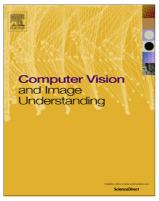
Computer Vision and Image Understanding 119 (2014) 102–115

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

journal homepage: www.elsevier.com/locate/cviu



Take your eyes off the ball: Improving ball-tracking by focusing on team play ☆

Xinchao Wang *, Vitaly Ablavsky, Horesh Ben Shitrit, Pascal Fua

Computer Vision Laboratory, École Polytechnique Fédérale de Lausanne (EPFL), EPFL/IC/ISIM/CVLab, Station 14, CH-1015 Lausanne, Switzerland

ARTICLE INFO

Article history:
Received 20 March 2013
Accepted 30 November 2013
Available online 12 December 2013

Keywords:
Tracking
Video analysis and event recognition
Occlusion reasoning
Behavior analysis
Sports video
Tracking the invisible

ABSTRACT

Accurate video-based ball tracking in team sports is important for automated game analysis, and has proven very difficult because the ball is often occluded by the players. In this paper, we propose a novel approach to addressing this issue by formulating the tracking in terms of deciding which player, if any, is in possession of the ball at any given time. This is very different from standard approaches that first attempt to track the ball and only then to assign possession. We will show that our method substantially increases performance when applied to long basketball and soccer sequences.

1. Introduction

Accurate ball tracking in sports is of tremendous importance to athletes, referees, coaches, and fans. However, the size and speed of the ball and prolonged occlusions make ball-tracking challenging even for trained human observers. Solutions for basketball or soccer have not gained acceptance, and a solution for hockey puck tracking [1] developed for a television broadcast network enjoyed only a brief lifespan.

Thus, a Computer Vision-based ball-tracking solution remains valuable and beyond the state of the art for most sports. Tennis is a rare exception because the ball is rarely occluded and its color is very different from that of the background, which makes tracking comparatively easy. It is far more difficult in team sports because the ball is often hidden by the players and follows an unpredictable trajectory as it is passed or taken from one player to another. Furthermore, the amount of image evidence for the ball, even when it is visible, is dwarfed by that for the players in its vicinity, as shown in Fig. 1. Frame-to-frame tracking is extremely unreliable in such cases. For example, even a state-of-the-art algorithm [2] that has been shown to be superior to many other state-of-the-art trackers for single object tracking and whose code is publicly available never tracks the ball for more than five consecutive frames in the sequences we present in this paper. Modern tracking-by-detection approaches that re-detect the object as often as necessary and aggregate results over several frames increase robustness but can still easily fail if the object is hard to detect in individual frames.

The contribution of this paper is to turn the common approach of first tracking the ball and then deciding which player is in possession of it on its head. In our approach, we first track the players and decide possession. We then use this as a means to achieve reliable ball tracking, which lets us turn unreliable image-based information into dependable trajectories. This is effective because the trajectory of the ball is intimately linked to that of the players who pass it to each other or steal it from one another. Therefore, exploiting this correlation allows for disambiguation and improved performance.

To this end we define a novel state space explicitly accounts for ball possession, a Conditional Random Field (CRF) [3] model that depends on the ball detections and the players' trajectories, and a practical approach to learning the model parameters from training data. More specifically, we start from video sequences from several synchronized cameras from which both the players' trajectories can be reliably extracted using publicly available software [4,5] and the ball ballistic trajectory can be also be extracted in consecutive frames when it is clearly visible. Between such frames, when individual players are in possession of the ball, its approximate position can be assumed to be their position up to the reach of their arms or legs. When the ball is passed, we take its path in the horizontal plane to be a straight line from one player to the next. As shown in Fig. 1, given perfect knowledge of ball possession as denoted by the "Oracle", this simple approach yields a relatively

☆ This paper has been recommended for acceptance by Nikos Paragios.
* Corresponding author.
E-mail addresses: xinchao.wang@epfl.ch (X. Wang), vitaly.ablavsky@epfl.ch (V. Ablavsky), horesh.benshitrit@epfl.ch (H.B. Shitrit), pascal.fua@epfl.ch (P. Fua).

1077-3142/$ - see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.cviu.2013.11.010

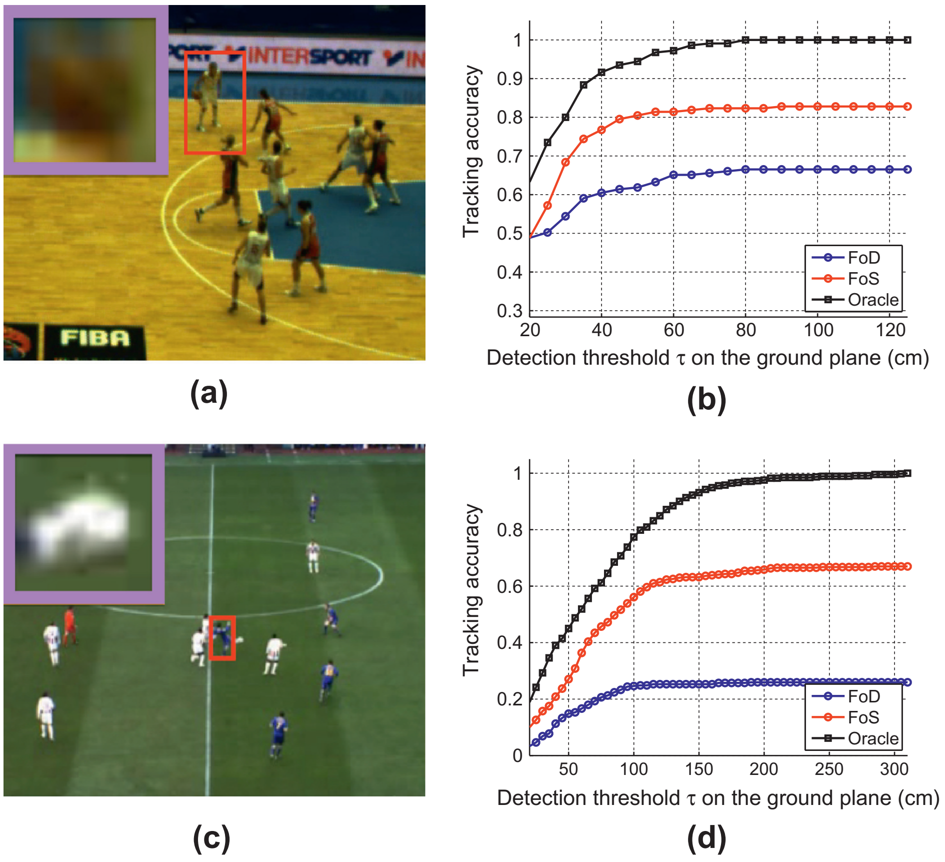


Fig. 1. Tracking the ball in team sports, such as basketball (a) and soccer (c) is hard due to the ball's small size, barely 12×12 pixels in the videos we work with as shown in upper left corner insets. Furthermore, it is subject to prolonged occlusions even when multiple views are available. Our key contribution is to overcome these difficulties by exploiting the contextual relation between the players and the ball, in particular with respect to *ball possession*. To illustrate this, we plot in (b) and (d) the accuracy we would obtain based on the sole knowledge of ball possession if it were given to us by an oracle, in this case a person looking at the videos. In other words, the "Oracle" tracker is the one with correct ball possessions, whose performance is the upper-bound of any tracker that is solely based on contextual information. We also plot the accuracy results of both a state-of-the-art approach that only looks at the ball and our attempt to establish ball possession without human input. The latter (red FoS curve) decisively outperforms the former (blue FoD curve) and approaches human performance levels (black Oracle curve). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

accurate estimate of the ball location and obtaining this knowledge therefore is what must be addressed. We formulate this problem in terms of assigning possession to a player, or to no-one when the ball is being passed, by minimizing a global objective function that takes into account players' positions and image information when the ball is visible.

We will show that this approach produces very reliable possession assignments and a final accuracy of the ball trajectory that is close to the best that can be expected over multiple attack phases and timeouts. This is in stark contrast to most other published approaches that have only been demonstrated on short sequences.

2. Related work

This section distinguishes two broad classes of approaches, those whose sole focus is on ball-tracking, and those that exploit spatio-temporal context to track a hard-to-discern target.

2.1. Tracking only the ball

We first discuss a handful of ball-tracking scenarios where the problem is considered solved, and off-the-shelf commercial solutions exist. Then we discuss the more broad class of problems, which are still beyond the reach of the current state of the art.

2.1.1. Ball tracking in the commercial world

Ball tracking has received considerable attention over the years. In some cases such as tennis, reliable commercial software is now available [6]. While the high velocity of the ball poses an engineering challenge, it is brightly colored and rarely occluded. Even then, achieving the required level of reliability and accuracy requires ten high-speed cameras looking at the scene from different angles.

The problem becomes substantially harder in team sports. For example in soccer, the ball can be surrounded by multiple players, who create occlusions in many views. This is probably why the soccer software sold by the company that developed the tennis ball tracking system discussed above [6] only aims at tracking the ball as it is being shot towards the goal, and therefore unoccluded.

In basketball, the problem is even harder because multiple players compete for the possession of the ball, generating even more occlusions. Since the basketball may be guided by hand, its trajectory is also more unpredictable. Lastly, the ball can be of similar color as the players' jerseys, adding yet another level of complexity. Even though some companies advertise the capabilities of post-game basketball trajectory analysis the amount of automation is unclear [7].

2.1.2. Ball tracking in the research world

Ball tracking has also been pursued in the academic world [8–10] before the advent of the commercial systems. For sports such as tennis or golf, it is now considered a solved problem. The same cannot be said of team sports for the reasons discussed above.

In many team sports, the trajectories of the ball and the players overlap in space-time. Therefore to track the ball one also has to localize the players. However, deriving a unified formulation for tracking the ball and the moving players is challenging. Typical approaches [11,12] to track the players and the ball independently relied on ad-hoc rules to decide if and when a player handled the ball. The composition and the parameters of these rules were defined manually rather than with respect to a clearly-defined objective function.

The fact that soccer players interact with the ball in a structured manner was exploited in [13]. A set of *ball phases* was defined—*in-possession*, *rolling*, *flying*—and for each such phase a physics-based motion model of the ball was introduced. Once the ball became un-occluded, and its phase was known, the appropriate motion model could be used for tracking. The *in-possession* phase was treated as a means to "initialize other phases", while in our work it is the main focus; thus the two approaches are complementary.

