ABSTRACT

In the context of team sport events monitoring, the various phases of the game must be delimitated and interpreted. In the case of a basketball game, the detection and the tracking of the ball are mandatory. This paper deals with the detection of the ballistic trajectory of a ball thrown between two players or toward the basket. Ballistic trajectories build on the 3D ball candidates previously detected at each timestamp. The proposed method is based on a graph, for which the nodes are the observed velocity between two 3D candidates, the edges link two nodes that have a common 3D candidate and the cost function is proportional to the distance between the observed acceleration and the gravity one. Hence, looking for the shortest path in this graph is equivalent to searching for 3D candidates that follow a ballistic trajectory. It appears to be both quite efficient and effective.

Index Terms—Detection, ballistic trajectory, foreground mask, graph, line graph

1. INTRODUCTION

Ball detection and tracking appear to be a key component of sport monitoring systems. In tennis or soccer games, particle filtering methods are involved to track the ball [1, 2], which is visible most of the time. In contrast, in sports like basketball, the ball is often occluded by players and an approach that is based on independent detections of the ball at each timestamp appears to be more relevant. Considering frames independently, a detection algorithm always yields an important rate of false positives. A post-processing is then useful. As highlighted in [3, 4], the analysis of the candidates trajectory helps in discriminating between true and false positives. A post-processing is then useful. As highlighted in [3, 4], the analysis of the candidates trajectory helps in discriminating between true and false positives, since the ball is supposed to follow a ballistic trajectory, which is not the case of most of the false detections.

Specifically, Chen et al. [3] only consider the shooting parts of the game, and thus are focused on the basket area. They consider the problem in a single view setting, and detect the ball on each individual frame. They first look at any three consecutive 2D candidates not too far from each other (velocity constraint), and approximate their neighborhood with a ballistic trajectory. They then link the trajectories and remove the false ones based on a graph formalism. Each trajectory defines a node, and there is an edge between two nodes if their corresponding time intervals are not too much disjuncted. Edges costs depend to the configuration of the two connected trajectories (overlapping or not, with conflict or not), and the shortest path is supposed to extract the best global trajectory.

We address the problem in a multiview setting. Candidates at each timestamp are thus defined in 3D, so that actual ballistic trajectories can be searched for. This is in contrast with [3] and [4], which search for parabolic trajectories (= projection of 3D ballistic ones) in 2D settings. In addition, our use case has two main specificities. The first one is that we have few 3D ball detections along the time, thus the local definition of trajectories as in [4] is difficult. The second one is that the 3D ball candidates positions are quite noisy and inaccurate. This is due to the triangulation part of the ball detection, which relies on noisy 2D candidates positions computed in loosely synchronized cameras. The lack of accuracy and reliability resulting from the detections prevents fine matching of parabolic models, as considered in [3]. Our framework has been designed to overcome these limitations.

We consider the shooting parts of the game as well as any pass (long enough) between two players. Our ballistic trajectory detection is based on a graph, that we call velocity graph. In fact, each node represents the observed velocity between two 3D candidates and each edge the locally observed acceleration of the 3D candidates. In the case of a ballistic trajectory, the ball acceleration should be constant. Thus, detected ballistic trajectory in this graph corresponds to find the shortest path by applying Dijkstra’s algorithm for example.

Section 2 surveys the ball detection algorithm. In Section 3, the graph formalism and the ballistic trajectory detec-
2. BALL DETECTION

Prior to ballistic trajectory analysis, 3D ball candidates have to be located. The main principles of our 3D ball detection method are summarized in Figure 1 and described below.

![Fig. 1. Framework of multi-view ball detection](image)

As the basketball match is observed by several (loosely) synchronized and calibrated cameras, we consider the fusion of the partial decisions taken in each view. On each view, a set of potential 2D candidates are selected, based on cleaned-up foreground mask analysis. In short, we remove the connected components that are too big to be the ball (thus bigger than \( \tau_{big} \) pixels), and then remove non moving pixels by computing the difference between the foreground masks of two consecutive frames. Finally we remove connected components that are too small (smaller than \( \tau_{small} \) pixels) and the ones with an aspect ratio that do not correspond to a ball (higher than the threshold \( \tau_{\lambda} \)). The potential 2D candidates are simply located at the barycenter of the remaining connected components. Pairs of candidates are then triangulated to define 3D candidates, whose relevance with respect to other views is checked.

From this ball detection algorithm, the rate of true detection is around 55% (see Table 1), and it can be observed that:

- there are true and false positive 3D candidates;
- at some timestamps, there is more than one 3D candidate.
- at many other timestamps, there is no 3D candidate.

