rates with high resolutions. The chosen Vari-focal lens has an interesting price and allows (manual) "zooming" of different pavilions that have ceilings at different heights. These advantages come at the expense of an important image distortion.

A three module software system is proposed, one responsible for acquiring the images from the two cameras (Acquisition System) and another for the off-line processing of the two video streams (Processing System). This last one detects and tracks the players and generates a log file with the players' positions, so that they can be used by the sports community to perform game analysis and infer game statistics. Additionally, another application (Visualizer) is able to merge the two video streams and the log file to create a global image of the field with the players highlighted, so that the user can latter analyze the collected information. Figure 1 illustrates the architecture described.

Figure 2 shows the images collected from the two cameras. It can be seen that the barrel effect is wide due to the usage of inexpensive lenses. The mentioned combination offered excellent performance at an interesting cost at the time of the start of the project, in 2010.

3.1 Player detection

Player detection is achieved through color identification and is composed of three steps. The first step (Sect. 3.1.1) consists of the user-based color calibration of each team, using a region growing method allied with a Fuzzy categorization methodology (evolution of [29]). This calibration is responsible for subdividing the color space into subspaces, which are not necessarily disjoint since there may be colors common to both teams (for example, it is common to have teams

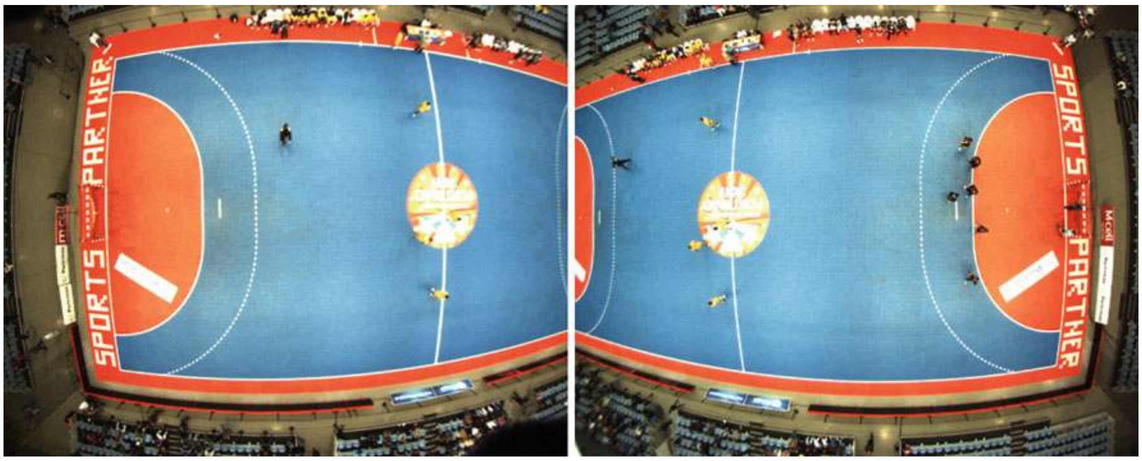


Fig. 2 Images collected from the two camera's system

with white stripes). Afterwards, the user manually indicates the location of each player on the field by clicking on the videos.

The second step (Sect. 3.1.2) consists of detecting foreground regions through a dynamic background subtraction method, which uses an empty image of the field and a dynamic threshold that is continuously and locally updated at each new frame.

After the foreground pixels are identified, their color is compared against the color subspaces and classified into one of the teams (Sect. 3.1.3). In case there is a belonging tie between teams, information of adjacent pixels is used to break the tie. Additionally, the teams' color subspaces are updated with new information.

Finally, pixels are aggregated to form blobs and categorized into player or no player (noise), according to size and density restrictions. The center of mass of the blob is considered the player's position, that is afterwards transformed into field coordinates using the cameras' homographies.

The usage of simple, robust and parallelizable methodologies is intentional, so that, making use of parallel technologies (multi-threading, Open Multi-Processing (OPENMP) [7], Graphics Processing Unit (GPU) [23] and code optimization), real-time processing can be achieved.

3.1.1 Color calibration

The color calibration is performed under user supervision and is achieved using a region growing method allied with a Fuzzy categorization methodology [29].

Let us define color subspace S_c as the set of RGB color triplets that are tagged as having the colors of the vests of team c. The initial colour seeds $C(x_s, y_s)$ for each color subspace S_c are set manually using the mouse to click on the objects that will be segmented. Afterwards, the surrounded pixels' colors $C(x_a, y_a)$ are agglomerated around these seeds using color distance criteria. Color expansion is performed on the HSL (Hue, Saturation and Luminance) color space to minimize the effects of shadows and light variations.

Regions growth is performed in all directions (using a 8 neighbour mask n_8) in a recursive way until reaching a pixel that, in terms of color, is more than a global threshold (C_{ThresG}) away from the seed or more than a local threshold (C_{ThresL}) away from its previous neighbor $C(x_p, y_p)$, according to the following definition (both thresholds are user definable):

Rule 0

$$\begin{aligned}&C(x_a, y_a) \in S_c \Leftrightarrow \forall(x_a, y_a) \in\\ &n_8(x_p, y_p) \wedge \Delta(C(x_a, y_a), C(x_p, y_p)) < C_{\text{ThresL}}\\ &\wedge \Delta(C(x_a, y_a), C(x_s, y_s)) < C_{\text{ThresG}})\end{aligned}$$

where

- $C(x, y)$ is the HSL color of pixel at location (x, y),
- $n_8(x, y)$ are the eight neighbors of the pixel at location (x, y), and
- $\Delta(C_1, C_2)$ is a configurable weighed distance function, involving HSL components of colors C_1 and C_2.

During the color expansion, each color value is attributed a given belonging degree to the subspace being calibrated. This value is stored in a lookup table that contains, for each color triplet, the belonging degree to each subspace. Despite the expansion being performed on the HSL color space, the color lookup table is built on the RGB (red, green, blue) color space, which is the format provided by the camera. Therefore, in execution time, labelling is performed via this lookup table, which allows fastening the process.

The Fuzzy belonging degree μ of the color C of a pixel P of coordinates (x_P, y_P) to a given color subspace Sc

Table 1 Mapping of B_{Sc} to Fuzzy belonging μ

Color	B_{Sc}	μ_{Sc}
Not the color	C_0	0
Resembles the color	C_L	0.5
Is the color	C_F	1
Is a seed color	C_S	1

is $\mu_{Sc}(C(x_P, y_P))$ and can assume 4 levels, according to Table 1. By default, and before the calibration takes place, all the colors are categorized with no belonging degree to every subspace.

In order to determine the belonging degree of the color triplet to the respective subspace, the following rules are sequentially applied during the color calibration region growing process:

Rule 1

$$B_{S_c}(C(x_a, y_a)) = C_S \Leftarrow C(x_a, y_a) \in (S_c \wedge \Delta((x_a, y_a), (x_s, y_s)) < \mathit{Small}_{\text{ThresG}})$$

If the pixel was assigned to the subspace, is physically quite close to the initial seed pixel and the color distance to the initial seed pixel is less than a small threshold ($\mathit{Small}_{\text{ThresG}}$), then it is also assumed to be a seed pixel with a full belonging degree.

Rule 2

$$B_{S_c}(C(x_a, y_a)) = C_F \Leftarrow C(x_a, y_a) \in (S_c \wedge \Delta(C(x_a, y_a), C(x_s, y_s)) < \mathit{Medium}_{\text{ThresG}})$$

If the color distance to the initial seed pixel is less than a medium threshold ($\mathit{Medium}_{\text{ThresG}}$) for the growing process, then the pixel is categorized with a full belonging degree but without being a seed.

Rule 3
Otherwise, and in case the pixel obeys to the region growing conditions (Rule 0), it is categorized with a low belonging degree (C_L).

By the end of the calibration process, the color space is subdivided into subspaces, which are not necessary disjoint since the same color can belong to different subspaces, with different belonging degrees. The motivation for allowing non-disjoint subspaces is that teams frequently share colors, for example uniforms with white stripes are common and, thus, the exact same well known color belongs to the two opposing teams.

As will be seen later, the belonging degrees assigned to each color triplet will allow to break ties but also to generate dynamic subspaces that can adapt (either grow or shrink) during the game. Subspaces do not have, nor ever create, any predefined specific shape as they are created from user-selected seeds on the image and based on the frames' characteristics.

3.1.2 Background subtraction

Since the background is more or less static, due to the semi-controlled environment of an indoor game, the subtraction is performed using an empty image of the viewed scene recorded prior to the Portuguese SuperCup Competition and only the threshold used to distinguish between foreground and background pixels is allowed to vary. This threshold is specific for each pixel.

Background subtraction is performed on the RGB color space, because tests showed that, for some pixels, a small difference between the RGB color components of the background and the processed images corresponded to a large difference on the Hue component (HSL color space). In fact, non-linear color spaces suffer from the non removable singularity problem as stated by [8].

In addition, to make the processing time shorter, the subtraction is executed locally and not to the entire image. In other words, only predefined regions, which are defined by the Kalman Filter predictive stage, suffer this process (Sect. 3.2).

The threshold applied to each pixel is only updated if the pixel is classified as background, otherwise its value remains unchanged. The update obeys to Eq. 1 and the value is never allowed to go below 4 % or above 23.5 % of the entire color range (0–255) for each color component (these values were obtained experimentally by trial and error).

$$\sigma^c_{t+1}(x, y) = \begin{cases} \alpha(I^c_t(x, y) - B^c(x, y)) + (1-\alpha)\,\sigma^c_t(x, y), \\ \qquad \text{if } I_t(x, y) \in B(x, y) \\ \sigma^c_t(x, y), \quad \text{otherwise} \end{cases} \tag{1}$$

where

- σ^c_{t+1} is the threshold of the pixel at position (x, y), time t+1 and color component c,
- I^c_t is the color intensity of the pixel at position (x, y), time t and color component c,
- B_c is the background color intensity of the pixel at position (x, y) and color component c, and
- α is a learning constant, that for our specific case was set to 0.02.