An approach to jointly track the soccer players and the ball was proposed in [14]. First, players were detected in each frame independently. Next, the ball was tracked across frames taking into account player detection. Last, player detections were linked into tracks, while taking into account the location of the soccer ball. Unfortunately, the ball-tracking formulation was not specified, except for a brief mention of an Adaboost classifier for scoring the ball trajectories. The validation was performed on sequences of 150 frames, spanning only a few seconds, making it hard to determine how useful the approach would be in practice.

Summary. Ball tracking in team sports in the presence of prolonged occlusions remains an unsolved problem. Ball-tracking approaches designed for individual or one-on-one games [8,9] only work for long passes and shots on the goal, while approaches that attempt to incorporate player trajectories [14,15] rely on ad-hoc rules for ball possession. Consequently, none of these approaches work on long video sequences. By contrast, our tracker minimizes a global objective function that not only takes into account image evidence about the ball but also players' location and the phase of the game.

2.2. Object-tracking and context

Oftentimes, a target that needs be detected or tracked is correlated in space and/or in time with its surroundings. We first discuss approaches to model complex individual and group behavior in ball-centric team sports, and then overview approaches that focus on a relation between a target and its local spatio-temporal context.

2.2.1. Exploiting the context of a ball-centric team sport

To the best of our knowledge, we are the first to develop an effective approach to exploit the highly complex context of a sports game for ball-tracking. In this section, we mention relevant work on analyzing the high-level state of a ball-centric game without the benefit of automatic ball tracking. We view the relation between the two sets of approaches as complimentary: many of those prior approaches to game analysis rely on a manual labeling of the ball and would therefore benefit from our formulation. Conversely, our end-to-end demonstration system requires an estimate of the phase of the game, which is obtained using the features of [16], and which we describe next.

Finding regions of player convergence. We first mention two approaches that rely on players' trajectories to estimate the regions of convergence. In some cases, these ground-plane regions happen to contain the ball.

In [15] the focus was on basketball. It was assumed that in an input video sequence it was known which team was attacking, and that this team was in the vicinity of the opponent's basket. Then, from the velocities of the players on the ground plane, *regions of convergence* were computed. It was assumed that each of these regions would be a candidate for the basketball location, and a Kalman-filter was used to *validate* ball-location candidates over time. Because of the strong assumptions on the content of the input video sequences, the algorithm could only be validated on sequences of up to 10 s in length. These test sequences are too short to fully understand the algorithm's failure modes and to determine how it could generalize to other basketball games. Furthermore, attempting to localize the basketball without taking into account image evidence cannot yield optimal results.

In [17], velocities of the soccer players in the ground plane were used to compute ground-plane regions where some interesting event could happen in the future. In some cases those regions happened to correspond to the future location of the ball, but in general it was not possible to attribute those regions to known elements of the game. Since the output of the algorithm could not be used to predict a sport-specific quantity, such as the location of the ball, the authors compared their regions of interests with the field of view selected by the "real camera operators" during professional games.

Recognizing elements of the game from player trajectories. A system for analyzing basketball games was described in [16]. The system comprised several components, including tracking, recognizing which team was attacking, which elements of the play were being executed, etc. Notably missing from the scope was tracking the basketball itself, which was explained in the paper by the "state of technology". However, the authors of [16] noted that knowing the ball location would "considerably improve the performance..." of their system.

Systems for analyzing hockey games were described in [18–20]. Similar to the system of [16], the system of [20] comprised components for player tracking, and the recognition of several elements of the play, e.g., *power play shot*. The location of the hockey puck was essential for their recognition algorithm, but the authors relied on the manual localization of the puck.

The approach to recognizing team-level activity in European handball was proposed in [21]. The set of six manually-specified labels included "slowly going into offense", "offense fast break", etc., and the features were based on players' on the player density function defined over the whole court (obtained by solving a Poisson equation). Notably, the location of the ball was not a part of the feature set, which precluded the recognition of "more complex activity classes".

An approach to recognize elements of a baseball game was presented in [22]. In that approach detections of the players in a video segment were automatically labeled with their roles, e.g., a pitcher, and a causal relations between actions were inferred. Recognition of social roles in field hockey, e.g., attacker, defender was proposed in [23]. The formulation integrated per-player location and role cues into a consistent interpretation via a conditional random field. Inference in such a model was computationally complex due to the combinatorial number of possible assignments, and furthermore a different model needed to be initialized depending on the user's query. The formulation was validated using "ground-truth person locations, as well as those using a simple automated detector"; it is unclear if in the complete system the ground-truth ball locations were utilized or if the ball location was ignored altogether.

Recognition of *plays* in American football was tackled in [24]. Their approach included a *temporal interaction matrix* defined with respect to the trajectories, and a manifold-based formulation for comparing such interaction matrices. The location of the ball was excluded from the formulation. The experiments were conducted with an assumption of "ground-truth data, where tracks for each player are manually labeled". Such an assumption does not hold true for our problem.

Summary. The approaches such as [15,16,21,23,24,17] would benefit from automatic ball tracking. An effective formulation for achieving this goal is the main contribution of our paper.

2.2.2. Exploiting the context of objects and parts

Models that exploit spatial and spatio-temporal context for object detection and tracking have been proposed in [25–27]. These models require a target to be surrounded by multiple moving objects or image patches. The appearance of these contextual objects should be sufficiently distinct so that they themselves can be reliably detected in novel video frames. To locate the target, each contextual object is employed as a predictor for its center.

In [25], which focused on vehicle tracking in aerial imagery, the model for a motion context captured "…an intuitive observation that the locomotive behavior of an object (e.g. car) provides information about locomotive behaviors of nearby objects (cars)". To identify a motion context among many moving objects, a measure of the chaos in a dynamical system was employed. Given the motion context, the target's location was implemented as a linear regression with respect to the motion of the contextual objects.

In [26] it was assumed that "relative position of feature and target is more or less fixed over short time intervals". The contextual features were derived from frames with "good visibility", and the presence of a feature in a novel frame was probabilistically estimated. Given the contextual features, the prediction of the target's location was implemented as generalized Hough transform.

The context model of [27] was developed for detecting a small body part, such as a hand, in a single image. In this model, the context comprised groups of image features, sequentially ordered into chains. All chains were constrained to originate from a single always-visible body part, such as a person's face.

Summary. While the above approaches have demonstrated the importance of context in tracking, they are not applicable to our problem. The initialization assumptions of [26] cannot be satisfied, and the requirement of [27] that an easy-to-detect anchor object is always visible, is inapplicable to our setup. The assumption of [25] and also of [26,27] that the target's context can be unambiguously re-acquired in a novel video frame does not hold, since at our image resolution body parts of players tend to look alike.

More importantly, the uncertainty in our context is fundamentally different. In our case, the spatio-temporal context is clear from the outset since the ball is always passed among a fixed set of players. The regression function to predict the target location from its context that was central to [25,26] is the identity since the player who is possession of the ball and the ball are co-located. The real challenge lies in deciding which player is indeed associated with the ball, and this challenge has not been resolved in prior work.

3. Formulation

Our formulation is general and is applicable to ball-centric team sports with a defined notion of ball possession. For the sake of simplicity and concreteness, we use the terminology of the sports where the ball may be passed to a teammate or shot on the goal over the players' heads. These include basketball, rugby, soccer, and many others.

When a ball undergoes ballistic motion in the air, such as during a long-distance pass, the resulting trajectory can, ignoring friction, be approximated by a parabola. This would suggest a second-order motion model for ball-tracking, and indeed, such models have been applied in prior work to track an unoccluded ball.

However, our focus is on a harder problem, that of tracking the ball while some player is *in possession of* the ball, and also during relatively-short passes when the ball is in the vicinity of other players. Since a player who is in possession of the ball is likely to make abrupt moves in order to confound the opponents, the second-order model becomes less useful.

In the remainder of this section, we begin by introducing our model in its most generic form and then specialize it so that its individual components can be learned from the limited amount of training data, which is often the case. In Section 4 we will provide an example of an end-to-end system that can track the ball during long game sequences, and in that system, the second-order motion model will become useful for accurately segmenting long passes.

3.1. Conditional random field model

We use the notation of Table 1 to express our model. We represent the state of the ball at time t as $Y_t \in \{1, \dots, K, \ominus\}$, where K is the number of players. Either a player k is *in possession of* the ball (e.g., player k is dribbling the ball), or the ball is in state $\ominus$, meaning it is *free* (e.g., the ball is being passed or shot at the goal). Inferring the state of the ball $Y_t = y_t$ at time t enables us to estimate its ground-plane locations $G_t = g_t$ at all times by using a player's location when that player is in possession of the ball, and by interpolation when the ball is in-between; this is shown in Fig. 2.

For a sequence of video frames in the temporal interval $t = 1, \dots, T$ our goal is to infer the most likely state sequence $(y_1, \dots, y_T)$ of $Y = (Y_1, \dots, Y_T)$ given the image evidence $X = (X_1, \dots, X_T)$. We do so by minimizing a loss function

Table 1
Notation for our formulation.

K	The number of players
$\ominus$	No-one is in possession of the ball; i.e., the ball is *free*
Y	The state of the ball, $Y \in \{1, \dots, K, \ominus\}$
T	Number of frames in a temporal window $t \in [1,\ T]$
y_t	The realization of Y at time t
r_t	Auxiliary random variable derived from y_t
G	Ground-plane location of the ball on a 2D grid
X	Image evidence for all video frames
ψ_1	Unary potential of our model
ψ_2	Pairwise potential of our model
W	Parameters of our model
S	Phase of the game, $S \in \{0, 1, 2\}$
s_t	The realization of S at time t timeout ($s = 0$) or one of the two teams attacks
$\rho(\cdot)$	$\rho : k \mapsto \{1, 2, 3\}$ group membership of actor k: referee ($\rho(k) = 3$) or one of the two teams

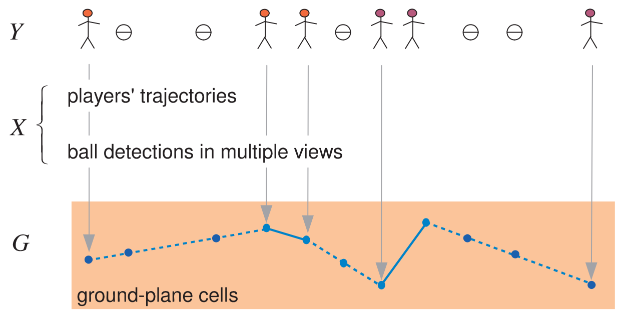


Fig. 2. An application of our approach to a ball-centric team sport. The possession of the ball Y is inferred from X; afterwards the ground-plane locations of the ball G are determined either by linking the locations of the corresponding players when $Y \neq \ominus$ (solid lines), and when $Y = \ominus$ by interpolation (dashed lines).