Finally, we have to deal with these specificities: few detections and false positives detections. We thus have to discriminate true from false positive 3D candidates. To do this, we propose to analyze the trajectory followed by plausible ball candidates. This is done efficiently through the graph-based approach described in the next section. Only ballistic trajectories are kept as eligible ball candidates.

3. GRAPH-BASED BALLISTIC TRAJECTORY DETECTION

Our temporal analysis is based on a graph, which analyzes the ballistic nature of the motion vectors induced between consecutive 3D ball candidates.

3.1. Ballistic trajectory

Neglecting the friction forces, a ball is only subject to the gravity force, inducing the acceleration \( \ddot{\mathbf{r}} = [0 \ 0 \ -g_z]^T \) with \( g_z = 9.81 \text{ m/s}^2 \). The acceleration of the ball is thus constant and the ballistic trajectory of a thrown ball is a parabola.

3.2. Position graph

The position graph \( \mathcal{P} = (\mathcal{N}_P, \mathcal{E}_P) \) (see figure 2) is a directed graph characterized by:

- its set of nodes \( \mathcal{N}_P \):

\[
\mathcal{N}_P = \{n_i\}_{i \in [1,N]} \tag{1}
\]

where the nodes \( n_i \) (\( N \) being the number of nodes) are the 3D positions \( P_t = [x_t \ y_t \ z_t]^T \in \mathbb{R}^3 \) of the ball candidates obtained by the detection algorithm presented in Section 2.

- its set of edges \( \mathcal{E}_P \):

\[
\mathcal{E}_P = \{e_{i,j} \mid n_i \in \mathcal{N}_P \\
& \& n_j \in \mathcal{N}_P \\
& \& f(i) < f(j) \\
& \& f(j) \leq f(i) + F_{max} \\
& \& ||P_j - P_i||_2 \leq (f(j) - f(i))D_{max}\} \tag{2}
\]

where \( f \) is a function that gives, for each candidate \( n_i \), its corresponding timestamp number \( f(i) \) (for example, on Figure 2, \( f(i) = f(7) = t + 3 \) and \( f(5) = t + 1 \), \( F_{max} \) is the maximal number of frames for which the connections are contemplated and \( D_{max} \) is the maximal displacement between two consecutive timestamps. \( D_{max} \) is supposed to represent the maximal velocity of a thrown ball.

In the previous equation (2), the third condition means that we are interested in forward temporal analysis. The fourth condition is here to be robust against missed detection at some timestamps. This ability to create connections on a long horizon (of more than one timestamp) makes the algorithm less sensitive to the detection inaccuracies. Finally the fifth condition means that the connections are considered if and only if the distance between candidates is smaller than the maximal allowed displacement, which is function of their corresponding timestamp numbers.

Each edge \( e_{i,j} \) supports the definition of a velocity vector \( \mathbf{v}_{i,j} \):

\[
\mathbf{v}_{i,j} = \frac{P_j - P_i}{(f(j) - f(i))\Delta}, \tag{3}
\]

where \( \Delta \) is the time interval in seconds between two consecutive timestamps.
3.3. Velocity graph

The velocity graph \( \mathcal{VG} = (\mathcal{NV}, \mathcal{EV}) \) is the directed line graph [5] of the position graph \( \mathcal{PG} \) (see Figure 3). It is defined by:

- its set of nodes \( \mathcal{NV} \), defined to be the set of edges of \( \mathcal{PG} \):
  \[
  \mathcal{NV} = \mathcal{EP},
  \]
  (4)
  A velocity vector is associated (see equation 3) to each element of \( \mathcal{NV} \);
- its set of edges \( \mathcal{EV} \), given by:
  \[
  \mathcal{EV} = \{e_{i,j,k} \mid e_{i,j} \in \mathcal{NV} \land e_{j,k} \in \mathcal{NV}\}_{i,j,k \in [1,N]}.
  \]
  (5)

For each edge \( e_{i,j,k} \), an acceleration vector \( \vec{a}_{i,j,k} \) is associated:

\[
\vec{a}_{i,j,k} = \frac{\vec{v}_{j,k} - \vec{v}_{i,j}}{(f(k) - f(i)) \Delta f}.
\]
(6)

3.4. Ballistic trajectory from the velocity graph

As a ball has a constant acceleration, looking for a ballistic trajectory is equivalent to finding path in the velocity graph for which the observed acceleration is constant and close to the gravity acceleration.