Pixels whose color difference from the background image is less than the respective threshold are labelled as background, the others are labelled as foreground.

3.1.3 Team identification

After the foreground pixels are identified, their color is compared against the color lookup table that resulted from the

Table 2 Fuzzy inference associative matrix

$S_c(x,y) =$		μ_{S_B}		
		0	0.5	1
μ_{S_A}	0	ϕ	S_B	S_B
	0.5	S_A	$\begin{cases} \frac{n_A}{n_B} > 1+\delta \rightarrow S_A \\ \frac{n_A}{n_B} < 1-\delta \rightarrow S_B \\ else \rightarrow \phi \end{cases}$	$\begin{cases} \frac{n_A}{n_B} \gg 1 \rightarrow S_A \\ else \rightarrow S_B \end{cases}$
	1	S_A	$\begin{cases} \frac{n_B}{n_A} \gg 1 \rightarrow S_B \\ else \rightarrow S_A \end{cases}$	$\begin{cases} \frac{n_A}{n_B} > 1+\delta \rightarrow S_A \\ \frac{n_A}{n_B} < 1-\delta \rightarrow S_B \\ else \rightarrow \phi \end{cases}$

calibration process (Sect. 3.1.1) and then classified into one of the subspaces.

Since the same color can belong to different subspaces it may occur that a pixel is classified into more than one subspace. To obtain a crisp value of the team the pixel belongs to (S_c), the Fuzzy inference model uses not only the belonging degree itself (μ), but also information about adjacent pixels that have already been classified, or more precisely the proportion of pixels belonging to each subspace ($\frac{n_A}{n_B}$) according to the inference associative matrix defined on Table 2 (δ is a small constant defined by the user and Φ is the empty set).

Using this Fuzzy inference model it is possible that, although the belonging degree of a pixel to a subspace based on the color calibration information is higher than the belonging degree to the other subspace, it may be the winner due to the neighbourhood characteristics.

Additionally, if the winning subspace has a full belong to that color triplet and corresponds to a seed color (C_S), then a region growing process is triggered, and the color lookup table that contains the information concerning the color subspaces is updated. This auto-expansion is more restrictive than the one performed during the manual initialization and is triggered at time intervals ($t_{\exp}$), that can be defined by the user.

For this update to add not only color triplets to the subspaces but also to remove them, each color triplet has associated a persistence ($p_{S_c}(R, G, B)$) to that subspace. Colors with lower belonging have lower persistence and colors with higher belonging have higher persistence. The initial persistence given to the color is proportional to the time between auto-expansions, according to Eq. 2.

$$\begin{cases} p_{S_c}(R,G,B) = C_{\text{Low}} \times t_{\exp}, & \text{if } B_{S_c}(R,G,B) = C_L \\ p_{S_c}(R,G,B) = C_{\text{Med}} \times t_{\exp}, & \text{if } B_{S_c}(R,G,B) = C_F \\ p_{S_c}(R,G,B) = C_{\text{High}} \times t_{\exp}, & \text{if } B_{S_c}(R,G,B) = C_S \end{cases} \tag{2}$$

The persistence is maximum, with the values defined in Eq. 2, when the color is added to the subspace and diminishes whenever it is not detected in a frame. When the persistence value reaches zero, the color triplet is removed from the subspace.

With the introduction of this dynamical behavior (the update of the look up table), it is possible to have mutable subspaces that adapt to lighting changes, either occurring at different regions of the same frame or between frames.

At the same time the foreground pixels are classified, they are also aggregated horizontally to form Run Length Encoding (RLE) structures characterized by the y, $x_{\min}$ and $x_{\max}$ positions of the RLE. An outline of the algorithm to generate these RLE structures ([30]) is presented next.

Algorithm 1 *findRLEs*

```
1:  /* Searches the input ROI and returns the existing RLEs */
2:  for $y$ in ROI do
3:      for $x$ in ROI do
4:          $curSubspaceN \leftarrow LUT(C(x,y))$
5:          if $nonEmpty(curSubspaceN)$ then
6:              $TmpRLE \leftarrow RLE(curSubspaceN, x, y)$
7:              $TmpRLE \leftarrow fillRLEEnd(TmpRLE, ROI)$
8:              if $valid(TmpRLE)$ then
9:                  $RLEs.add(TmpRLE)$
10:                 $x \leftarrow TmpRLE.xEnd$
11:             end if
12:         end if
13:     end for
14: end for
```

The *fillRLEEnd* function is described using Algorithm 2.

Algorithm 2 *fillRLEEnd*

```
1:  /* Accepts an incomplete RLE with the start of the segment
       and fills-in the end X coordinate in the returning RLE */
2:  for $x \leftarrow RLE.xInit$ to $ROI.xMax$ do
3:      if $LUT(C(x, RLE.y)) = RLE.subspaceN$ then
4:          $xLast \leftarrow x$
5:      else
6:          $xGap \leftarrow x$
7:          if $(xGap - xLast) > interRLEGap$ then
8:              break
9:          end if
10:     end if
11: end for
12: if $(xLast - RLE.xInit) > minRLESize$ then
13:     $RLE.xEnd \leftarrow xLast$
14: else
15:     $setInvalid(RLE.xEnd)$
16: end if
17: return $RLE$
```

As shown in Algorithm 2, if the distance between two RLEs is small and they belong to the same color subspace (line 12), they are considered as being part of the same RLE and connected together.

Finally, the RLEs are merged vertically to form blobs. The blobs resulting from this pixel aggregation are further refined, according to size and color density constraints. Therefore, blobs that are too small or too large or blobs that have low color density are discarded as being players. The color density is measured as the percentage of pixels inside the rectangular bounding box of the blob that belong to the subspace divided by the total number of pixels. The remaining blobs are considered players that belong to a given subspace (team) (S_c) and have an (x, y) position on image and world coordinates.

The position on image coordinates is calculated as the blob's center of mass, according to Eq. 3. This is an weighted center of mass calculation because it uses the belonging degrees defined on Table 1. This way, pixels that are seeds or fully belong to the subspace (C_S or C_F) have a higher contribution to the final result.

$$(x_{\mathrm{cm}}, y_{\mathrm{cm}}) = \left(\frac{\sum_x \sum_y \mu_{\mathrm{Sc}}(C(x, y))x}{\sum_x \sum_y \mu_{\mathrm{Sc}}(C(x, y))}, \frac{\sum_x \sum_y \mu_{\mathrm{Sc}}(C(x, y))y}{\sum_x \sum_y \mu_{\mathrm{Sc}}(C(x, y))} \right) \quad (3)$$

where

- c is the team the blob belongs to,
- $C(x, y)$ is the color of pixel at location (x, y), and
- μ_{Sc} is the Fuzzy belonging degree assigned during the color calibration phase

The world coordinates are obtained by first removing the barrel effect produced by the lens (only radial effect was considered, since the tangential component was found to be insignificant) using Eq. 4. The unknowns in this equation system (k_1, k_2, k_3, x_c and y_c) are determined using the information extracted from the field lines.

$$\begin{cases} x_u = x_d + (x_d - x_c)(k_1 r^2 + k_2 r^4 + k_3 r^6) \\ y_u = y_d + (y_d - y_c)(k_1 r^2 + k_2 r^4 + k_3 r^6) \end{cases} \quad (4)$$

where

- $r^2 = (x_d - x_c)^2 + (y_d - y_c)^2$,
- (x_u, y_u) are the undistorted coordinates,
- (x_d, y_d) are the distorted coordinates,
- (x_c, y_c) are the coordinates of the center of distortion of the lens, and
- k_1, k_2 and k_3 are the radial coefficients for barrel distortion.

Figure 3 illustrates the images before and after removing the barrel effect for the two cameras.

Once the barrel effect is removed from the images, it is possible to apply the pinhole camera model to obtain the world coordinates. This model uses intrinsic parameters *(K)* and extrinsic parameters *(R* and *T)* to map image coordinates *(X)* into world coordinates *(x)*, according to Eq. 5, a process known as homography.

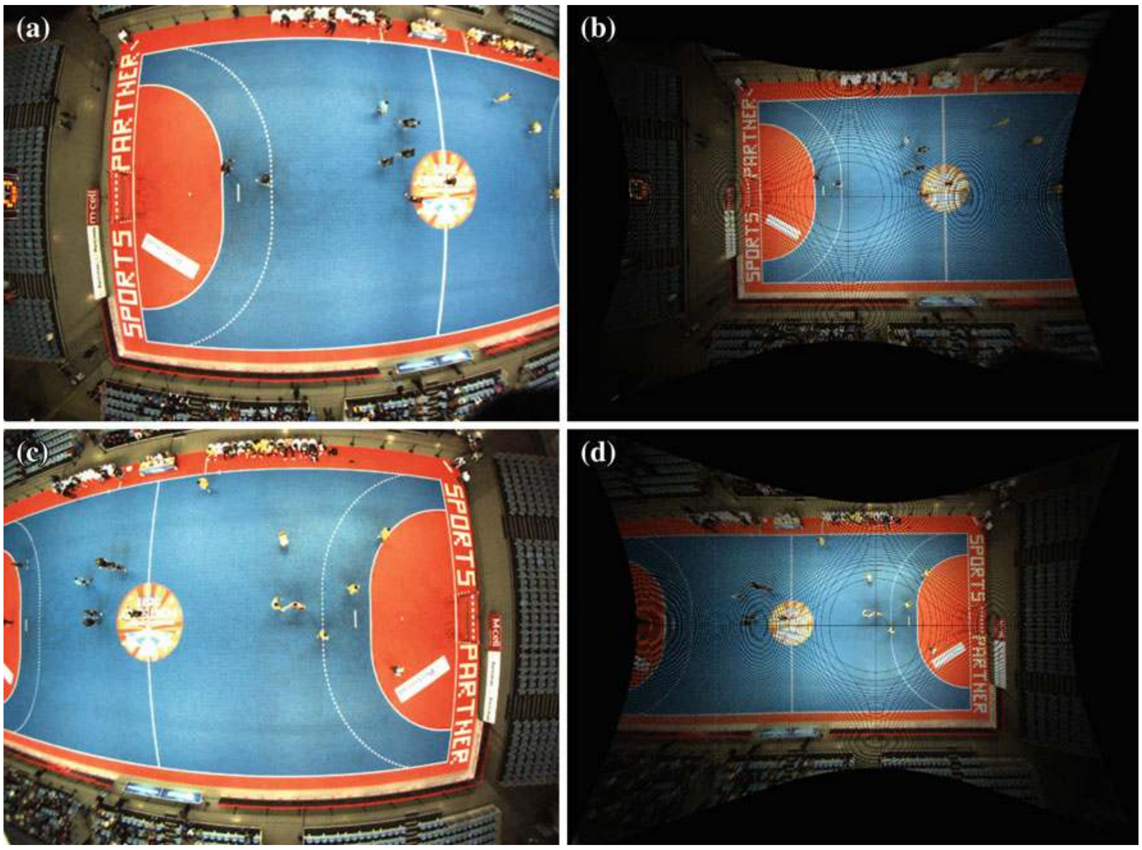


Fig. 3 **a** and **b** *left image* before and after removing the barrel effect distortion. **c** and **d** *right image* before and after removing the barrel effect