$\ell(y_1,\ldots,y_T;X,W)$, where W represents a set of learned parameters. We express the loss function ℓ in terms of a Conditional Random Field (CRF) [3] as

$$\sum_{t=1..T}\psi_1(y_t;X,W)+\sum_{t=1,\ldots,T-1}\psi_2(y_t,y_{t+1};X,W), \tag{1}$$

where our image evidence comprises per-view ball detections and the player trajectories.

Given analytical expressions for ψ_1 and ψ_2, as well as appropriate values for W, finding an optimal sequence of y's can be accomplished efficiently with dynamic programming. We now turn to deriving and learning these expressions and values.

In our implementation, ψ_1 and ψ_2 take the form of negative log-likelihood:

$$\psi_1(y_t;X,W_1)=-\log p(y_t;X,W_1)\ \text{and} \tag{2}$$
$$\psi_2(y_t,y_{t+1};X,W_2)=-\log p(y_{t+1}|y_t;X,W_2), \tag{3}$$

where $p(y_t)$ are $p(y_{t+1}|y_t)$ are probability distributions.

A naive learning of the potentials in Eqs. 2 and 3 would be quite complicated. The reason is that monolithic models for $p(y_t;X,W_1)$ and $p(y_{t+1}|y_t;X,W_2)$ would have to account for all the complexities of a team sport, such as the structure of the game, roles, and skill level. Instead, we factorize these probability densities to arrive at several well-defined learning tasks, as discussed in the following two sections.

3.2. Unary potential

In this section we factorize the unary potential in Eq. 2, derive the features to evaluate it, and define a method to learn its parameters.

3.2.1. Factorizating the unary potential

We first factorize the unary potential with respect to whether or not the ball is free and then with respect to the team that is in possession of the ball. Although in practice, the probability densities derived in the factorizations will all become functions of the image evidence X, our derivations do not require a particular form of X. We will therefore employ a shorthand notation, such as $p(y)\doteq p(y_t;X,W_1)$.

Factorizing ball possession. Frequency and duration of the ball possession tend to be influenced by the players' roles and skill levels. Thus, in theory, the knowledge of a player's identity can be used to design the ball-possession prior. While our formulation does not preclude such a prior, in our current implementation we treat all players as indistinguishable.

When the players' roles are unknown, the states of Y lose their semantics. This influences our supervised learning problem since the label $y=k$ cannot be applied consistently. This is in contrast with, say, learning semantic segmentation, where labels such as y = "sky", y = "grass" can be defined in a consistent manner.

In principle, one could learn $p(y)$ for $y\in\{1,\ldots,K,\ominus\}$. However, the learning task would have to contend with the fact that the label $y=\ominus$ is semantically different from the remaining K labels, and as noted earlier, the labels $\{1,\ldots,K\}$ have no semantic meaning. We therefore factorize $p(y)$ so that instead of dealing with one complicated learning task we solve several simpler ones.

We introduce an auxiliary random variable $R\in\{0,1\}$, which depends on Y as follows:

$$\begin{aligned} r=1 &\iff 1\leqslant y\leqslant K\\ r=0 &\iff y=\ominus. \end{aligned} \tag{4}$$

Therefore, we can write $p(y=\ominus|r=1)=0, p(1\leqslant y\leqslant K|r=0)=0, p(y=\ominus|r=0)=1$, and

$$p(y)=\underbrace{p(y|r=1)}_{\text{Eq.6}}p(r=1)+p(y|r=0)p(r=0), \tag{5}$$

where the computation of $p(y|r=0)$ follows from Eq. 4.

Factorizing the phase of the game. We now factorize the first term of Eq. 5 with respect to the phase of the game. Such a factorization will make it practical to track the ball throughout an entire game period.

Let $s=0$ denote timeout and $s\in\{1,2\}$ denote which of the two teams is attacking. When $y\neq\ominus$, i.e., $r=1$ we have:

$$\begin{aligned} p(y=k|r=1)&=\sum_{s\in\{0,1,2\}}p(y=k|s,r=1)p(s|r=1)\\ &=p(y=k|s=\rho(k),r=1)p(s=\rho(k))+\frac{1}{K}p(s=0), \end{aligned} \tag{6}$$

where all the probability densities are functions of X and W_1. In deriving Eq. 6 we approximated $p(s|r=1)$ by $p(s)$ which we found to be appropriate. However, if required, one could include the knowledge of the teams's relative strengths into the estimate of the phase of the game. During the timeouts we model the ball possession as uniformly distributed among the two teams and the referees, i.e. $p(y=k|s=0,r=1)=1/K$.

Summary of factorization. We have reduced the problem of learning $p(y)$ to the problem of learning the distributions $p(r)$ and $p(y=k|s=\rho(k),r=1)$, the latter being the probability that a player holds the ball given his team is in possession of the ball. Our factorization also requires learning $p(s)$, which can be accomplished using known approaches; our implementation is presented in Section 4.1.1.

3.2.2. Learning the unary potential

In deriving the factorized unary potential in Eqs. 5 and 6, we did not make any assumptions about the particular form of the image evidence X. Therefore, one has the freedom to derive features from X that are compatible with the envisioned application. For the sake of demonstration, we define geometric features based on projections of the monocular ball detections on the ground plane.

We accumulate multi-view evidence for the ball in a sparse ground-plane representation called the Ball Occupancy Map (BOM), and we have one such BOM for each time instant. The details of constructing the BOM are presented in Section 4.2.2. Briefly, monocular detections give rise to likely 3D locations of the ball, and these locations are then projected onto the ground plane, yielding peaks in the BOM; the geometric construction is shown in Fig. 3.

Given the BOM at time t, let $d_t^{j,k}$ be the distance between peak j and player k, see Fig. 4, and let c_t^j be the peak's score as defined in Section 4.2.2. We will use $d_t^{j,k}$ and c_t^j as primitives for deriving feature vectors, which will make it practical to learn $p(r)$ and $p(y)$.

Learning $p(y=k|s=\rho(k),r=1)$. For a player k we define a feature vector $\mathbf{x}_{\text{player}}(k;n)\in\mathbb{R}^{2n}$, where n specifies the number of BOM peaks that are used to construct this vector. This feature vector takes the form of $\mathbf{x}_{\text{player}}(k;n)=[\mathbf{c},\mathbf{d}]$. The first component of $\mathbf{x}_{\text{player}}(k;n)$ is $\mathbf{c}\in\mathbb{R}^n$, a vector of sorted BOM scores corresponding to the n peaks. The second component of $\mathbf{x}_{\text{player}}(k;n)$ is $\mathbf{d}$, which comprises a sorted set of distances $d_t^{j,k}$.

We train a probabilistic classifier to predict if a player k has the ball; in our implementation this classifier takes the form of a random forest. Classifier training is accomplished using examples of the form $(\mathbf{x}_{\text{player}}(k;n),\mathbf{1}(y^{\text{gt}}=k))$, where $\mathbf{1}(\cdot)$ is an indicator function, and $y^{\text{gt}}\neq\ominus$ is the ground-truth label. During prediction we obtain K predictions α_k, one for each player, and set $p(y=k|s=\rho(k),r=1)=\alpha_k\cdot\left(\sum_{i\in\rho(k)}\alpha_i\right)^{-1}$.

Learning $p(r)$. We define a feature vector $\mathbf{x}_{\text{possess}}(n)\in\mathbb{R}^{n\cdot(K+1)}$, where n specifies the number of BOM peaks used to construct this vector, and which takes the form of $\mathbf{x}_{\text{possess}}(n)=[\mathbf{c},\mathbf{d^1},\ldots,\mathbf{d^n}]$. The

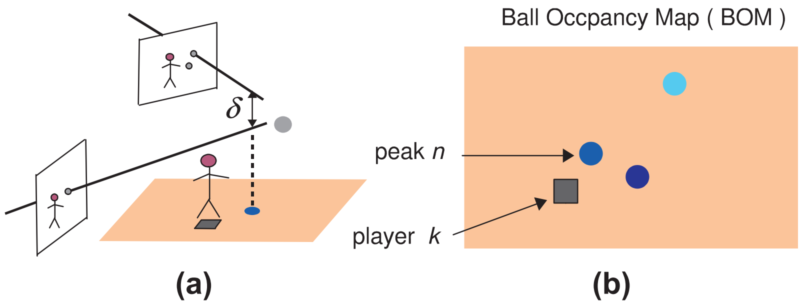


Fig. 3. The Ball Occupancy Map (BOM) is a sparse ground-plane representation of the image evidence. (a) For every ball detection in every view a ray is cast from the camera center. Two rays, each from a different view, cross within distance δ. (b) Projections of these crossing on the ground plane define peaks of the BOM. Each BOM peak has a score that is derived from the confidences of its corresponding monocular detections. Three peaks, each with a different score, are shown as circles of different color intensity. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

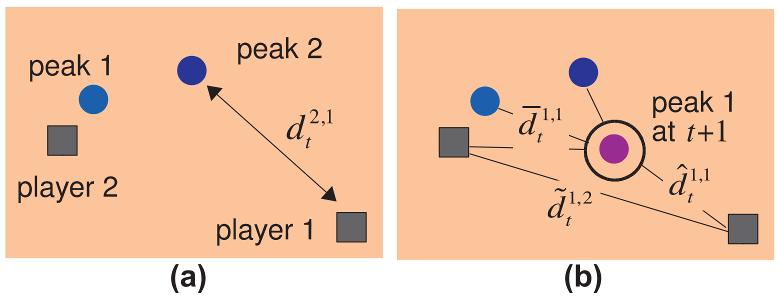


Fig. 4. Given the locations of the players and BOM at time t and $t+1$, we define a set of geometric primitives. These primitives will then be used to derive the features for learning ψ_1 and ψ_2. (a) We define $d_t^{j,k}$ as the distance between peak j and player k at time t. (b) We define $\tilde{d}_t^{k,k'}$ as the distance between player k and player k' at time t, $\hat{d}_t^{j,k}$ as the distance between peak j at time $t+1$ and player k at time t, and $\bar{d}_t^{j,j'}$ as the distance between peak j at time t and peak j' at time $t+1$.

first component of $\mathbf{x}_{\text{possess}}(n)$ is $\mathbf{c} \in \mathbb{R}^n$, a vector of sorted BOM scores. The remaining components of $\mathbf{x}_{\text{possess}}(n)$ are n vectors $\mathbf{d}^{\mathbf{j}} \in \mathbb{R}^K$. Each such $\mathbf{d}^{\mathbf{j}}$ comprises a sorted set $\{d_t^{j,k}\}_{k=1}^K$. An example of constructing $\mathbf{x}_{\text{possess}}(n)$ for $n = 2$ is shown in Fig. 4. Our intuition for this feature is that if one observes a BOM peak with a high score, and that peak is distant from any player on the ground, then it is more likely that the ball is free, i.e., $r = 0$.

