www.czasopisma.pan.pl PAN www.journals.pan.pl

Opto-Electronics Review 26 (2018) 296–306

Contents lists available at ScienceDirect

Opto-Electronics Review

journal homepage: http://www.journals.elsevier.com/opto-electronics review

Accurate ball tracking in volleyball actions to support referees

P. Kurowski *, K. Szelag, W. Zaluski, R. Sitnik

Warsaw University of Technology, Faculty of Mechatronics, 8 Św. Andrzeja Boboli St. 02-525, Warsaw, Poland

ARTICLE INFO

Article history:
Received 10 September 2018
Accepted 22 October 2018
Available online 8 November 2018

Keywords:
Object recognition
Object tracking
Object trajectory
Light intensity changes
Volleyball

ABSTRACT

This paper proposes a method for offline accurate ball tracking for short volleyball actions in sport halls. Our aim is to detect block touches on the ball and to determinate accurate trajectory and impact positions of the ball to support referees. The proposed method is divided into two stages, namely training and ball tracking, and is based on background subtraction. Application of the Gaussian mixture model has been used to estimate a background, and a high-speed camera with a capture rate of 180 frames per second and a resolution of 1920×1080 are used for motion capture. In sport halls significant differences in light intensity occur between each sequence frame. To minimize the influence of these light changes, an additional model is created and template matching is used for accurate determination of ball positions when the ball contour in the foreground image is distorted. We show that this algorithm is more accurate than other methods used in similar systems. Our light intensity change model eliminates almost all pixels added to images of moving objects owing to sudden changes in intensity. The average accuracy achieved in the validation process is of 0.57 pixel. Our algorithm accurately determined 99.8% of all ball positions from 2000 test frames, with 25.4 ms being the average time for a single frame analysis. The algorithm presented in this paper is the first stage of referee support using a system of many cameras and 3D trajectories.

1. Introduction

Sports video analysis has become an important topic owing to its wide spectrum of applications and potential commercial benefits. For most applications a ball is the focal point of interest, making ball detection and tracking the crucial elements of these systems. The most well-known sport video analysis application is Hawk-Eye system. It supports referees by determining 2D and 3D trajectories of a ball and providing animated replays of ball–court impact in tennis, volleyball and badminton, and while also utilizing detailed match statistics. Ball and player tracking can be used as aids in training and tactical analysis, with another system capability being the support of camera systems during broadcasts to assure the best transmission quality without losing ball image.

For various reasons, ball tracking in sport video is a troublesome task. Automatic ball tracking requires accurate determination of ball position but ball motion blur, the movement of the players and fans, and changing environmental conditions impede many solutions to this problem. The degree of influence from these factors mentioned above can vary for different sports.

Recently, many algorithms have been developed to track ball motion in sports video. Many of these first use colour thresholds to estimate field region; this is especially true for soccer [1–3]. The next stage of these algorithms is to detect lines, players, and referees, along with image noises based on object features like size, colour, and shape. There are often several possible ball candidates in images when using these methods. Potential trajectories are created using Kalman filter verification [4]. Finally, a real ball trajectory is chosen from all potential trajectories. This approach can be used only in sports with a large field area and under assumptions of no changes in field colour range and lack of similarity between ball and field colours. A similar algorithm proposed by Seo [5] is based on field estimation but uses template matching to find both the ball and players on video. Since finding the ball in an image is more difficult than finding players, the position and bounding box of the ball are manually initialized at the start time. Estimation of field region by colour threshold is inadequate to conditions in sport halls.

Some algorithms used machine learning to track ball. Ji [6] proposed algorithm based on Adaboost. He tracked table tennis ball with low speed camera. For feature extraction he used Local Gradient Patterns. Speck [7] used Convolutional Neural Networks to track ball in Robocup Soccer. However, to create accurate algorithm for ball tracking with the use of machine learning the training dataset should be very large and containing many possible ball images,

* Corresponding author.
E-mail address: p.kurowski@mchtr.pw.edu.pl (P. Kurowski).

https://doi.org/10.1016/j.opelre.2018.10.004
1230-3402/

including occlusion and other possible scenarios during a volleyball match. In addition, each ball (with other colours or pattern) should have its own dataset.

Liang [8] proposed an algorithm based on ball colour, however this algorithm is not resistant to external conditions that change ball colour, i.e., ball blur.

The algorithm proposed by D'Orazio [9] uses directional Circle Hough transform to track the ball in a soccer game. Their results are inaccurate when the ball is occluded or there are objects similar to the ball on video. Additionally, ball blur also impedes ball detection.