$$x = K[R|T]X \Leftrightarrow x = H_w X \quad (5)$$

The H matrix is defined according to Eq. 6

$$H_w = \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R_{T_{11}} & R_{T_{12}} & R_{T_{13}} & R_{T_{14}} \\ R_{T_{21}} & R_{T_{22}} & R_{T_{23}} & R_{T_{24}} \\ R_{T_{31}} & R_{T_{32}} & R_{T_{33}} & R_{T_{34}} \end{bmatrix} \quad (6)$$

where

- $R_{T_{11}} = \cos\phi\cos\alpha$
- $R_{T_{12}} = \sin\omega\sin\phi\cos\alpha - \cos\omega\sin\alpha$,
- $R_{T_{13}} = \cos\omega\sin\phi\cos\alpha + \sin\omega\sin\omega\sin\alpha$,
- $R_{T_{14}} = T_x$,
- $R_{T_{21}} = \cos\phi\sin\alpha$,
- $R_{T_{22}} = \sin\omega\sin\phi\sin\alpha + \cos\omega\cos\alpha$,
- $R_{T_{23}} = \cos\omega\sin\phi\sin\alpha - \sin\omega\cos\alpha$,
- $R_{T_{24}} = T_y$,
- $R_{T_{31}} = -\sin\phi$,
- $R_{T_{32}} = \sin\omega\cos\phi$,
- $R_{T_{33}} = \cos\omega\cos\phi$,
- $R_{T_{34}} = T_z$,
- f is the focal length,
- c_x and c_y are the coordinates of the optical center,
- ϕ, ω and α are the rotations around the x, y and z axes respectively, and
- T_x, T_y and T_z are the translations on x, y and z directions.

Since we are only interested in the players center of mass position, only these coordinates are converted and not the entire field. In addition, these coordinates are projected at an average player height, which allows a more correct measure of the players' positions and enables better information fusion between cameras, that will be crucial for the tracking methodology when a player passes from one camera to the other.

3.2 Player tracking

Player tracking is based on a vector of Kalman filters [17,34] (one per player) with state x_k (Eq. 7), measure z_k (Eq. 8) and input u_k (Eq. 9) at instant time k.

$$x_k = [xy]^T \quad (7)$$

$$z_k = [xy]^T \quad (8)$$

$$u_k = [v_x v_y]^T \quad (9)$$

where

- x and y are the player's center of mass position in real-world coordinates and
- v_x and v_y are the player's velocity in real-world coordinates.

And modelled according to the following linear stochastic difference equations (Eqs. 10, 11).

$$x_k = Ax_{k-1} + Bu_k + w_{k-1} \quad (10)$$

$$z_k = Hx_k + v_k \quad (11)$$

where A represents the state model matrix, B the control input model and H is the observation model matrix. These matrices correspond to an identity matrix of size 2×2. The random variables w_k and v_k represent the process and measurement noise.

The input to the system (u_k) is determined based on the players subsequent positions according to Eq. 12.

$$v_k(x, y) = \left(\frac{x_k - x_{k-1}}{\Delta t}, \frac{y_k - y_{k-1}}{\Delta t}\right) \quad (12)$$

The usage of real-world coordinates allows a transparent tracking between the two video streams. Moreover, in the overlapped region, two measures can be extracted from both images. The Kalman filter deals with these two measures in a straightforward way because they are in real-world coordinates.

Whenever the user indicates a player (with the mouse), a new Kalman filter is added to the vector with the player's real world position and a default velocity of 0 m/s. Afterwards, the players' locations on the subsequent frames are predicted using Eq. 10. The area around the predicted measure corresponds to a region of interest (ROI) that is searched, according to the process explained on Sect. 3.1.3, to generate a measure (z_k) to update the estimate.

By predicting the position of the players on the subsequent frames it is possible to reduce the computational cost because only a few regions of the entire image are searched for players.

For the cases when the tracking is lost beyond a given configurable threshold (called Tracking Prediction Window-TPW), the system prompts the user to locate the player on the field. If the subsequent tracking is successful, no further user actions are required and current tracking is linked to previous history as a normal result of the Kalman Filter. The user may also signal other special game circumstances that are not in the scope of this article.

4 Results

In order to validate the approach, the system was mounted at a public sports hall to capture the official games of the Portuguese Handball SuperCup, 2011. The video footage collected supports the proposed approach since the entire field was covered with good resolution and a large overlapped zone as shown in Fig. 2.

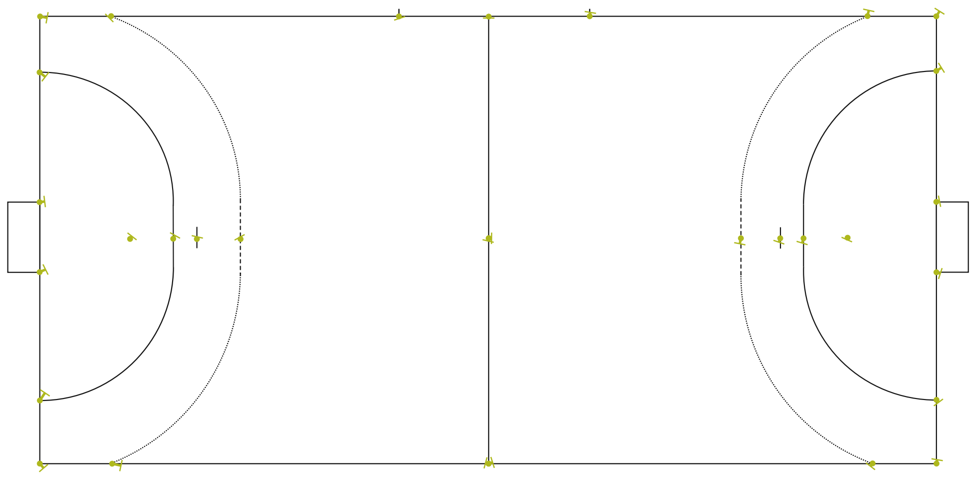


Fig. 4 Measure of *error: dots* represent real coordinates and *bars* represent the measure obtained using the homography estimated

The presented results relate to system's accuracy, player detection and tracking rates. Also, an in-depth sensitivity analysis was performed to evaluate the robustness of the approach. The last subsection illustrates how the resulting information can be visualized by the user.

4.1 Processing time

The ultimate goal of this work would be to make the system work in real time. Results indicate that the system takes, on average, about 160ms to process each frame using an Intel Core i7-2630QM@(2.00–2.90) GHz computer operating on Windows7 and can go up to 1s during an auto-expansion process.

In order to obtain real-time processing, a speed-up of around 5 times would be required, since a frame should be processed in less than 33 ms (cameras operate at 30fps), excluding auto-expansions. Although our algorithm is prepared to be optimized with strategies that promise to deliver such speed-ups (GPU aided processing, parallel and/or distributed processing or even smart cameras), they are beyond the scope of this paper.

4.2 Measurement accuracy

Tests conducted on the accuracy of the measure show that the error is less than 35 cm for the 2 cameras for the total of 20×40 m. This error is comparable with other works devoted to handball [3] (7–28 cm) and [21] (30–50 cm). The following image (Fig. 4) illustrates the errors at known points of the field. The green dots indicate the real positions, while the bars represent the distances to the measures obtained with the calculated homographies.

The error obtained is satisfactory given the characteristics of the implemented system: high area to be covered, the distortion induced by the used wide angle lenses and the system's inherent resolution (3.6, 3.7 cm). Additionally, the maximum errors are achieved in the field periphery and not in critical areas for the game.

4.3 Player detection

In order to validate the Fuzzy methodology, two distinct teams that we named green and red teams (examples of both teams can be found in Fig. 5b) were calibrated as explained on Sect. 3.1.1. The original seeds were selected by clicking on the players of both teams, which resulted on the initial color subspaces illustrated in Fig. 5a.

After 1020 frames and $t_{\text{exp}} = 30$ (which corresponds to 34 expansions), it is possible to confirm that the teams' color subspaces have updated and dynamically changed from the original color subspaces (Fig. 5a) into the new color subspaces of Fig. 6. It is convenient to disable the auto-expansion process when the color subspaces become stable and the detection rates high because the overhead time induced by the auto-expansion is high and can slow down the processing time.