In our implementation, $p(r)$ takes the form of a random forest classifier. The classifier is trained in a standard fashion on a set of pairs of the form $(\mathbf{x}_{\text{possess}}(n), r)$.

3.3. Pairwise potential

In this section we factorize the pairwise potential in Eq. 3, derive the features to evaluate it, and define a method to learn its parameters.

3.3.1. Factorization of the pairwise potential

In the previous section we derived a factorized form of the unary potential with respect to R and S, which in turn made it practical to learn this potential from the training data. We factorize the pairwise term with respect to teams' tactics and spatial context. The tactics defines a team's behavior such as frequency of passes, which is independent of players' locations, while the spatial context models the temporal evolution of the ball conditioned on image evidence. In other words, the pairwise term encodes both empirical knowledge and contextual information. Formally, we write

$$p(y_{t+1}|y_t; X, W_2) = p_{\text{tactics}}(y_{t+1}|y_t; W_2)\ \underbrace{p_{\text{context}}(y_{t+1}|y_t; X, W_2)}_{\text{Eq.8}}, \qquad (7)$$

where p_{tactics} and p_{context} model teams' tactics and spatial context respectively. In the next section we show our features derived from geometric primitives defined on the BOM, and how we apply these features to learn Eq. 7.

3.3.2. Learning of the pairwise potential

We define $\tilde{d}_t^{k,k'}$ as the distance between player k and player k' at time t; $\hat{d}_t^{j,k}$ as the distance between peak j at time $t+1$ and player k at time t; and $\bar{d}_t^{j,j'}$ as the distance between peak j at time t and peak j' at time $t+1$. An example of these primitives is shown in Fig. 4.

Learning p_{context}. To learn the spatial context term, we apply a truncated zero-mean Laplace distribution $\boldsymbol{\kappa}(\cdot\,; \boldsymbol{\lambda})$ with parameter $\boldsymbol{\lambda} \in \mathbb{R}^2$ specifying its bandwidth and the truncation threshold. Note that, this distribution is a radius basis function whose probability depends only on distances. Since all our features for learning p_{context} are derived based on distances, this simple model serves our purpose well. We write,

$$\begin{aligned} p_{\text{context}}(y_{t+1} = k'|y_t = k) &= C_{t+1}^{-1} \\ &\times \boldsymbol{\kappa}(\min_{j,j'}\{|\bar{d}_t^{j,j'} - v_{\text{ball}}|\}_{j,j'}; \boldsymbol{\lambda}_1)^{\mathbf{1}(r_t=0)\cdot\mathbf{1}(r_{t+1}=0)} \\ &\times \boldsymbol{\kappa}(\min_{j}\{|\hat{d}_t^{j,k} - v_{\text{ball}}|\}_j; \boldsymbol{\lambda}_2)^{\mathbf{1}(r_t=0)\cdot\mathbf{1}(r_{t+1}=1)} \\ &\times \boldsymbol{\kappa}(\min_{j}\{|\hat{d}_t^{k,j} - v_{\text{ball}}|\}_j; \boldsymbol{\lambda}_3)^{\mathbf{1}(r_t=1)\cdot\mathbf{1}(r_{t+1}=0)} \\ &\times \boldsymbol{\kappa}(\tilde{d}_t^{k,k'}\,; \boldsymbol{\lambda}_4)^{\mathbf{1}(r_t=1)\cdot\mathbf{1}(r_{t+1}=1)}, \end{aligned} \qquad (8)$$

where C_{t+1} is a normalization constant, v_{ball} is the average speed of the ball in flight, $\mathbf{1}(\cdot)$ is an indicator function, and X and W_2 are implicit. Note that, p_{context} is modeled as a product of four zero-mean Laplace distributions, and the indicator variable guarantees that only one distribution can be selected. The four distributions model the following four scenarios:

1. **The ball being free at time** t **and** $t+1$. In this case, the min operator operates on the distances between BOM peaks at time t and $t+1$, and selects the one that is closest to the average

speed of the ball. With the Laplace distribution $\boldsymbol{\kappa}(\cdot;\boldsymbol{\lambda})$, the probability is peaked when the distance is equal to the average ball speed. Intuitively, the fact that two ball detections whose distance approximates the average speed of the ball, is an evidence of the ball being free.

2. **Player** k **receiving the ball at time** $t+1$. With the min operator, we select the BOM peak at time t whose distance to player k at time $t+1$ best approximates the average speed of the ball. When they are equal, it is an evidence of a player receiving the ball, in which case the probability is peaked.
3. **Player** k **shooting the ball at time** t. It is semantically symmetric to last case: we select the BOM peak at time $t+1$ whose distance to player k at time t best approximates the average speed of the ball. When they are equal, it is an evidence of a player shooting the ball, in which case the probability is peaked.
4. **Players in possession of the ball at time** t **and** $t+1$. In this case, the probability decreases as the distance between players increases. This term encodes that, the probability of ball transition between players is high when the players are closed, and vice versa.

Note that, in all the above cases, when the distance is larger than the truncation threshold of $\boldsymbol{\kappa}(\cdot;\boldsymbol{\lambda})$, the probability collapses to zero. This corresponds to the fact that, a ball cannot physically transit to a distant place within one frame given sufficient frame rate.

We learn the $\boldsymbol{\lambda}$'s by minimizing the classification error on a training sequence $(y_1^{gt},\dots,y_T^{gt})$. In our experiments we have found it sufficient to use a single $\boldsymbol{\lambda}$ for all the terms in Eq. 8, and perform a grid search with fewer than ten values in each dimension.

Learning p_{tactics}. We learn $p_{\text{tactics}}(y_{t+1}|y_t)$ by decomposing it into two multinomial distributions with respect to the two possible outcomes for r_t. The first distribution takes the form of $p(\Omega_1|r_t=0)$, where $\Omega_1=\{y_{t+1}=\ominus, y_{t+1}\neq\ominus\}$. The second distribution takes the form of $p(\Omega_2|y_t,r_t=1)$, where $\Omega_2=\{y_{t+1}=\ominus, y_{t+1}=y_t, (y_{t+1}\neq y_t)\wedge(\rho(y_{t+1})\neq\rho(y_t)), (y_{t+1}\neq y_t)\wedge(\rho(y_{t+1})=\rho(y_t))\}$. We learn the parameters of these multinomial distributions using maximum-likelihood estimation.

In other words, we learn the following empirical frequencies: (1) the ball being free, (2) the ball received by a player, (3) the ball shot by a player, (4) a player in possession of the ball, (5) the ball being stolen by a player from the other team and (6) the ball given to a player in the same team.

4. Complete system

To validate our approach we implemented a complete system for tracking a ball in team sports over an extended period of time. Because the system, summarized in Alg. 1, exploits knowledge about the scene, we will refer to it as Focus on the Scene (FoS).

The input to our system comprises multi-view video sequences in the temporal interval $t\in[1,T]$, and W, the parameters of for the unary and the pairwise potential. In Step 1 players and referees are tracked, and labeled according to their team membership. In Step 2, the phase of the game, i.e., $\mathbb{S}=\{p(s_t)\}$, is estimated. In Step 3, $\mathbb{B}=\{\text{BOM}_t\}$ is obtained for $t\in[1,T]$. In Step 4, the algorithm performs temporal segmentation by detecting *long pass* segments $\mathbb{T}_0$ and labeling the rest as $\mathbb{T}_1$; this step is defined in Section 4.2.3. In Step 5, for every time index $t\in\mathbb{T}_0$ the algorithm assigns y_t the $\ominus$ label. In Step 6, the algorithm finds the y_t's for $t\in\mathbb{T}_1$ that minimizes the loss in Eq. 1, given that $y_{t'}=\ominus$ for $t'\in\mathbb{T}_0$. The optimal sequence of ball states $\hat{y}_1,\dots,\hat{y}_T$ computed in Step 6 is transformed into a sequence of ground-plane cells $\hat{g}_1,\dots,\hat{g}_T$ by indexing player locations in X when $y\neq\ominus$ and linearly interpolating between player locations for $y=\ominus$.

In the remainder of this section we state the algorithm's main components. We then describe the parameter settings.

Algorithm 1. Focus-on-the-Scene (FoS) ball tracking

```
Input: Multi-view video sequences for $t \in [1, T]$ and
$W$ = {parameters of $\psi_1$ and $\psi_2$ (Sections 3.2 and 3.3)}.
 1. track players and assign team labels (Section 4.1.2)
 2. estimate $\mathbb{S}$, the phase of the game (Section 4.1.1)
 3. compute $\mathbb{B} = \{\text{BOM}_t\}$ for $t \in [1, T]$ (Section 4.2.2)
 4. segment $[1, T] = \mathbb{T}_0 \cup \mathbb{T}_1$, where $\mathbb{T}_0 \cap \mathbb{T}_1 = \emptyset$, and $\mathbb{T}_0$
    corresponds to long passes (Section 4.2.3)
 5. set $y_t \leftarrow \ominus$ for $t \in \mathbb{T}_0$
 6. let $X$ =(player tracks, $\mathbb{B}, \mathbb{S}$) and
      set $\hat{y}_1,\dots,\hat{y}_T \leftarrow \arg\min_{y_1,\dots,y_T,\ t\in\mathbb{T}_1} \ell(y_1,\dots,y_T; X, W)$
Output:
$\hat{g}_1,\dots,\hat{g}_T \leftarrow$ linearly interpolate $(\hat{y}_1,\dots,\hat{y}_T)$
```

4.1. Game phase and player trajectories

Our implementation of player-tracking and trajectory-estimation takes advantage of known approaches and publicly-available software.

4.1.1. Inferring the phase of the game

Our set of features $\mathbf{x}_{\text{phase}}$ is inspired by [16], except that in our case player trajectories and team membership are determined automatically, and referees are also tracked. Our feature vector $\mathbf{x}_{\text{phase}}$ comprises: 1. locations of players, sorted along the court length within their own team; 2. velocities of all players, sorted within their own team; 3. locations of centroids of each team; and 4. velocities of centroids of each team.