Some ball tracking algorithms are based on background subtraction. The main idea of these algorithms is to estimate the background of a video, and on the basis of this background, to detect the foreground (in other words, detect changes in the video). There are many methods of background subtraction, and the easiest and fastest of these is to calculate the differences between all pixels of two images. Chen [10] based his algorithm on simple frame differencing. After movement detection, he chose ball candidates by checking object features. On each image there could be several ball candidates, so a few possible trajectories are first generated, and at the end of the video, the final trajectory is chosen. Trajectory is approximated with a second degree polynomial. Audio event detection (referees' whistles or players' spike sounds) is added for more accurate results. A similar approach is proposed in another algorithm [11], but ball trajectory is described using a physical model. Chen [12] uses the *Positive Frame Difference Image* method to track the movement and trajectory of a baseball. In this algorithm, frame differencing is based on the sign and value of pixel difference. This is because a baseball bright white and the intensity of the ball in a frame should be higher than that of the background pixels. Several algorithms [13,14] use variation of frame difference method based on *three-frame differencing* [15]. Ohno [16] proposed an algorithm using frame differencing and field region estimation to track the ball in a soccer game; however, this algorithm tracks a player with the ball rather than ball itself.

Gomez [17] proposed an algorithm to track a beach volleyball based on the previously mentioned Chen [11] method, however their method tracked players, as well. They also proposed using frame differencing to determine ball positions on images to estimate background, and Gaussian mixture-based algorithm [18] to track players. A Gaussian mixture model background estimation algorithm is useful as it is adaptive to dynamic changes such as camera motion and changes in light intensity.

Ekinci [19] used a median filter to create a background image to detect a tennis ball on video. In this method, each background pixel is calculated as a median value of all frames.

An algorithm with a double approach is proposed by Ariki [20] to detect the position of a soccer ball. First, background subtraction is performed. Next, the image is compared with a ball template using cross-correlation. The ball template is prepared manually in advance. This method is called *global search*. When previous ball coordinates are known, ball positions are searched by using a particle filter. This stage of the algorithm is called *local search*. When the ball is lost on video, *global search* begins again.

Many of the tracking algorithms are based on Particle Fitler. Cheng [21] proposed approach with ball size adaptive tracking window, a ball feature likelihood model and an anti-occlusion likelihood measurement based on Particle Filter for improving the accuracy. However, deleting some camera information with low likelihood makes the particles unstable [22]. Approaches based on Particel Filter are much less accurate when size of the ball becomes very small or when its colour appears different from original, because the number of particles reduces drastically which leads to failure of the tracker. When the ball merges with similar appearing colour background or player, the particles assigned to the ball are distributed over both the regions.

Wang [23] used *eigen-background subtraction* [24] to extract the foreground pixels for ball tracking in several sports. He presented an approach whereby players are tracked first, an algorithm decides which player is in possession of the ball, and player trajectories are then used to achieve real ball tracking. Maksai [25] proposed a generic algorithm to modelling the interaction between the ball and the players while also imposing appropriate physical constraints on the ball's trajectory. To detect the ball, he used an SVM [26] to classify image patches in each camera view based on Histograms of Oriented Gradients, HSV colour histograms, and motion histograms. To detect the players, he proposed algorithm based on a Probability Occupancy Map.

The algorithms listed above are not accurate enough to meet the criteria for effectively supporting referees in volleyball matches. These criteria are ball touch detection and ball impact placement (on or off the field). Excessive exposure time causes blur and extension of ball contour, and in these instances the determined position of centre of the ball is not accurate. Camera capture frequency should be high to minimize distance between two identified ball centres on two subsequent frames, as this increases the accuracy of detection of trajectory changes. Problems of motion blur, low camera frequency, and significantly noisy ball contour all affect final ball trajectories, and the increased influence of these factors decreases the precision of determined ball positions and the accuracy of referee support.

In this paper the authors propose a solution to the problem of offline accurate ball tracking tested on real short actions in volleyball videos. The goal of our research is to aid volleyball referees in making decisions concerning block touches and ball out events. To reach this goal (especially block detection, which is based on ball trajectory changes) determined 2D ball positions (and eventually 3D coordinates calculated from 2D positions) must be highly accurate, the number of frames for which ball positions were not designated must be low. The success of this approach is hindered by specific environmental conditions that occur in sport halls, especially light intensity changes.

The novelty of our approach is the application of a Gaussian Mixture Model [18] background estimation algorithm enriched in order to reduce noise in foreground images. This algorithm reduces the effect of light intensity changes and was validated on both simulated and real sequences.

The proposed ball tracking algorithm uses only 2D image sequences from one camera. Our goal is to support referees using a system comprised of cameras placed along the volleyball court, and 3D trajectories determined from the 2D image sequences mentioned above. Each camera will independently analyse 2D sequences using the proposed algorithm.