Graphs in Fig. 7 provide an overview of the color spaces evolution during the auto-calibration process. It is possible

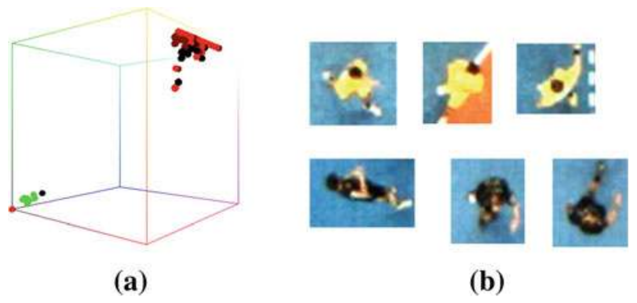


Fig. 5 **a** Initial color subspaces for *green* and *red* teams. *Lighter dots* are seed colors (C_S), intermediate are team color (C_F) and the darkest resemble the team color (C_L). **b** Examples of players from *red team* (*up*) and *green team* (*down*) (color figure online)

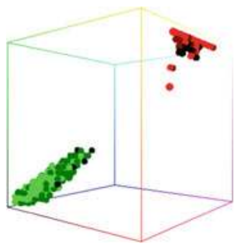
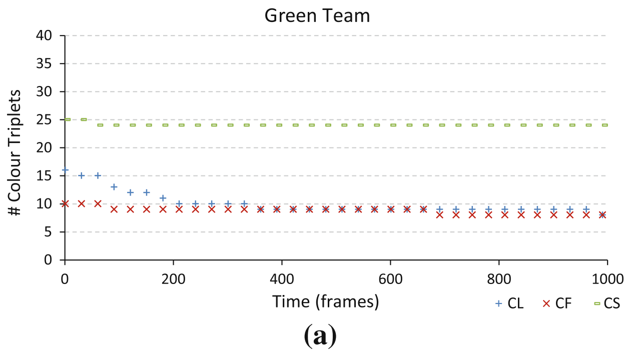


Fig. 6 Final color subspaces after 34 auto-expansions (color figure online)



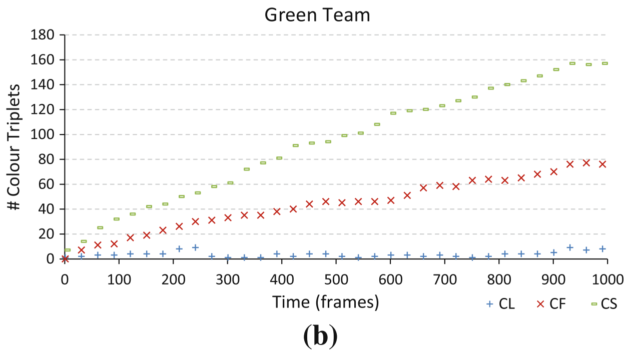
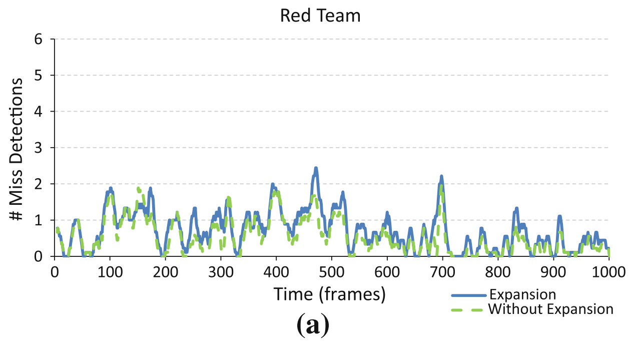


Fig. 7 Evolution of the number of color triplets that belong to each team and the respective belonging degree. The expansion is performed every 30 frames ($t_{\text{exp}} = 30$) (color figure online)



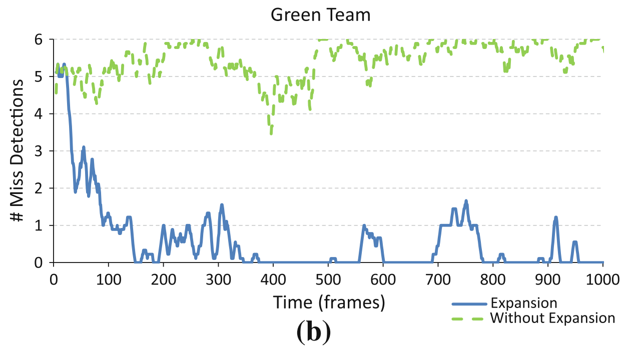


Fig. 8 Number of miss-detected field players in each frame with and without color auto-expansion: **a** *red team*, **b** *green team*. Each point on the graphs was obtained by passing an averaging filter of size 9 to the original points (color figure online)

to verify that, from the initial color subspaces (Fig. 5a) to the final ones (Fig. 6), the color triplets that belong to each class are adapting.

On the red team, the initial color subspace defined by the user consisted of color triplets that in fact do not belong to it. By making use of the persistence the auto-expansion process adaptively "pruned" those color triplets from the subspace which resulted in a slight more condensed and stable form.

On the other hand, the initial green team calibration did not reflect well the characteristics of the team's uniform. Therefore, it is possible to state that the color triplets that are seeds increase greatly during the auto-expansion, the C_F color triplets increase less due to their smaller persistence and color triplets that resemble the color C_L stay quite stable at low values.

From what has been said, it is important to highlight that the initial seed choice (a well known problem of region growing methods) as well as the specific characteristics of the team's uniform will influence on how well the color subspace adapts to the environment conditions. In fact, the initial seeds for the red team resulted in a faster adaptation: the color subspace adaptation is very quick at the initial process on pruning color triplets (C_F) that were miss categorized by the user. On the other hand, for the green team the stabilization seems to occur more at the end of the process.

Comparing the number of not detected players in each frame with and without the Fuzzy model of color expansion, it is possible to verify that the overall player detection achieves better results with the mutable color subspaces, as depicted in Fig. 8 (each handball team is composed of six field players).

The usage of the auto-expansion methodology greatly improves the detection rate for the green team, in fact, the number of miss-detected players per frame, after frame 100, sporadically gets higher than 1 but never higher than 2, and most of the time is 0, while with the initial color subspace continuously oscillates between 4 and 6.

For the red team the behavior is somewhat different, and the number of non-detections is similar for both cases because the initial color subspace already reflected the team characteristics well. An important aspect to highlight is that, despite some pruning, the auto-expansion process was able to maintain the color triplets that really belong to the team.

The results for the two teams indicate that the auto-expansion is extremely important and can greatly improve the results when the initial color subspace does not reflect

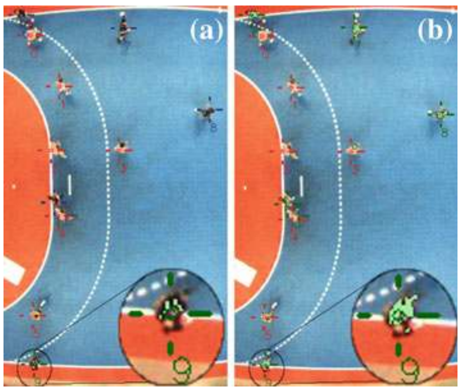


Fig. 9 Results of the player detection at frame 671: **a** without auto-expansion and **b** with auto-expansion. *Green* and *red* crosses indicate correct detection while *blue* crosses indicate the position was predicted by the Kalman filter (color figure online)

well the team characteristics. Furthermore, during the game if the light conditions change, the color subspace will also adapt to accommodate the new color triplets and remove the ones that no longer belong to the team.

Figure 9 shows a zoom around the 6 meter-line of the players' detection at frame 671, where the green/light red pixels correspond to pixels that were labelled as belonging to green/red team, the green/red crosses correspond to players detected from green/red team, while the blue crosses indicate that the player's position was predicted by the Kalman filter and, therefore, it was not detected. Analyzing the two images, it is possible to verify that, using the Fuzzy auto-expansion model, all players from both teams were detected (Fig. 9b), while using the initial color subspaces (Fig. 5a) only one player from the green team was detected (Fig. 9a). In addition, the detected area of the players is higher with the Fuzzy model (visible on player 9) which allows to have a better measure of the player's center of mass.

4.4 Sensitivity analysis

In order to better assess the proposed global methodology, systematic tests were carried out. A sensitivity analysis of the most relevant parameters—time between expansions (t_{exp}), background learning constant (α), tracking prediction window (TPW) and initial seeds choice—was performed. In addition, robustness tests to changes in brightness and image resolution were conducted.

4.4.1 Expansion time

As explained on Sects. 3.1.1 and 3.1.3, the evolution of the color subspaces is governed by the t_{exp}. This dependence is twofold, first because the auto-expansion process is triggered at intervals of t_{exp} and second the triplets persistence is also dependent on the t_{exp}, according to Eq. 2.

The choice of the t_{exp} plays an important role in the overall detection, because a value too low, while providing a fast response, will also increase the processing time since auto-expansions occur more often. On the other hand, higher time between expansions will make the system adaptation slower and a good calibration will not be achieved so fast. Figure 10 shows the miss-detection rates using different t_{exp} values. It is possible to verify that for lower values (15 and 30) the system quickly adapts, while for $t_{\text{exp}} = 60$ it takes around 400 frames to obtain similar rates.

The overall values indicate that, for this specific case, the best performance is achieved with $t_{\text{exp}} = 30$ which has a 6.95 % global average of miss detections. Similar values are 7.72 % for $t_{\text{exp}} = 15$, 8.13 % for $t_{\text{exp}} = 45$ and 8.09 % for $t_{\text{exp}} = 60$.