We learn a random forest to infer the phase of the game. Our training set comprises pairs $(\mathbf{x}_{\text{phase}}, s)$.

4.1.2. Player tracking

To track the players on the ground plane, we apply the Multi-Commodity Network Flow proposed in [28], which relies on the publicly-available software [4,5]. We obtain people tracks with team memberships from the two teams and the referees.

4.2. Ball detection and ballistic trajectory estimation

Our monocular ball detection relies on standard methods described in Section 4.2.1. These detections are accumulated in BOM, a novel data structure which is described in Section 4.2.2.

4.2.1. Monocular ball detection

To detect the ball from a single view, we first conduct eigen-background subtraction [29] to extract the foreground pixels. We obtain the template color-histogram of the ball from our training sequence as follows: for each training image, we obtain the histogram of the Hue, Saturation and Value channels separately, and then we concatenate them into one single histogram; finally we average all such histograms from all training images and obtain the template histogram of the ball. Then we scan all the foreground pixels in a sliding-window manner, obtain the color-histogram of the current window and compute the inner-product between the template histogram and the current histogram. Thus, we get a confidence for each pixel being the center of the ball. We also apply the hough circle transform to detect round objects. For each detected circle, we increase the confidence by β_1. Moreover, we increase the confidence of pixels that are not part of players by β_2. This is because we conduct detection by background subtraction, moving object that is not part of players is likely to be the ball. In our implementation, we obtain the per-view projections of the players, and then increase the confidence of the ball detections that fall outside of the players' projections.

4.2.2. Constructing the Ball Occupancy Map (BOM)

Monocular image evidence for the ball can be quite noisy due to the small apparent size of the ball and frequent occlusions. Reconciling this noisy information across multiple views is therefore non-trivial. Since our inference is performed in terms of the ground-plane player locations we accumulate evidence for the ball in a sparse ground-plane representation called the Ball Occupancy Map (BOM), which was introduced in Section 3, which we now define precisely.

Starting with the per-view detection results, we apply triangulation to reconstruct the 3D ball detections in a pair-wise manner. In other words, for any two detections from two views, we cast the lines of sight. Only those pairs of lines that cross within a threshold δ are kept, from where a 3D position of the point closest to both lines is recovered. The score of such 3D detection is set to be the product of per-view detections. In case there are more than two views, we project the obtained 3D detection on other views, and multiply the corresponding scores. Once we compute the scores, we project the 3D intersections onto the ground plane. We refer to these projections as BOM *peaks*, and we set the scores of BOM peaks to be the scores of the corresponding 3D detections. In Fig. 3, we show an example of BOM, where players are denoted by squares and the peaks of BOM are denoted by circles.

4.2.3. Temporally segmenting long passes

As mentioned in Section 3, we consider long passes as evidence for the ball being free. The approaches of [13,30] and others have addressed the problem of finding segments of parabolic trajectories. However, our objective is to only temporally segment the long pass, rather than smoothing those track segments with the second-order model. In the basketball case, our algorithm grows parabolic segments from detections and stops when the *coefficient of determination* is below a threshold; in the soccer case, our algorithm detects segments whose distance to the nearest player is larger than a distance threshold.

4.3. Parameters

Our parameter settings are as follows. For the monocular ball detection, we set $\beta_1 = 0.2$ and $\beta_2 = 0.1$. For long passes detection, we set the coefficient of determination to be 0.95 for temporal segmentation in the basketball case, and distance threshold to be 1.5 m in the soccer case. We use a publicly-available implementation [31] of the random forest classifier with default parameters and 1000 trees. For the p_{context} term in Eq. 8 we set $\lambda = (10^{-3}, 300\,\text{cm})$ and $\lambda = (10^{-3}, 800\,\text{cm})$ for basketball and soccer respectively.

5. Experiments

We validate our approach on challenging basketball and soccer video sequences, described below. For all datasets, we provide videos with our tracking results. They are available from: http://cvlab.epfl.ch/research/balltracking.

5.1. Datasets

We have used two sequences of the FIBAW (women's championship at Karlovy Vary) dataset captured at the 2010 women's world championship: 1. Mali vs. Senegal match (FIBAW-1), 8480 frames; and 2. Czech Republic vs. Belarus (FIBAW-2), 2850 frames. The matches are captured by ten stationary synchronized 25-frame-per-second cameras with resolution of 1294×964. For the two matches, the cameras are placed at different places around the court. We use three cameras for FIBAW-1 and four for FIBAW-2 with overlapping fields-of-view; these cameras are sufficient for having a coverage of at least two different views for the entire court. We have manually annotated one out of every ten frames for FIBAW-1 and 500 consecutive frames for FIBAW-2, and we perform the quantitative comparison using these frames.

The APIDIS dataset is presented in [32]. It comprises a basketball match sequence captured by seven stationary unsynchronized 2-megapixel 22-frames-per-second cameras placed above and around the court. The authors also provide a pseudo-synchronized sequence and we conduct tracking on it. The APIDIS dataset is challenging for ball tracking because the camera locations are not optimized for capturing the ball and the lightning conditions are far from optimal, as there are many direct light sources which are reflected on the court while some other regions are shaded. Especially when the ball is near the basket, the ball is completely merged with the background. We use two cameras with overlapping fields-of-view and choose a sequence of an attacking phase, whose length is much longer than other ball-based work on the same dataset, such as [15].

The ISSIA dataset comprises a multi-view soccer sequence [33] of 3000 frames captured by six full-HD 25-frames-per-second cameras placed in two sides of the soccer field. We use all the cameras for our experiments. However, the camera settings are designed specifically for player tracking, and thus they do not provide a complete converge of the ball when it is flying in the air. We conduct tracking on a sequence of 2000 frames where the ball is mostly in sight.

5.2. Baseline methods

Because our FoS approach is novel and our testing sequences are challenging, there is no closely related baseline that we can use for comparison. As mentioned in the introduction, the state-of-the-art approach of [2], which has been shown to be successful in single object tracking, never tracks the ball for more than five consecutive frames. We therefore compare our approach both to a detection-linking tracker operating in a batch-smoothing framework and to a version of our approach that does not use the features that we derived from BOM and the player trajectories. We describe the two baselines below.

Focus on Detection (FoD). We implement a detection-linking approach that we call Focus on Detection (FoD). The baseline FoD ball tracker involves three steps: ball detection in each camera view using the same detector that was defined in Section 4.2.1, generation of the candidate ball locations in 3D using the same approach as defined in Section 4.2.2, but without projecting these candidates into BOM in the ground plane, and linking the 3D detection candidates. Similarly to the FoS tracker, we use two motion models: a second-order model during long passes and a first-order model otherwise.

FoS with non-conditioned potentials. We implement ψ_1^* and ψ_2^* that are not conditioned on X. The non-conditioned ψ_1^* follows Eq. 5 as does ψ_1, but we now set $p(r=0)$ to a constant learned from the data, and we set $p(y|r=1)$ to be $(1-p(r=0))/K$. The non-conditioned ψ_2^* follows Eq. 7, but since now p_{context} does not depend on X, it becomes a uniform distribution. This baseline is useful in evaluating the contribution of the novel features we derived from BOM and the player trajectories. In particular, we will evaluate the contribution of ψ_1 vs. ψ_1^* and ψ_2 vs. ψ_2^* to the overall accuracy of our algorithm.

In addition, we also test the CRF model with only the unary term in Eq. 1 to judge the importance of the pairwise potential.

5.3. Performance metrics

For experimental validation we compute the tracking accuracy score (TA), which is defined as

$$\mathrm{TA}(\tau) = \frac{1}{T} \sum_{t=1,\dots,T} \mathrm{TP}(t;\tau) \tag{9}$$

and summarize the quality of the tracker as a non-negative score in [0, 1], the higher the score the better. The TA curve is also known as the "precision plot" in [34]. When the output of the tracker is within a distance threshold τ centimeters from the annotated ground truth location, we define the detection as a True Positive $\mathrm{TP}(t;\tau)$ detection. Rather than picking a τ value arbitrarily, we present our results for a range of τ as a monotonically increasing curve, e.g. $\tau \in [5,\dots,125]$ cm for basketball. With a fixed τ, the True Positive rate over all frames is known as the Success Rate (SR) [2].

To provide a complete summary of performance, we also use the Center Location Error (CLE) metric [34], which is defined as the distance between the tracking result and the ground truth on the ground. A small CLE indicates accurate tracking. The mean of CLE (i.e. MCLE), defined as the average CLE over a sequence, is also used in our evaluation.

5.4. Results

For basketball, we train classifiers for the unary potential using the FIBAW dataset, test on the APIDIS dataset, and vice versa. For soccer, due to there being only one sequence available, we are not able to learn a unary potential conditioned on image evidence. Therefore, we apply FoS with a non-conditioned unary potential. We set $n = 1$ for basketball sequences and fix the parameters as specified in Section 4.3. We compute the TA curve and the MCLE for each test sequence. If the ball is not detected as part of a long pass and is inferred to be free, we conduct the following processing for better visualization: first, we obtain the BOM peak whose distance is closest to the interpolated location on the ground, then we find the corresponding 3D ball detection, and last we project the 3D ball detection on each camera view.

In Fig. 8, we show the performance of long pass segmentation using the approach discussed in Section 4.2.3. We take the segmentation accuracy to be the number of frames that are correctly classified as long passes, over the number of frames whose ground truth are long passes *or* classified as long passes. As can be seen, the segmentation does not perform well by itself, due to the noisy ball detections. However, our proposed approach is able to take advantage of these noisy input and produce dependable ball tracking as shown in Fig. 5. In Fig. 6, we show ball trajectories on an orthogaraphic view of the court. We observe that on all datasets, our obtained trajectories and ground-truth trajectories are almost superposed.

FoS tracking vs. FoD tracking. In Fig. 5, we show the comparison of tracking performance by TA curve. We compare three approaches: FoD, FoS and oracle, which corresponds to a human observer specifying which player, if any, is in possession of the ball.