2. System design

A stationary camera is situated on the side of the court at a height of 2.5 m and a distance of 5 m from the external line of the court.

5 m is the minimum distance in accordance with volleyball regulations [27], and a greater distance would cause less accurate determination of ball positions. The ball has a suitable surface for high accuracy tracking. The camera's field of view captures half of the volleyball court. The camera location is shown in Fig. 1.

Our system must accurately detect slight changes in ball trajectory for reliable block detection. To avoid influence of motion blur, short exposure times were employed. The maximum recorded ball speed was 130 km/h., and exposure time was set to 0.7 ms. 2.5 cm is the maximum motion blur for these values, which is 12% of the ball diameter. In this case motion blur would not a significantly affect ball contour.

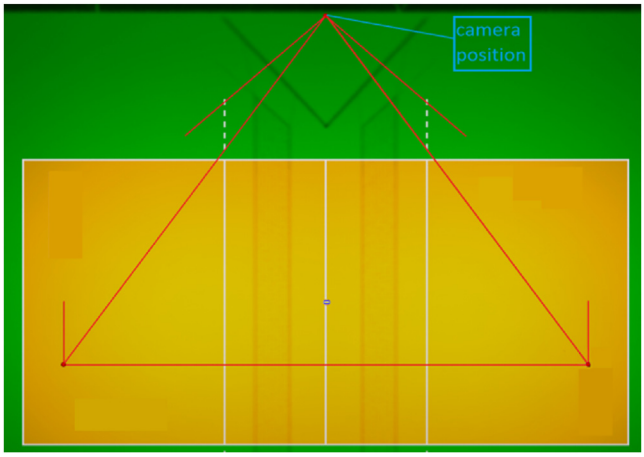

Fig. 1. Aerial view of the location of the camera and its field of view.

A high speed camera is used to enhance the precision of ball tracking. Because of the high speed of the ball, the camera frequency is set to 180 frames per second, as this speed produces a satisfactory relation between time of analysis and intervals between two subsequent ball positions. For this frequency, the maximum distance between ball positions in two subsequent frames was of 20 cm, which is equal to the ball diameter. This frequency also allows for intervals between balls on subsequent frames that allow for the detection of block touches. Larger distances would increase the likelihood of missing block touches. The camera captures images with a resolution of 1920×1080, and the ball can be clearly seen. With lower resolution, the accuracy of determined ball positions decreases owing to a less spherical ball shape and increased blur.

The established maximal average time of analysis of one frame is 30 ms, which is equal to an analysis of 180 frames (one second of video) in 6 s. This assumption allows the time of analysis of short volleyball actions to be short enough to aid referees online.

3. Ball tracking algorithm

The proposed algorithm is divided into two stages. The first stage is a training stage, where background and light change models are created based on N = 20 first frames of the analysed sequence, and background is estimated using a Gaussian mixture model (GMM) [18]. Calculated foregrounds are highly noisy because of light intensity changes present on each frame. These are caused by the flashing of fluorescent lamps proportional to the supplying current frequency. Image acquisition for tests was carried out with value of frames per second higher than the supplying current frequency in the sport hall. To eliminate this problem, an additional model for light changes was created. This additional model finds and removes pixels assigned to the foreground owing to their change of intensity on an image and leaves pixels related to moving objects.

In the second stage of the algorithm, resulting foreground images are analysed for ball detection. This analysis is based on contour, shape, size, colour and previous ball positions. The ball trajectory is then updated, and ball positions are interpolated as needed. Only one ball position per actual frame is added to the trajectory. Using predicted positions allows for analysis only of a fragment of the next frame.

Template matching is carried out if there are previous ball positions, but the ball was not detected on the current image. The used ball template is determined automatically from previous frames. The block diagram of the proposed method is shown in Fig. 2.

3.1. Background subtraction and light changes model

For background estimation a Gaussian mixture model [18] algorithm was employed. This algorithm is adaptive to dynamic changes of environmental conditions, however fluorescent lamps are used in many sport halls and flashing of these lamps causes uneven illumination of the pixels on processed images. The intensity of the pixels changes with every frame, and as such every foreground image is very noisy even when using high number of Gaussian mixtures. Examples of intensity changes for one pixel are shown in Fig. 3. The intensities of pixels were calculated according to the following formula:

$$\mathrm{I} = 0.299 \cdot \mathrm{R} + 0.59 \cdot \mathrm{G} + 0.11 \cdot \mathrm{B} \tag{1}$$