4.4.2 Illumination

The impact that illumination changes may have on the system's performance was tested by artificially changing the brightness of the video, similarly to what is expected to happen in a sudden illumination change. Figure 11 shows the miss-detection rates for the original behavior (plot "Original") compared with the ones obtained when the videos' brightness is changed by

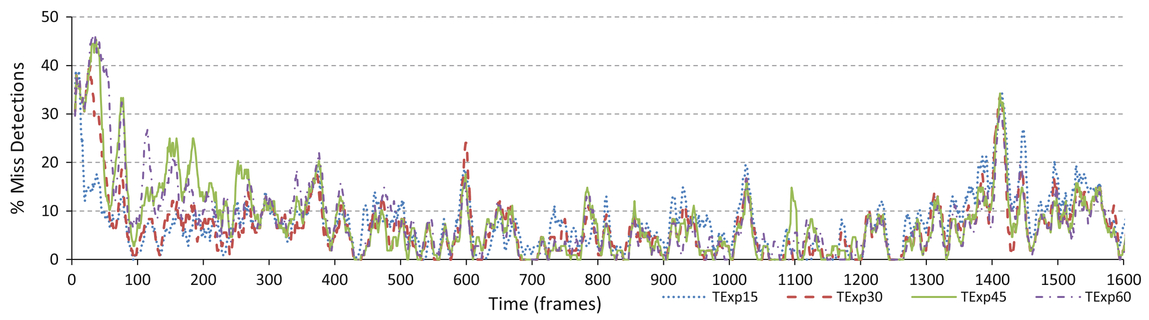


Fig. 10 Comparison of miss-detection rates using different t_{exp}: 15, 30, 45 and 60 frames (each point on the graphs was obtained by passing an averaging filter of size 9 to the original points)

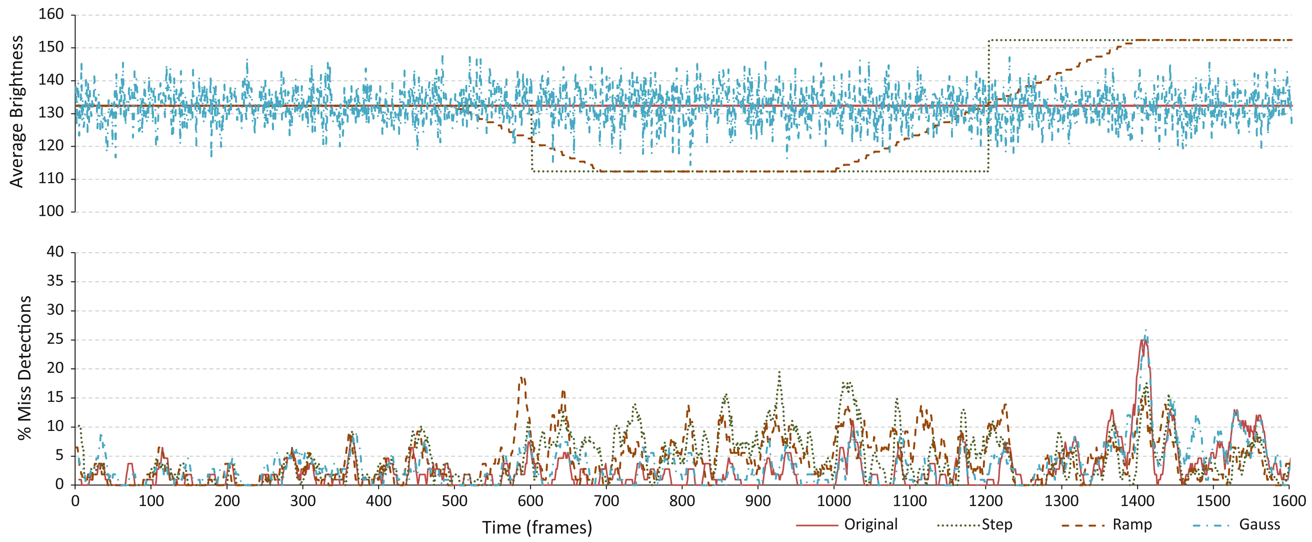


Fig. 11 Comparison of miss-detection rates (*bottom*) between the original video and video with artificially changed brightness. Original video and corresponding detection deficiencies are shown in the "Original" plots. Video with long-term added brightness step is shown on "Step" plots. Video with added ramp brightness is show on "Ramp" plots. Video with added noise of Gaussian distribution is shown on "Gauss" plot. Top graph indicates the average brightness of the video. Results were collected with $t_{\text{exp}} = 30$ and each point on the graphs was obtained by passing an averaging filter of size 9 to the original points

- applying a step of brightness of −20 at frame 600 and changing it to a step of +20 (above baseline) at frame 1200 (plot "Step")
- applying a ramp of brightness from frame 500 until frame 700 of −20 and a ramp from frame 1100 to 1400 of +40 (plot "Ramp")
- applying Gaussian noise at every video frame (plot "Gauss")

The visual difference between the brightness of the several test limits (−20, +20) and the original image can be seen in Fig. 12.

By analyzing the plots it is possible to verify that the system behaves well when Gaussian noise is applied. Although the miss-detection rate is slightly higher than that in the original case the overall performance is very similar. For the Step and Ramp tests, the miss-detection rate increases specially when applying the −20 brightness delta.

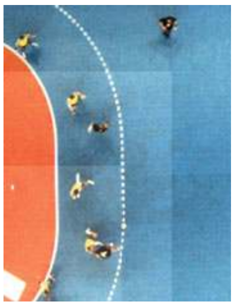
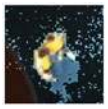
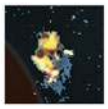
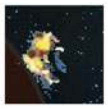
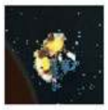
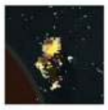
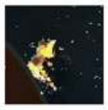
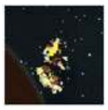
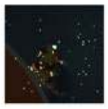
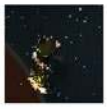
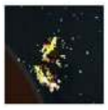
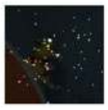
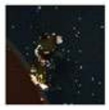


Fig. 12 Example of images obtained by applying a brightness step of −20 (*top*), 0 (*middle*) and +20 (*bottom*) to the original videos

4.4.3 Color learning constant

Concerning the background learning constant, the choice of its value must take into account that high values tend to induce fast absorption of foreground features into the background which causes miss detections, while low values make the background detection slower and, therefore, more pixels are considered foreground.

Table 3 Foreground detection using different learning constants ($\alpha = 0.02$, $\alpha = 0.08$, $\alpha = 0.16$ and $\alpha = 0.20$) at frames 31, 82 and 120. The pixels classified as background were darkened

		Frame Number		
		31	82	120
Learning Constant	0.02			
	0.08			
	0.16			
	0.20			

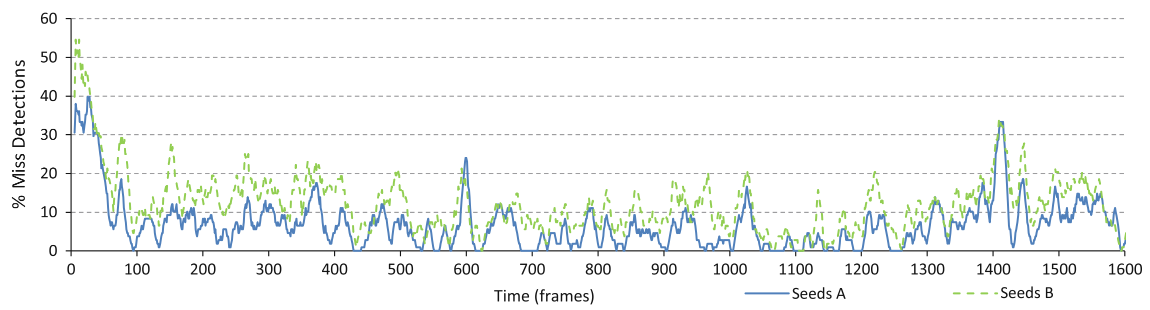


Fig. 13 Comparison of miss-detection rates between two different initial sets (Experiment A and Experiment B). All results were collected with $t_{\text{exp}} = 30$ and each point on the graphs was obtained by passing an averaging filter of size 9 to the original points

The following table (Table 3) illustrates how the background detection behaves when two players stay still for a quite long period of time using four different learning constant values (0.02, 0.08, 0.16 and 0.20).

High learning constants (0.16 and 0.20) rapidly absorb foreground features, and the two static players are miss detected from frame 82 onwards. A smaller constant (0.08) has a better behavior; however, on frame 82 only one of the players is detected, the other was included on the background. The smallest tested constant (0.02) proved to have better results, since it was able to completely absorb the players' shadows in a relatively short period of time (51 frames), while keeping the detection rates high.

4.4.4 Seed pixels choice

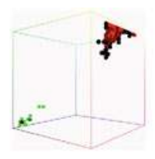
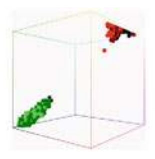
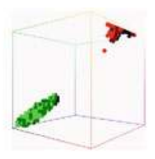
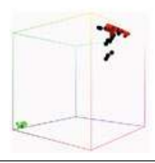
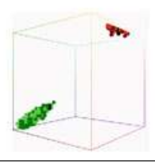
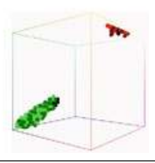
Another important aspect to take into consideration is the initial choice for seed pixels which impacts heavily on detection rates, as previously mentioned. Figure 13 illustrates how the detection rate may be influenced, depending on the initial color calibration. Therefore, Experiment B, due to a poorer choice of the initial seeds, presents higher miss-detection rates, particularly at the initial phase, only getting similar values after frame 600.

It is also possible to verify that the dynamic of the color subspaces behaves differently, as illustrated on Table 4. This difference is more noticeable for the red team, for which Experiment B results in a more condensed color subspace, compared with the one from Experiment A on frame 1600. For the green team, despite the differences on the initial subspaces the final ones are very similar.

4.4.5 Video resolution

A final sensitivity test on the detection methodology consisted on evaluating how the camera resolution would influence the miss-detection rate. The videos were down-sampled by a factor of two, and the resulting miss-detection rates can be seen on Fig. 14. For the down-sampled video, the parameters concerning the minimum area that allowed for a blob

Table 4 Color subspaces evolution depending on the initial seeds. Lighter dots are seed colors (C_S), intermediate are team color (C_F) and the darkest resemble the team color (C_L)

	Frame Number		
	0	800	1600
Experiment A			
Experiment B			

to be considered a player were adjusted accordingly. Down-sampling the video proved to induce a higher miss-detection rate, which is justified by the smaller number of pixels that composes each player, as can be seen in Fig. 14a.

4.4.6 Tracking prediction window (TPW)

Normal Kalman filtering transforms the sequence of detections into tracking. In case of missed detections, the system is able to make a limited prediction in time, to avoid user intervention. However, if a given player is not detected beyond a given configurable threshold called TPW, the user is prompted to locate the player or indicate a special game circumstance.