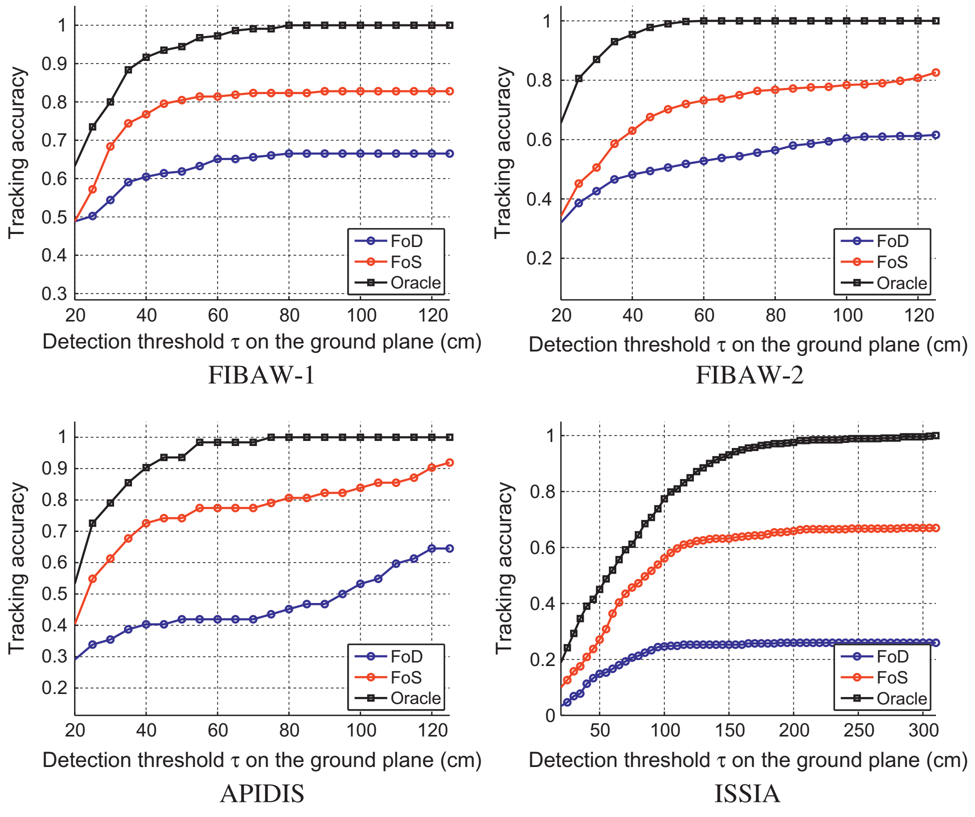


Fig. 5. Comparison of FoD and FoS tracking. The performance of FoS approaches that of the oracle and is decisively better than FoD, particularly in the accuracy range of $\leqslant$100 cm. The results are obtained by fixing $n = 1$. From left to right: results on FIBAW-1, FIBAW-2, APIDIS and ISSIA dataset.

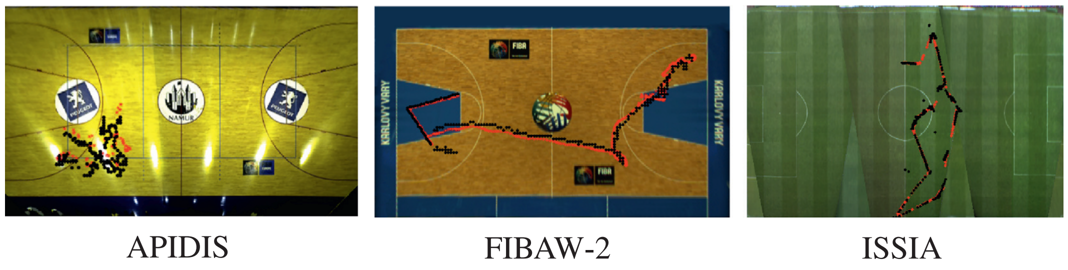


Fig. 6. Projection of FoS tracking results (in black circles) and the ground truth (in red stars) overlaid on an orthographic view of the court. The real and the recovered trajectories are almost superposed. (left) Results on the APIDIS dataset, frames [1 400] (middle) FIBAW-2 dataset, frames [975 1354] (right) ISSIA dataset, frames [880 1399]. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

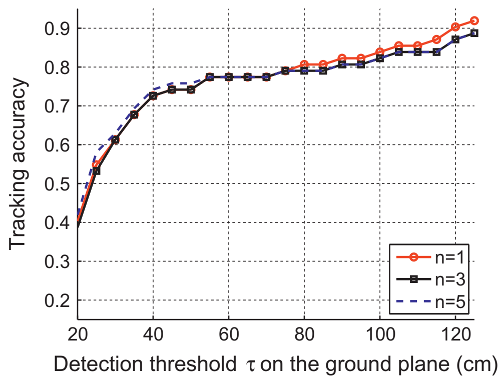


Fig. 7. Tracking performance with different settings of n, i.e. the number of BOM peaks used for training the unary potential.

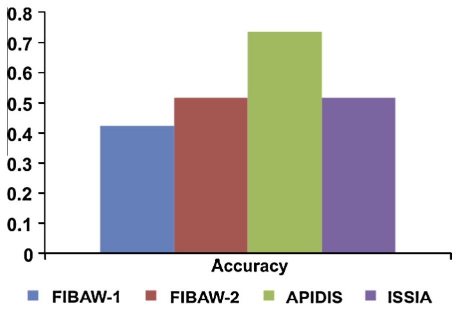


Fig. 8. Accuracy of long pass segmentation. We apply the approach as described in Section 4.2.3, and show the resulting accuracy on the four datasets.

Our FoS tracker significantly improves over the baseline FoD tracker and approaches the oracle tracker on all sequences. For the APIDIS dataset, the huge improvement suggests that with such bad lighting conditions and occlusions of the players it is difficult to track the ball relying only on direct ball detections. The same trend is observed on the other datasets.

Parameters of FoS. In Fig. 7, we show the tracking accuracy of our FoS tracker for different settings of n, trained on FIBAW-1 and tested on APIDIS. The obtained results are similar. We have also trained on APIDIS and tested on both FIBAW-1 and FIBAW-2, where the same trend is observed.

Performance of classification tasks. We learn classifiers to estimate $p(s), p(r)$ and $p(y|s, r = 1)$. Here we show performance of classification using random forest, trained on FIBAW-1 and tested on APIDIS dataset. We apply the approach as described in Section 4.1.1 to learn $p(s)$, and we obtain a resulting accuracy of 0.81, where the accuracy is defined as the number of frames whose phase is correctly classified over the total number of frames. Recall that this classification task is a 3-class classification problem (i.e., each team attacking and time-out) and each frame is classified independently. We found that most classification errors happen in frames that are temporally close to the phase transitions, such as from one team attacking to time-out. For example, when a ball falls into the basket, we consider it as the start of a time-out. However, by looking at the players' locations alone, it is even challenging for a human observer to tell whether such frame is time-out or not. On learning $p(r)$, we obtain a recall of 0.71 and a precision of 0.53. The recall and precision are defined as the number of frames where the ball is correctly classified as being free, over the total number of frames that the ball is indeed free, and the total number of frames that the ball is classified as free, respectively. On learning $p(y|s, r = 1)$, we obtain a resulting accuracy of 0.45 where the accuracy is defined as the number of frames where ball possession is correctly

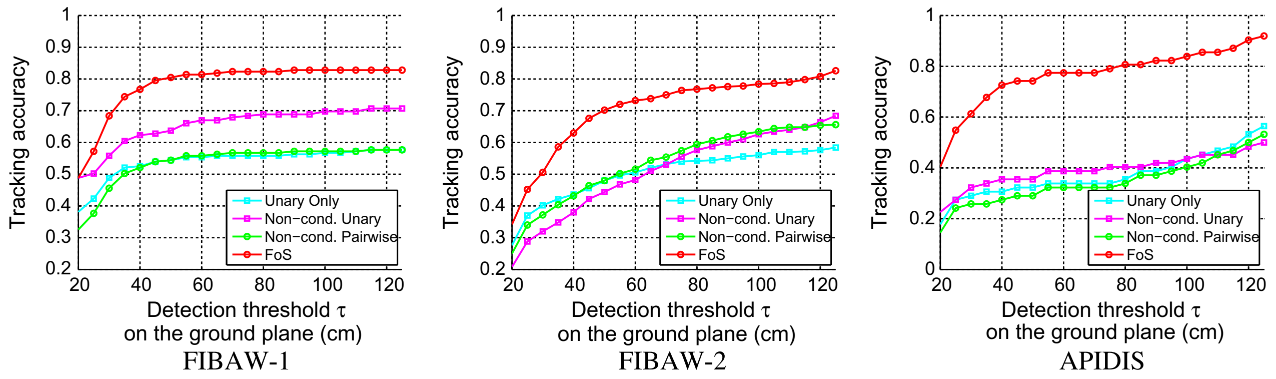


Fig. 9. Contribution of the pairwise term and conditioned unary/pairwise terms. (left) Results on the FIBAW-1 dataset (middle) Results on the FIBAW-2 dataset (right) Results on the APIDIS dataset. We compare FoS tracker against three baselines: the one without pairwise term (i.e., Unary Only), the one with non-conditioned unary (i.e., Non-cond. Unary) and the one with non-conditioned pairwise (i.e., Non-cond. pairwise).

Table 2
Success Rate (SR) at the threshold of 30 cm.

Algorithm	FIBAW-1	FIBAW-2	APIDIS
FoD	0.544	0.426	0.354
Non-cond. unary	0.558	0.320	0.323
Non-cond. pairwise	0.456	0.372	0.258
FoS	0.684	0.506	0.613

Table 3
Success Rate (SR) at the threshold of 100 cm.

Algorithm	FIBAW-1	FIBAW-2	APIDIS
FoD	0.665	0.604	0.532
Non-cond. unary	0.697	0.626	0.436
Non-cond. pairwise	0.572	0.634	0.403
FoS	0.828	0.784	0.839

classified over the total number of frames where someone is in possession of the ball. Note that, the classifiers do not perform very well on their own, however our CRF model corrects the part of errors with the help of the pairwise term and produces reliable ball trajectories.

Contribution of image evidence. As Fig. 9 indicates, using our proposed conditioned potentials yields decisive improvement. In particular, when the non-conditioned pairwise potential is used, the accuracy is significantly lower, particularly for $\tau > 60$ cm. The effect of the non-conditioned unary potential exhibits the same trend for both datasets, roughly following the curve for the non-conditioned pairwise term, and below the accuracy for the model that uses X in both terms. When only the unary term is used, the tracking accuracy drops significantly on all the datasets, which indicates that the pairwise potential plays an important role in improving the tracking results.

In Tables 2 and 3 we show the Success Rate (SR) of all algorithms, at distance thresholds of 30 cm and 100 cm respectively.