For any edge \( e_{i,j,k} \) of the velocity graph \( \mathcal{VG} \), the cost \( c_{i,j,k} \) is thus given by:

\[
c_{i,j,k} = \alpha_{i,j,k} ||\vec{a}_{i,j,k} - \vec{g}||_2, \tag{7}
\]
where \( \alpha_{i,j,k} = f(k) - f(i) \) is a coefficient such that any edge has a cost weighted by the time interval between detections \( n_i \) and \( n_k \). If the 3D points follow a perfect ballistic trajectory, then the cost is equal to zero. Thus looking for a ballistic trajectory is equivalent to finding the shortest path in this velocity graph.

The shortest paths and their costs between all pairs of nodes \( e_{i,j} \) and \( e_{k,l} \) (when a path exists) in the velocity graph \( \mathcal{VG} \) can be computed with Dijkstra’s algorithm. As the time duration of any paths can be different, the path costs are normalized with their time duration. The total time duration of a shortest path between the nodes \( e_{i,j} \) and \( e_{k,l} \) (with \( f(j) \leq f(k) \)) is given by \( f(k) + f(l) - (f(i) + f(j)) \). To favour long (in terms of time durations) shortest path, each path cost is normalized by its total time duration at a power \( \beta \), where \( \beta > 1 \).

Paths shorter than a threshold are discarded. If longer paths exist, the smallest normalized cost shortest path is selected.

As the time duration of a thrown ball is usually not longer than two seconds, the velocity graph is processed over window periods of two seconds.

The resulting algorithm is presented in algorithm 1.

```
for each period of two seconds of the match (these periods start every second)
  build the velocity graph \( \mathcal{VG} \) (adjacency and cost matrix)
  apply Dijkstra’s algorithm
  remove paths that are not long enough in terms of time duration
  normalize each path cost
  select the shortest path of minimal cost if small enough
end for
```

Algorithm 1: Ballistic trajectory detection algorithm based on a velocity graph

4. EXPERIMENTAL RESULTS

Results are provided with respect to the basketball dataset of the APIDIS project (http://www.apidis.org/Dataset). The basketball match has been captured by \( K (= 7) \) loosely synchronized and calibrated cameras at 25 frames per second.
4.1. Data

The values of the parameters are the following: $\tau_\lambda = 3$, $\tau_{\text{big}} = 25 \times 25 \times \pi$ pixels, $\tau_{\text{small}} = 100$ pixels, $\Delta = 1/25$ second, $D_{\text{max}} = 1$ meter, $F_{\text{max}} = 5$ and $\beta = 3$. The results have been computed over the minute of the match starting at 18h47’00”, that is for 1501 frames. By our ball detection approach, we should be able to detect it at 261 timestamps.

4.2. Results

Figure 4 shows for two periods of two seconds (one where there is a thrown ball, the other not), the projection on a view of the detected 3D ball candidates and of the ones following a ballistic trajectory detected by the proposed graph-based approach. On each image, all ball candidates that are selected within a two seconds period are depicted. To visualize the temporal succession of candidate detections, the radius of the circle representing the location of a given candidate increases with time. Blue and magenta circles represent the ground truth. Blue corresponds to the positions for a throw or a pass, thus when the ball detection algorithm is able to detect the ball and when the ball should follow a ballistic trajectory. On this figure, in green, we see that ballistic trajectories are well detected.

Table 1 shows the number of true $C^+$ and false $C^-$ positive detection before and after the temporal analysis as well as the rate of detections $r^+$ and $r^-$ (mean number of detections) by timestamp with detections. It shows that almost all false positive ball detections are removed and almost all true positive detections are preserved.

<table>
<thead>
<tr>
<th>3D ball detection</th>
<th>Ballistic Trajectory</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(C^+, r^+)$</td>
<td>$(C^-, r^-)$</td>
</tr>
<tr>
<td>139, 1.41</td>
<td>255, 1.38</td>
</tr>
<tr>
<td>$(C^+, r^+)$</td>
<td>$(C^-, r^-)$</td>
</tr>
<tr>
<td>98, 1</td>
<td>19, 1</td>
</tr>
</tbody>
</table>

Table 1. Numbers and rates of true and false positive detections for the 3D ball detection part and the ballistic trajectory filtering one.

5. CONCLUSION

In this paper, we have presented a temporal analysis of 3D ball candidates, and we have selected the ones following a ballistic trajectory. The approach effectively discriminate between true positive and false positive candidates. Our approach is computationally efficient, and appears to work even in case of sporadic detections.

6. REFERENCES