The TPW plays an important role in the tracking rate. Therefore, this experiment evaluated how the tracking rate is affected using TPWs of 1, 5, 10 and 20 frames. The results are shown in Fig. 15.

As can be seen, with the smallest TPW, which corresponds to 1, the tracker behaves like a pure filtered tracker, where the results are based solely on the detection and, therefore,

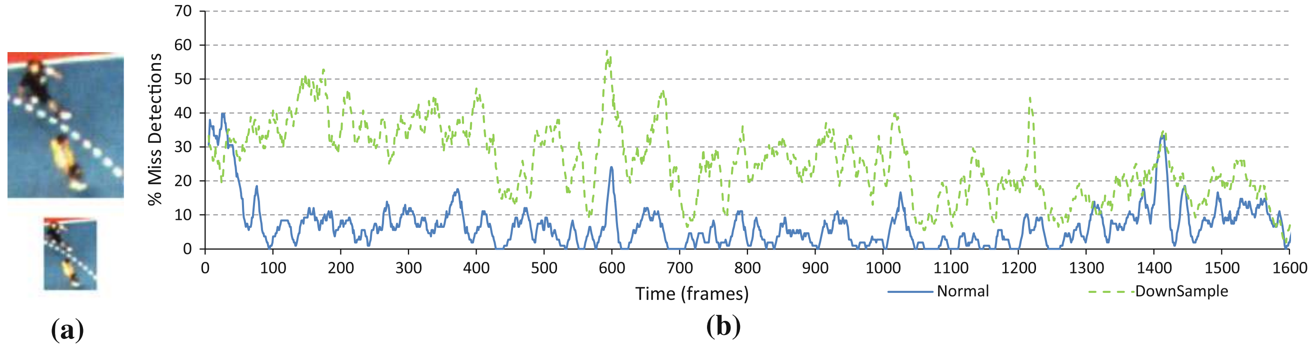


Fig. 14 Comparison of miss-detection rates using original image and down-sampled image with a scale factor of 1/2 (**b**) and examples of the two images (**a**). All results were collected with $t_{\mathrm{exp}} = 30$ and each point on the graphs was obtained by passing an averaging filter of size 9 to the original points

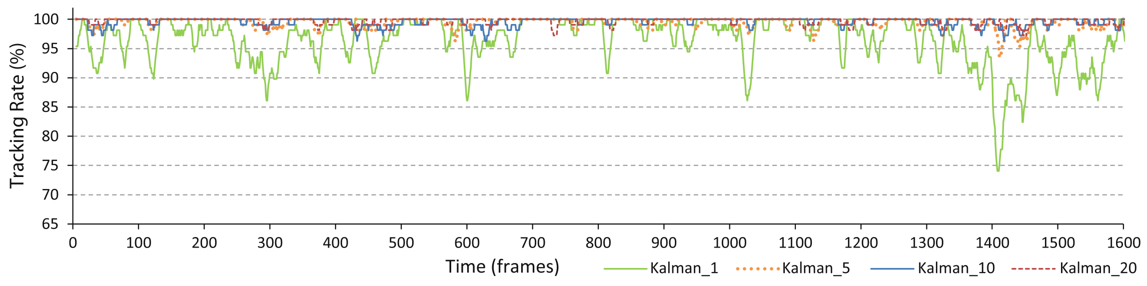


Fig. 15 Comparison of tracking rates using different temporal windows (1, 5, 10 and 20 frames) for the predictive stage of the Kalman Filter (TPW) (results were collected with $t_{\mathrm{exp}} = 30$ and each point on the graphs was obtained by passing an averaging filter of size 9 to the original points)

the tracking rate is lower (around 99.44 %). Increasing the TPW allows for higher tracking rates (around 99.70 % for TPW = 20 frames), because the user is prompted to correct the tracker less often. However, the operator must pay more attention to the entire process, otherwise the tracker may be lost and continue to rely on a misleading prediction.

4.5 Detection with shared colors

The Fuzzy methodology enables a better auto-expansion process, and also the possibility of having one color belonging to more than one team. To test it, a game where two teams (team A and team B) have the color white on their uniforms was used. The color subspaces are shown in Fig. 16a and the resulting process in Fig. 16b.

Despite the fact that the color white is common to both uniforms (shown by the color yellow triplet on the color subspaces Fig. 16a), the processing is able to identify the neighbouring pixels and correctly label the pixel under analysis. So for player 7 from team A, the white pixels are labelled as belonging to team A, because they are near red pixels that only belong to the team A, while for players 2 and 3, most of the white pixels are labelled as belonging to team B, due to their neighborhood to blue pixels that only belong to team B’s color subspace.

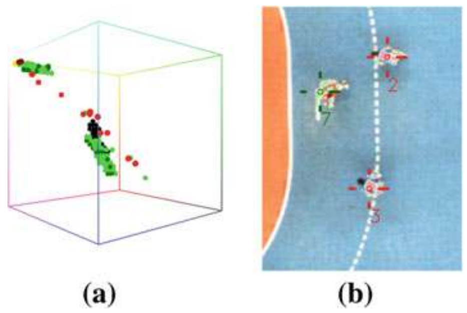


Fig. 16 **a** Color subspaces for two teams with common color (*white*) and the **b** corresponding image processing (color figure online)

4.6 Player tracking

The miss detections achieved with the auto-calibration process are not consistent and persistent with time, so they are compensated by the predictive stage of the Kalman filter (Eq. 11), and the benefits of the method are evident on the results obtained for the green team.

Taking into account the results of Sect. 4.4.6, tests were performed with TPW = 5 frames, which allows not only a good tracking rate, but also an accurate measure, because in case the tracker is lost, this fact will be highlighted to the operator sooner.

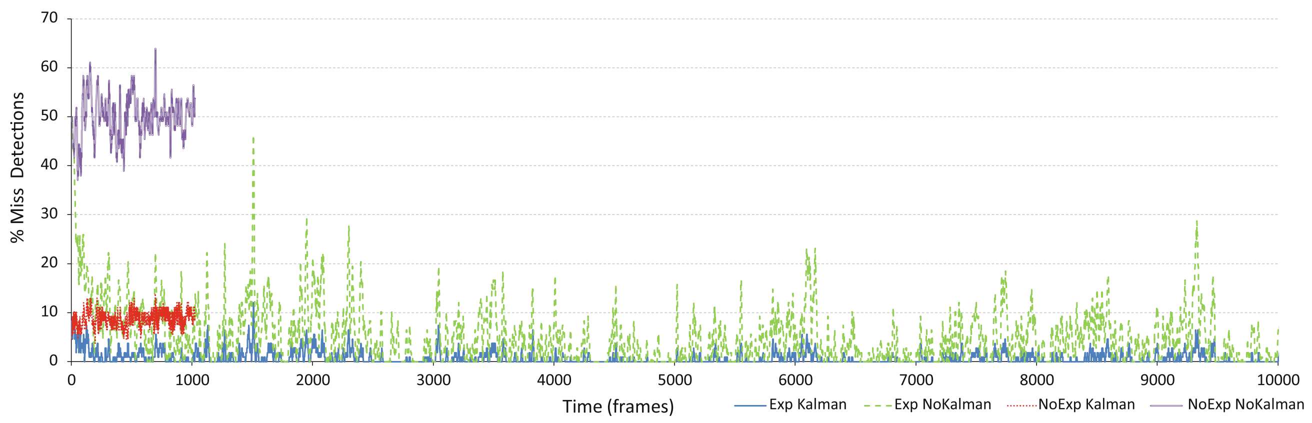


Fig. 17 Comparison of miss-detection rates with different setups: without auto-calibration and without Kalman filter (noExp noKalman); with auto-calibration and without Kalman filter (Exp noKalman); without auto-calibration and with Kalman filter (noExp Kalman); with auto-calibration and with Kalman filter (Exp Kalman). The Kalman filters used a $TPW = 5$ frames and each point on the graphs was obtained by passing an averaging filter of size 9 to the original points

Figure 17 compares the miss-detection rates with and without the auto-calibration as well as with or without the Kalman filter. As expected, the worst case is when no Kalman filter nor auto-calibration is performed. The usage of the auto-calibration method greatly improves the overall detection rates, which are further enhanced with the aid of the Kalman filter. Results for the non auto-calibration case are only plotted until frame 1000, because the human operator effort of correcting the tracking is very high, and more data do not provide extra information for what we are demonstrating.

Numerical results for each player on the tracking rate during the auto-calibration process can be found on Table 5 (these data were obtained by manual validation performed by an expert during the tracking process, and combine miss detections resulting from the TPW parameter, as well as corrections made by the operator in case the prediction was wrong). As expected, despite some miss detections, the tracking achieved very good rates, having a success that ranges from 95.44 to 99.90 % (and a corresponding average of 98.79 %).

When compared with other approaches, these results prove to be similar to the ones obtained with other methodologies, such as the fast Rao-Blackwellized Resampling particle filter proposed by [4] (>90 %), the Condensation particle filter proposed by [21] (ranging from 99.12 to 99.57 %) or directed weighted graphs [26] (around 93.26 %).

Table 5 Tracking rates of all the players (6 field players per team) during 11000 frames analyzed (≈370 s)

Player ID	Rate (%)	Player ID	Rate (%)
P_0	99.44	P_6	99.78
P_1	99.61	P_7	99.62
P_2	97.87	P_8	98.83
P_3	96.95	P_9	99.90
P_4	99.32	P_{10}	99.67
P_5	99.04	P_{11}	95.44

4.7 Data visualization

The log file, generated during the processing of the 11000 frames, contains the players' positions and velocities and, as explained in chapter 3, can be used by the Visualizer not only to see the two images fused into a single image with the tracked players highlighted, but also to extract statistics of the players' behavior. The image fusion (Fig. 18) is performed by first converting each pixel coordinate into real-world coordinates and then converting back to a common coordinate system (in this case the right image coordinate system).

In addition, the application provides statistics for each player, position and velocity maps per player and the team tactical map.