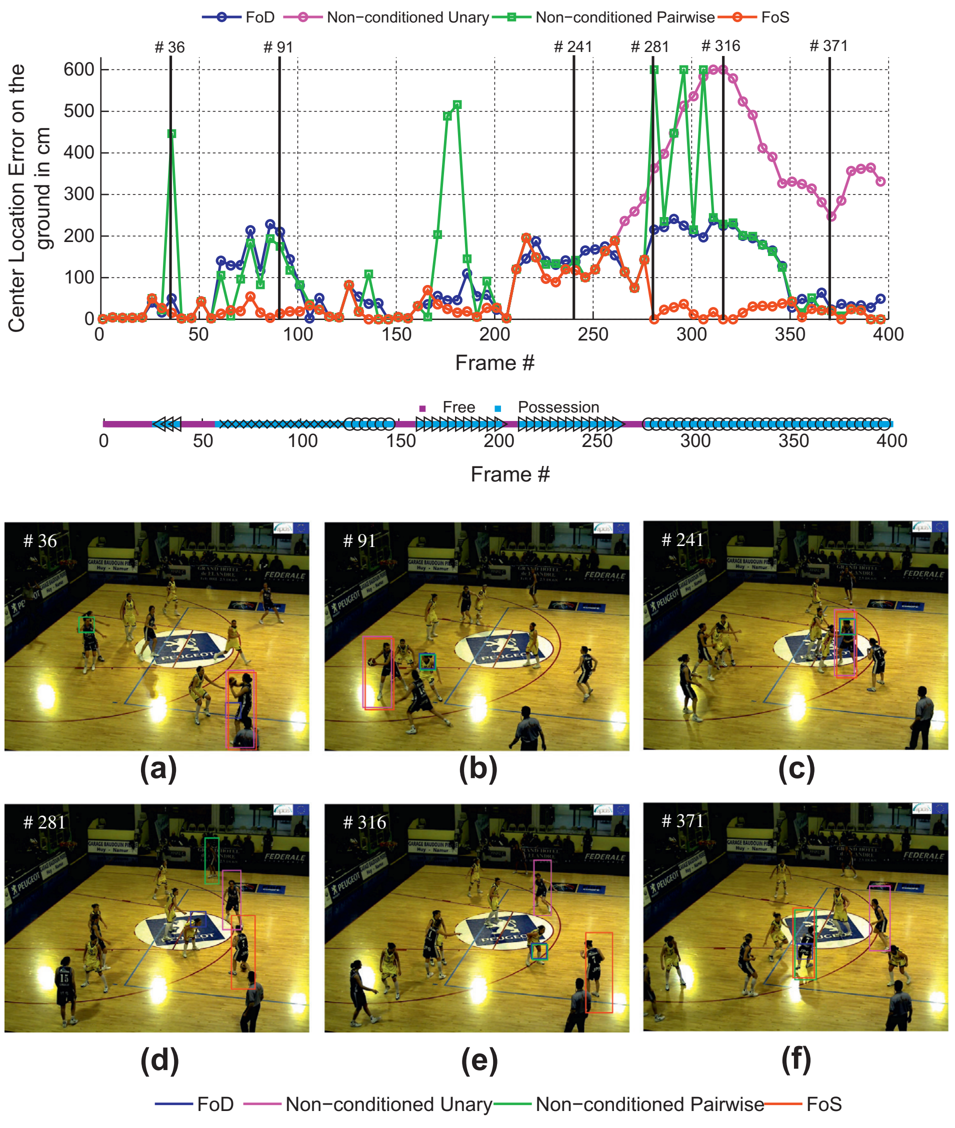


Fig. 10. (top) Center Location Error (CLE) vs. time, (center) obtained ball states, and (bottom) example tracking results on the APIDIS dataset. The temporal range evaluated is frames [1 400], which is an attacking phase that contains interesting cases such as shot-passes and FoS failures. In the obtained ball states, we show the frames where the ball is predicted to be free by purple, and possessed by blue. We use different markers to represent different players in possession of the ball. In (a)–(f), the FoS tracker assigns the ball to the correct player. The non-conditioned unary tracker succeeds in (a)–(c) but fails in (d)–(f); the non-conditioned pairwise tracker succeeds only in the case of (f); the FoD tracker fails in all the cases. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

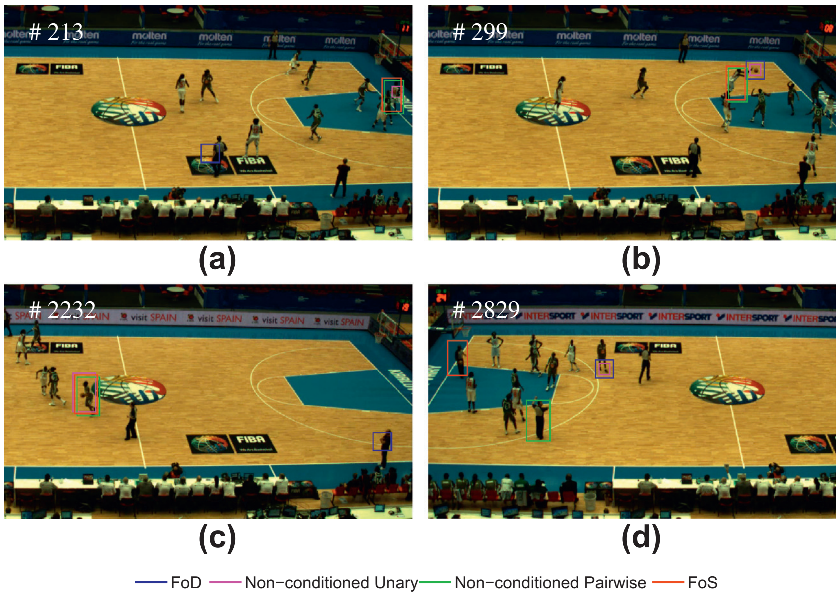


Fig. 11. Example tracking results on the FIBAW-1 dataset. FoS assigns the ball to the correct player in (a)–(c) during attacking phases, and in (d) during timeout. In (b), although FoS assigns the ball to the correct player, the tracking is less precise than FoD, which tracks the ball itself.

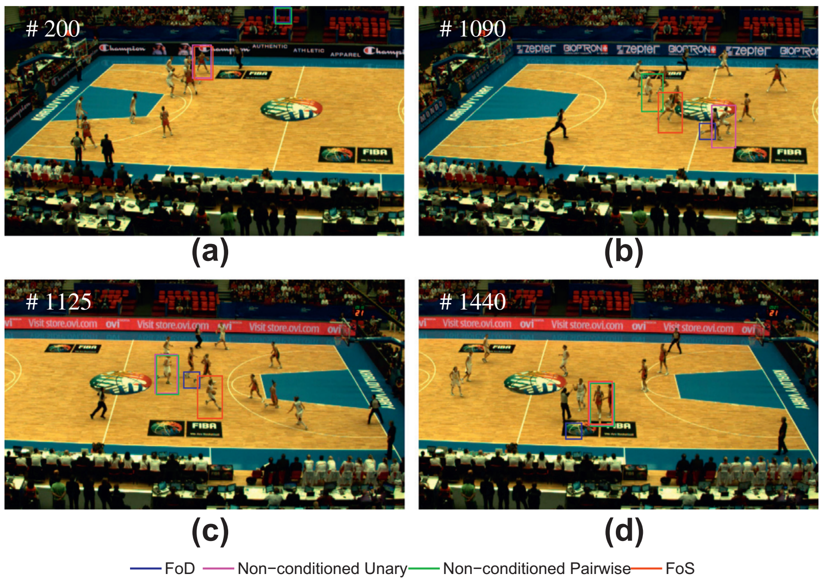


Fig. 12. Example tracking results on the FIBAW-2 dataset. FoS assigns the ball to the correct player in (a)–(c) during attacking phases, yet fails in (d) during timeout: a player in a white jersey is in possession of the ball, while FoS tracks a nearby player in a red jersey. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

On all the datasets, FoS yields an SR over 50 percent at a threshold of 30 cm and over 78 at a threshold of 100 cm, which are significant higher than other trackers.

In Fig. 13, we show the performance of different trackers using MCLE. The FoS agains yields significant improvement. On the FIBAW-1 dataset, the MCLE is around 100 cm for FoS and >300 cm for FoD. This means that on average, our tracking result is as close as 100 cm to the ground truth, which is reasonably satisfactory given the size of basketball court is 28 m $\times$ 15 m. The same trend is also observed on the APIDIS dataset: the MCLE is less than 40 cm for FoS and around 75 cm for FoD. On the ISSIA dataset, the MCLE is 162.7 cm for FoS and 272.3 cm for FoD. Note that the MCLE we obtained for soccer is larger than that of basketball. This can be explained in part by the definition of ball possession in soccer: when players dribble the soccer ball, they may let it roll

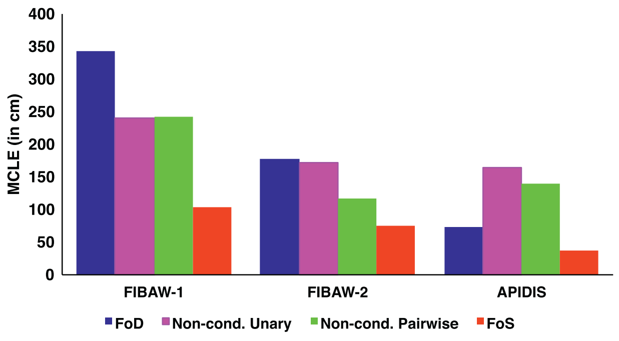


Fig. 13. Performance comparison using the Mean Center Location Error (MCLE) metric. The MCLE of FoS is lower than other trackers on all datasets.

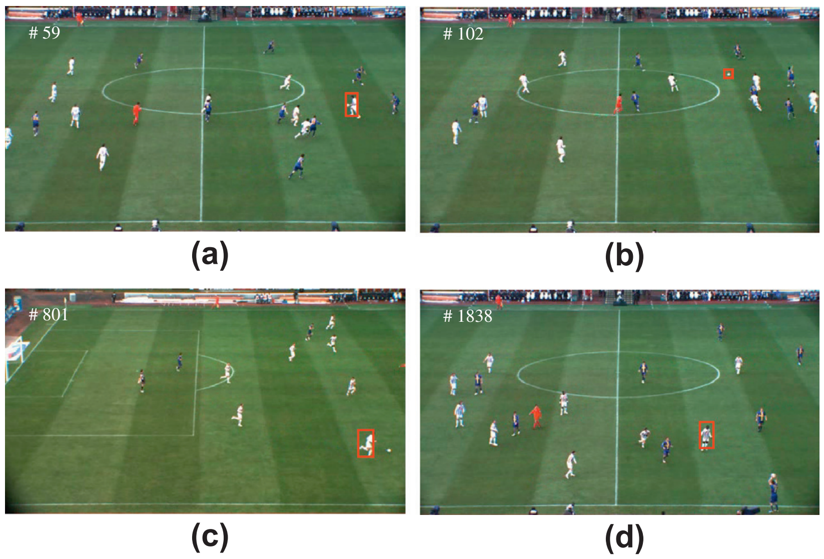


Fig. 14. Example tracking results on the ISSIA dataset. FoS tracks the correct player in (a) and (c), but tracks the incorrect one in (d). In (b), the ball is detected as part of a long pass. In (c), the player dribbles the ball by letting it roll. In this case, although the player is in possession of the ball, the distance between the ball and the player is large compared with that of basketball.

on the ground ahead of them. Therefore, the distance between the player and the ball is larger than that of basketball.