Table 6 shows the statistics for the 12 field players that started the game (it is important to mention that handball

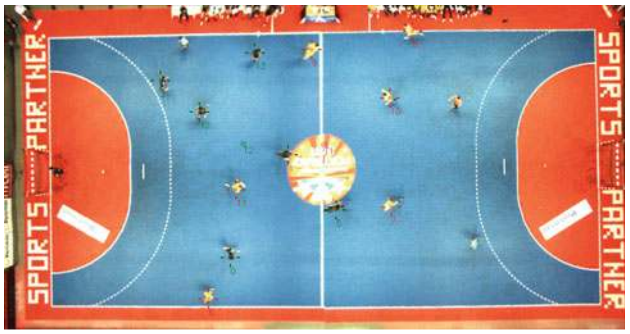


Fig. 18 Image provided by the visualizing application. Crosses above players indicate the player ID, and the two concentric *circles* with Tx indicate the *center* of mass of the respective team

Table 6 Player statistics

T	P	D (m)	T_i (s)	T_e (s)	Avg vel (m/s)	Max vel (m/s)
A	P_0	569.64	0	367	1.51	7.15
	P_1	600.21	0	367	1.59	7.44
	P_2	71.36	0	42	1.55	4.92
	P_3	106.09	0	56	1.67	9.41
	P_4	589.43	0	367	1.54	6.55
	P_5	570.29	0	367	1.49	7.15
	P_{12}	78.30	42	72	2.34	6.43
	P_{14}	58.25	56	73	3.25	6.75
	P_{15}	61.15	72	106	1.62	4.69
	P_{16}	185.31	73	186	1.58	6.78
B	P_6	645.93	0	367	1.74	8.34
	P_7	135.61	0	79	1.67	5.84
	P_8	165.65	0	73	2.10	6.64
	P_9	585.69	0	367	1.58	7.56
	P_{10}	539.65	0	367	1.43	5.61
	P_{11}	148.82	0	50	2.10	6.47
	P_{13}	135.27	51	108	2.09	5.95
	P_{17}	209.61	73	178	1.91	7.62

is a game that allows unlimited substitutions and, therefore, players are constantly being replaced). From the table it is possible to verify that

- for the same amount of time, players P_1 and P_6 have covered more distance (above 600 m) than players P_0 or P_{10},
- the average velocities of the players ranged from 1.43 m/s (player P_{10}) to 2.11 m/s (player P_{11}) and
- the maximum instantaneous velocity ranged from 4.92 m/s (player P_2) to 9.41 m/s (player P_3).

The field areas as well as the velocity maps for players P_5 and P_6 can be seen in Fig. 19.

Analysing these maps, we see that player P_5 plays more on the downside of the field (Fig. 19a), while player P_6 plays more on the upper side of the field and seems to have made a dynamic swap of tactical position because he also plays on the center of the field (Fig. 19b). Also, players tend to have higher velocities (represented by the color blue) during transition phases (when players change from an attacking position into a defending one or vice versa).

Using the team's tactical maps (Figs. 19, 20), it is also possible to verify where the teams preferentially occupy the field during specific situations. The Visualizer allows the user to interact and choose specific time frames to be displayed.

This statistical information can be very useful for sports experts since it allows them to have perception of the preferred field areas of each player as well as the effort spent during the game (distance travelled, average velocity and peaks of velocity).

5 Conclusions

This paper presented a cost-interesting system for tracking sports players in indoor games. Although the proposed system aims to be generic, only handball is currently addressed. It uses two cameras to retrieve high-quality video, and the system is somewhat portable among sports hall.

Player detection is based on an initial manual color calibration that is, during the processing stage, able to dynamically and automatically adapt to the light conditions that influence the color (different field zones, shadows, influence from outside conditions due to the presence of windows). The methodology adopted includes the identification of foreground pixels, using dynamic background subtraction, and the notion of team color subspaces, using a Fuzzy inspired dynamic model to detect players based on the color properties of their clothes. Due to the Fuzzy color classification, a given color may be shared among teams. Player tracking is further improved with Kalman Filtering.

In addition, the conversion of the players' image coordinates into real-world coordinates (measures errors below 35 cm) allows not only to have a transparent tracking of the players among cameras and to include more cameras in the system easily, but also to extract metrics of the players' performance (players' positions, velocity and distance traveled).

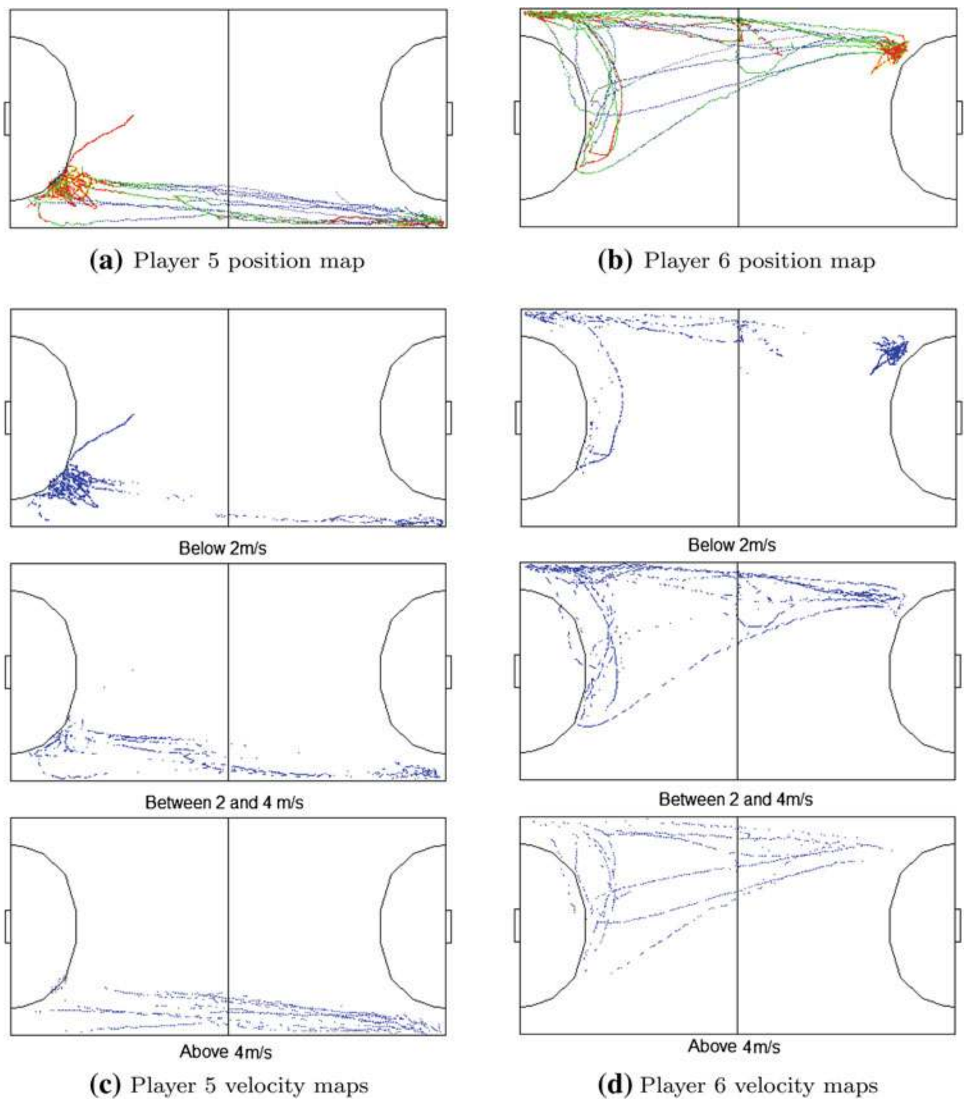


Fig. 19 **a** and **b** position maps for player P_5 and P_6, **c** and **d** velocity maps for player P_5 and P_6

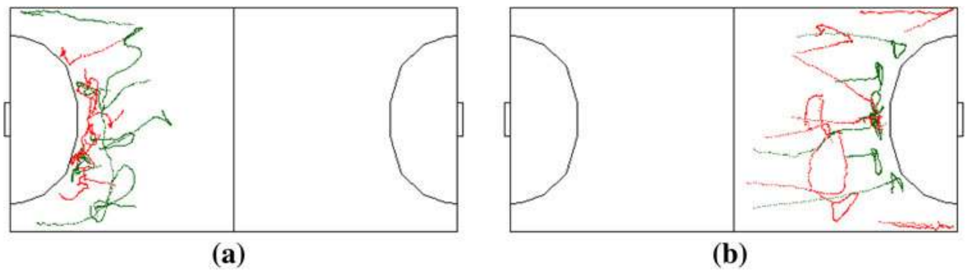


Fig. 20 Position maps for two game situations: **a** *red team* defending *green team* attacking, **b** *green team* defending *red team* attacking (color figure online)

The distinctive feature of the presented approach is the usage of a Fuzzy Logic-based color processing of the video stream, which allows not only to identify the teams but also to enable a dynamic behavior of the color subspaces that characterizes each team. It is expected that these simple and robust methodologies can be easily parallelized, so that, in future, the system can operate in real-time. Moreover, the visualization tool allows for a better understanding of the teams' behavior by providing a global, undistorted view of the field, as well as, schematic views concerning the players' movements and teams' interactions.

Results obtained with a video footage of a professional handball game (during the Portuguese Handball SuperCup competition, year 2011) validated the proposed system and indicate that it is possible to obtain high tracking rates (above 95 %) using simple clues, such as color and physical constraints aided by a robust tracking method (Kalman Filter). The usage of adaptive color subspaces generated by the Fuzzy inspired methodology allowed to better define the teams' color properties during the game and increased the overall detection rates, minimizing the user intervention.

An in-depth sensitivity analysis of the proposed methodology, which evaluated the relevant parameters (time between expansions, background learning constant, initial seeds choice and tracking prediction window) and the system's robustness to changes in brightness and image resolution, was performed. It was demonstrated that the t_{exp} value must be carefully chosen, so that a fast adaptation of the color subspaces is achieved without compromising the processing time, while the TPW value choice influences the tracking rate and the amount of user intervention. This analysis also proved the robustness of the methodology to lighting changes and, as expected, its downside dependence on the initial seeds choice (due to the region growing nature of the methodology).