We show CLE vs. frame and the corresponding tracking results on the APIDIS dataset in Fig. 10. We observe that the CLE of FoS is less than 30 cm in most frames and is never larger than 200 cm. From frame 205 to 275, the CLE of FoS is large, as it tracks a player who is not in possession of the ball. FoS makes such errors because the tracked player and the player who is indeed in possession of the ball are of the same team and close to each other. From frame 260 to 400, there is a significant divergence of the non-conditioned unary tracker, as it incorrectly assigns the ball to a player who is far from the ground truth. From frame to frame, the CLE of non-conditioned pairwise tracker fluctuates widely. This is because we ignore X by assuming p_{context} as a uniform distribution, and the distances between players are not encoded in the transition probability. Therefore, the transition between two players who are far apart does not get penalized, which results in the fluctuation in tracking results and the CLE.

Analysis of failures. Our proposed FoS tracking approach decisively improves on the state-of-the-art. However, given the ambiguous image evidence and close proximity of the players in our datasets, the FoS tracker may sometimes yield incorrect results. We show some cases where the estimated ball location differs from the ground truth in the example tracking results. In Fig. 10(c), Fig. 12(d) and Fig. 14(d), the tracker assigns incorrect player possession. In Fig. 11(b), because the ball detection is close to a player, FoS assigns the ball to that player, while the FoD maintains the correct 3D location.

One approach to preventing these types of mistakes is to refine the model of ball possession. In particular, the tactics and the context terms of Eq. 7 could be guided by a more nuanced knowledge of the particular sport. In applications where post-processing of the ball trajectories is feasible, such post-processing can be guided by a higher-order model of the tactics.

Summary of experiments. Our FoS approach decisively outperformed the state-of-the-art FoD on all of our datasets as measured by the standard tracking-performance metrics. This outcome was obtained by automatically learning the parameters of our model, and keeping all other parameters of our complete system fixed across all experiments.

In addition to higher accuracy, FoS is computationally-efficient. On a Quad-Core 2.50 GHz CPU running MacOS, given pre-computed monocular ball detection and excluding the frame rate, the throughput of FoS is 8–10 frame/s.

6. Conclusion

We have presented a novel approach for tracking the "invisible" ball in team sports, which works in contrast to the common ball-tracking approaches. In our approach, we first track players, decide who is in possession of the ball and then utilize players' trajectories to achieve reliable ball tracking. By introducing state space that explicitly accounts for the ball possession, we define our loss function as a CRF, whose unary and pairwise potentials are conditioned on image evidence. We factorize the unary potential with respect to whether the ball is free or not as well as the phase of the game, and then learn the potentials using properly trained classifiers. We factorize the pairwise potential with respect to tactics and context.

We have applied our proposed tracker to challenging basketball and soccer sequences up to 8480 frames. Our approach compares favorably to the state-of-the-art using all evaluation metrics. We also demonstrates the contribution of conditioned unary and pairwise potentials using the same evaluation metrics.

In future work we plan to jointly learn parameters of the unary and the pairwise terms. We also look forward to applying our technique to other sports, such as handball and volleyball. Finally we ran an informal experiment at the CVPR'13 conference [35]. It showed convincingly that given well-defined team formations, domain experts and even some amateur players can predict the location of the basketball. We expect to include such sport-specific prior knowledge into the framework in our future work.

Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, at http://dx.doi.org/10.1016/j.cviu.2013.11.010.

References

[1] R. Cavallaro, The Foxtrax Hockey puck tracking system, IEEE Comput. Graph. Appl. 17 (1997) 6–12.

[2] K. Zhang, L. Zhang, M.H. Yang, Real-time compressive tracking, in: European Conference on Computer Vision, 2012.
[3] J. Lafferty, A. Mccallum, F. Pereira, Conditional random fields: probabilistic models for segmenting and labeling sequence data, in: International Conference on Machine Learning, 2001.
[4] J. Berclaz, F. Fleuret, P. Fua, Pom: Probabilistic Occupancy Map, 2007. <http://cvlab.epfl.ch/software/pom/index.php>.
[5] J. Berclaz, F. Fleuret, E. Türetken, P. Fua, KSP: Multiple Object Tracker Using K-Shortest Paths, 2011.<http://cvlab.epfl.ch/software/ksp/index.php>.
[6] Hawk-Eye Tennis Officiating System, 2012. <http://www.hawkeyeinnovations.co.uk/page/sports-officiating/tennis>.
[7] SportVU Hoops Solutions, 2012. <http://www.sportvu.com/basketball.asp>.
[8] V. Lepetit, A. Shahrokni, P. Fua, Robust data association for online applications, in: Conference on Computer Vision and Pattern Recognition, 2003.
[9] F. Yan, W. Christmas, J. Kittler, Layered data association using graph-theoretic formulation with applications to tennis ball tracking in monocular sequences, IEEE Trans. Pattern Anal. Mach. Intell. 30 (10) (2008) 1814–1830.
[10] V. Lumikero, Football Tracking in Wide-Screen Video Sequences, Master's thesis, KTH, 2004.
[11] Y. Ohno, J. Miura, Y. Shirai, Tracking Players and Estimation of the 3D Position of a Ball in Soccer Games, in: International Conference on Pattern Recognition, 2000.
[12] M. Leo, N. Mosca, P. Spagnolo, P. Mazzeo, T. D'Orazio, A. Distante, Real-time multiview analysis of soccer matches for understanding interactions between ball and players, in: Conference on Image and Video Retrieval, 2008.
[13] J. Ren, J. Orwell, G.A. Jones, M. Xu, Real-time modeling of 3D soccer ball trajectories from multiple fixed cameras, IEEE Trans. Circ. Syst. Vid. Technol. 18 (3) (2008) 350–362.
[14] Y. Zhang, H. Lu, C. Xu, Collaborate ball and player trajectory extraction in broadcast soccer video, in: International Conference on Pattern Recognition, 2008.
[15] F. Poiesi, F. Daniyal, A. Cavallaro, Detector-less ball localization using context and motion flow analysis, in: International Conference on Image Processing, 2010.
[16] M. Perše, M. Kristan, S. Kovačič, J. Perš, A trajectory-based analysis of coordinated team activity in a basketball game, Comput. Vis. Image Und. 113 (5) (2009) 612–621.
[17] K. Kim, D. Lee, I. Essa, Detecting regions of interest in dynamic scenes with camera motions, in: Conference on Computer Vision and Pattern Recognition, 2012.
[18] K. Okuma, A. Taleghani, N. de Freitas, J. Little, D. Lowe, A boosted particle filter: multitarget detection and tracking, in: European Conference on Computer Vision, 2004.
[19] W.-L. Lu, K. Okuma, J.J. Little, Tracking and recognizing actions of multiple hockey players using the boosted particle filter, Image Vis. Comput. 27 (2009) 189–205.
[20] F. Li, R.J. Woodham, Video analysis of hockey play in selected game situations, Image Vis. Comput. 27 (2009) 45–58.
[21] C. Direkoglu, N.E. O'Connor, Team activity recognition in sports, in: European Conference on Computer Vision, 2012, pp. 69–83.
[22] A. Gupta, P. Srinivasan, J. Shi, L.S. Davis, Understanding videos, constructing plots learning a visually grounded storyline model from annotated videos, in: Conference on Computer Vision and Pattern Recognition, 2012–2019, 2009.
[23] T. Lan, L. Sigal, G. Mori, Social roles in hierarchical models for human activity recognition, in: Conference on Computer Vision and Pattern Recognition, 2012.
[24] R. Li, R. Chellappa, S. Zhou, Learning multi-modal densities on discriminative temporal interaction manifold for group activity recognition, in: Conference on Computer Vision and Pattern Recognition, 2009.
[25] S. Ali, V. Reilly, M. Shah, Motion and appearance contexts for tracking and re-acquiring targets in aerial videos, in: Conference on Computer Vision and Pattern Recognition, 2007.
[26] H. Grabner, J. Matas, L. Van Gool, P. Cattin, Tracking the invisible: learning where the object might be, in: Conference on Computer Vision and Pattern Recognition, 2010.
[27] L. Karlinsky, M. Dinerstein, D. Harari, S. Ullman, The chains model for detecting parts by their context, in: Conference on Computer Vision and Pattern Recognition, 2010.
[28] H. BenShitrit, J. Berclaz, F. Fleuret, P. Fua, Multi-commodity network flow for tracking multiple people, IEEE Trans. Pattern Anal. Mach. Intell, 2014.
[29] N. Oliver, B. Rosario, A. Pentland, A Bayesian computer vision system for modeling human interactions, IEEE Trans. Pattern Anal. Mach. Intell. 22 (8) (2000) 831–843.
[30] P. Parisot, C.D. Vleeschouwer, Graph-based filtering of ballistic trajectory, in: International Conference on Multimedia and Expo, 2011.
[31] A Matlab Implementation Random Forest Classifier, 2012. <http://www.mathworks.com/matlabcentral/fileexchange/31036-random-forest>.
[32] APIDIS European Project FP7-ICT-216023, 2008–2010. <www.apidis.org>.
[33] T. D'Orazio, M. Leo, N. Mosca, P. Spagnolo, P.L. Mazzeo, A semi-automatic system for ground truth generation of soccer video sequences, in: International Conference on Advanced Video and Signal Based Surveillance, 2009, pp. 559–564.
[34] B. Babenko, M.H. Yang, S. Belongie, Robust object tracking with online multiple instance learning, IEEE Trans. Pattern Anal. Mach. Intell. 33 (8) (2011) 1619–1632.
[35] Play basketball roulette!, 2013. <http://cvlabwww.epfl.ch/ablavsky/bball-roulette/>.