As future work, it would be interesting to explore more belonging degrees on the Fuzzy inspired model, the possibility of having colors that belong to the subspace, but not labelled as seeds, could also trigger the auto-expansion process and the automatic enable/disable of the auto-expansion. Another aspect that is of much interest is the prospect of using the processing system's output not only to generate simple metrics, as described on this paper, but also to perform more high-level analysis which may include game tactics and game events. Long-term future work includes addressing other sports, such as volleyball or basketball.

Acknowledgments We would like to thank Fundação Calouste Gulbenkian by the support given through a PhD scholarship with ref. 104410, to Prof. Paulo Costa and Paulo Malheiros from the University of Porto, Faculty of Engineering for the specific camera calibration software and to Prof. Maria Luísa Estriga from the University of Porto, Faculty of Sports for providing the video footages of the games used for test.

References

1. Alahi, A., Boursier, Y., Jacques, L., Vandergheynst, P.: Sport players detection and tracking with a mixed network of planar and omnidirectional cameras. In: Distributed Smart Cameras, 2009. ICDSC 2009. Third ACM/IEEE International Conference on, pp. 1–8. IEEE (2009)
2. Barron, J.L., Thacker, N.A.: Tutorial : computing 2D and 3D optical flow. Biomed. Eng. (2004), 1–12 (2005). http://www.tina-vision.net/docs/memos/2004-012.pdf
3. Barros, R., Menezes, R., Russomanno, T., Misuta, M., Brandao, B., Figueroa, P., Leite, N., Goldenstein, S.: Measuring handball players trajectories using an automatically trained boosting algorithm. Comput. Methods Biomech. Biomed. Eng. **14**(1), 53–63 (2011)
4. Beetz, M., Gedikli, S., Bandouch, J., Kirchlechner, B., von Hoyningen-Huene, N., Perzylo, A.: Visually tracking football games based on TV broadcasts. In: Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI), pp. 2066–2071 (2007)
5. Beetz, M., von Hoyningen-Huene, N., Kirchlechner, B., Gedikli, S., Siles, F., Durus, M., Lames, M.: ASPOGAMO: automated sports game analysis models. Int. J. Comput. Sci. Sport **8**(1), 4–12 (2009)
6. Bogdanis, G., Ziagos, V., Anastasiadis, M., Maridaki, M.: Effects of two different short-term training programs on the physical and technical abilities of adolescent basketball players. J. Sci. Med. Sport **10**(2), 79–88 (2007)
7. Chapman, B., Jost, G., Van Der Pas, R.: Using OpenMP: Portable Shared Memory Parallel Programming, vol. 10. The MIT Press, Cambridge (2007)
8. Cheng, H., Jiang, X., Sun, Y., Wang, J.: Color image segmentation: advances and prospects. Pattern recognit. **34**(12), 2259–2281 (2001)
9. Delannay, D., Danhier, N., De Vleeschouwer, C.: Detection and recognition of sports (wo)men from multiple views. In: Distributed Smart Cameras, 2009. ICDSC 2009. Third ACM/IEEE International Conference on, pp. 1–7. IEEE (2009). doi:10.1109/ICDSC.2009.5289407
10. Deng, Y., Manjunath, B.: Unsupervised segmentation of color-texture regions in images and video. IEEE Trans. Pattern Anal. Mach. Intell. **23**(8), 800–810 (2001)
11. Figueroa, P., Leite, N., Barros, R.: Tracking soccer players aiming their kinematical motion analysis. Comput. Vis. Image Underst. **101**(2), 122–135 (2006)
12. Gedikli, S., Bandouch, J., Hoyningen-Huene, N.V., Kirchlechner, B., Beetz, M.: An adaptive vision system for tracking soccer players from variable camera settings. In: Proceedings of the 5th International Conference on Computer Vision Systems (2007)
13. Grimson, W., Stauffer, C., Romano, R., Lee, L.: Using adaptive tracking to classify and monitor activities in a site. In: Computer Vision and Pattern Recognition, 1998. Proceedings. 1998 IEEE Computer Society Conference on, pp. 22–29. IEEE (1998)
14. Heikkila, J., Silvén, O.: A real-time system for monitoring of cyclists and pedestrians. In: Visual Surveillance, 1999. Second IEEE Workshop on, (VS'99), pp. 74–81. IEEE (1999)
15. Hoyningen-Huene, N., Beetz, M.: Rao-blackwellized resampling particle filter for real-time player tracking in sports. Fourth Int. Conf. Comput. Vis. Theory Appl. (VISAPP) **1**, 464–471 (2009)
16. Hu, M., Chang, M., Wu, J., Chi, L.: Robust camera calibration and player tracking in broadcast basketball video. IEEE Trans. Multimed. **13**(2), 266–279 (2011)
17. Kalman, Rea: A new approach to linear filtering and prediction problems. J. Basic Eng. **82**(1), 35–45 (1960)
18. Kasiri-Bidhendi, S., Safabakhsh, R.: Effective tracking of the players and ball in indoor soccer games in the presence of occlusion. In: Computer Conference, 2009. CSICC 2009. 14th International CSI, pp. 524–529. IEEE (2009)
19. Koprinska, I., Carrato, S.: Temporal video segmentation: a survey. Signal Process. Image Commun. **16**(5), 477–500 (2001)
20. Kristan, M., Perš, J., Perše, M., Bon, M., Kovacic, S.: Multiple interacting targets tracking with application to team sports pp. 322–327 (2005)
21. Kristan, M., Perš, J., Perše, M., Kovačič, S.: Closed-world tracking of multiple interacting targets for indoor-sports applications. Comput. Vis. Image Underst. **113**(5), 598–611 (2009). doi:10.1016/j.cviu.2008.01.009
22. Lu, W., Ting, J., Murphy, K., Little, J.: Identifying players in broadcast sports videos using conditional random fields.In: IEEE Conf. Comput. Vis. Pattern Recognit. (2011)
23. Luebke, D., Humphreys, G.: How gpus work. Comput. Inf. Sci. **40**(2), 96–100 (2007)
24. Monier, E., Wilhelm, P., Ruckert, U.: Template matching based tracking of players in indoor team sports. 2009 Third ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC) pp. 1–6 (2009). doi:10.1109/ICDSC.2009.5289408

25. Needham, C., Boyle, R.: Tracking multiple sports players through occlusion, congestion and scale. In: British Machine Vision Conference, pp. 93–1022. BMVA (2001)
26. Pallavi, V., Mukherjee, J., Majumdar, A., Sural, S.: Graph-based multiplayer detection and tracking in broadcast soccer videos. IEEE Trans. Multimed. **10**(5), 794–805 (2008)
27. Perse, M., Kristan, M., Kovacic, S., Vuckovic, G., Pers, J.: A trajectory-based analysis of coordinated team activity in a basketball game. Comput. Vis. Image Underst. **113**(5), 612–621 (2009)
28. Ren, J., Xu, M., Orwell, Jones, G.A.: Multi-camera video surveillance for real-time analysis and reconstruction of soccer games. Mach. Vis. Appl. 21, 855–863 (2010). doi:10.1007/s00138-009-0212-0
29. Santiago, C., Sousa, A., Reis, L.: Pseudo fuzzy colour calibration for sport video segmentation. Comput. Vis. Med. Image Process. VipIMAGE **2011**, 351–366 (2011)
30. Santiago, C., Sousa, A., Reis, L.: Real time colour based player tracking in indoor sports. Comput. Vis. Med. Image Process. **19**, 17–35 (2011)
31. Spencer, M., Lawrence, S., Rechichi, C., Bishop, D., Dawson, B., Goodman, C.: Time-motion analysis of elite field hockey, with special reference to repeated-sprint activity. J. Sports Sci. **22**(9), 843–850 (2004)
32. Spencer, M., Rechichi, C., Lawrence, S., Dawson, B., Bishop, D., Goodman, C.: Time-motion analysis of elite field hockey during several games in succession: a tournament scenario. J. Sci. Med. Sport **8**(4), 382–391 (2005)
33. Tong, X., Liu, J., Wang, T., Zhang, Y.: Automatic player labeling, tracking and field registration and trajectory mapping in broadcast soccer video. ACM Trans. Intell. Syst. Technol. (TIST) **2**(2), 15 (2011)
34. Welch, G., Bishop, G.: An introduction to the kalman filter. Department of Computer Science University of North Carolina at Chapel Hill, EUA, North Carolina, Tech. rep. (2002)
35. Zadeh, L.A.: Fuzzy sets. Inf. Control **8**, 338–353 (1965)
36. Zhu, G., Xu, C., Huang, Q., Rui, Y., Jiang, S., Gao, W., Yao, H.: Event tactic analysis based on broadcast sports video. IEEE Trans. Multimed. **11**(1), 49–67 (2009)

Author Biographies

Catarina B. Santiago received her M.Sc. in Electronics and Computer Engineering in 2009 from the Faculty of Engineering of the University of Porto, Portugal. She is currently a Ph.D. student of the Doctoral Program in Informatics Engineering by the same university. Her main research interests include vision systems, data mining, artificial intelligence and Fuzzy logic.

Armando Sousa received his Ph.D. in Electrotechnical and Computer Engineering from the University of Porto in 2004. He is currently a “Professor Auxiliar” (lecturer) in the Faculty of Engineering of the University of Porto and integrated researcher in the “INESCTEC” associate laboratory. He received several international awards in robotic soccer under the RoboCup Federation (mainly in the small size league). He has co-authored over 50 international peer-reviewed publications and participated in over ten projects of several kinds. His main research interests are robotics, data fusion and vision systems.

Luis Paulo Reis is an associate professor at the University of Minho and a member of the Directive Board of LIACC, Portugal. He was the principal investigator of ten research projects on Artificial Intelligence and Robotics. He supervised 10 Ph.D. and 80 M.Sc. thesis and he is the author of more than 250 publications in international conferences/journals.