The maximum difference of intensity between two consecutive frames is equal to 83. To solve problem of light intensity changes on consecutive frames, a novel model is proposed. This model is employed as a complement to estimated background. The proposed algorithm assumes that each pixel has only two possible ranges of values: bright range B and dark range D. These ranges contain information about previous values of Hue (H), Saturation (S) and Value (V) from the HSV colour space for each pixel. The model for each pixel of image has 12 values: 6 for bright range, and 6 for dark range. These values are maximal and minimal values of H, S, and V.

$$R(x,\ y) = \begin{cases} \langle H_{minb}, H_{maxb} \rangle ,\ \langle S_{minb}, S_{maxb} \rangle ,\ \langle V_{minb}, V_{maxb} \rangle \\ \langle H_{mind}, H_{maxd} \rangle ,\ \langle S_{mind}, S_{maxd} \rangle ,\ \langle V_{mind}, V_{maxd} \rangle \end{cases} \tag{2}$$

Values of $_{\mathrm{b}}$ index are the ranges for *bright range* and values for $_{\mathrm{d}}$ index are the ranges for *dark range*. The assumption of equality of S ranges between the *dark and bright ranges* were made for the simplicity of calculations, and this did not significantly affect the resulting model.

The light intensity changes model is created as follows:

1 For the first few frames, the average values of H, S and V are calculated for each pixel. These average values are then added as limits to all ranges of the light changes model.
2 For each new frame first S values of ranges are changed according to actual P(x, y) pixel values:

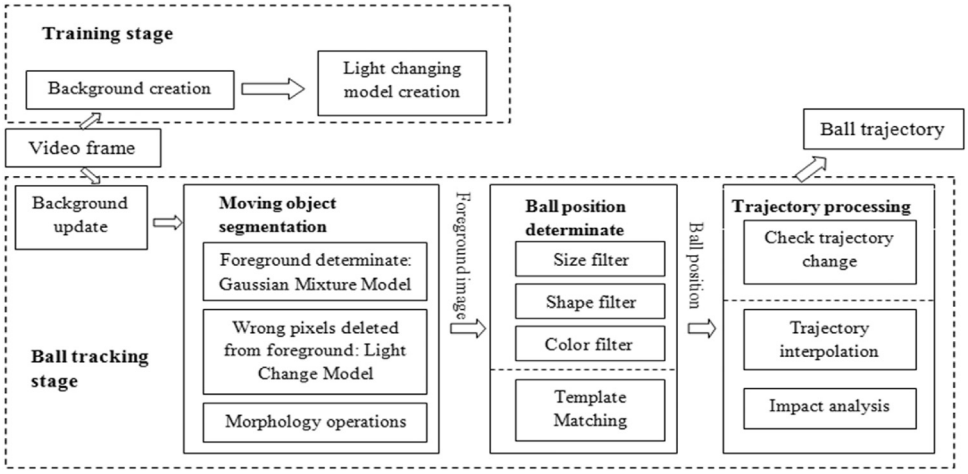

Fig. 2. Algorithm scheme.

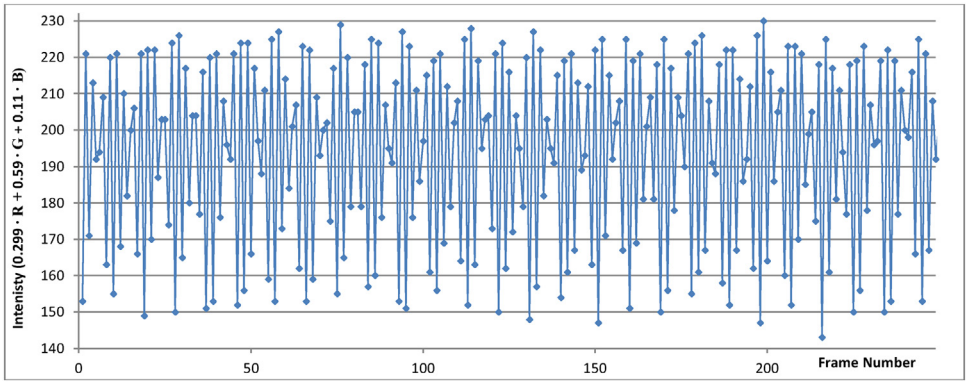

Fig. 3. Graph showing pixel intensity changes for fragment of the real sequence.

$$\begin{aligned} S(x,y)_{minb} &= P(x,y)_s,\ if\ P(x,y)_s < S(x,y)_{minb} \\ S(x,y)_{maxb} &= P(x,y)_s,\ if\ P(x,y)_s > S(x,y)_{maxb} \\ S(x,y)_{mind} &= P(x,y)_s,\ if\ P(x,y)_s < S(x,y)_{mind} \\ S(x,y)_{maxd} &= P(x,y)_s,\ if\ P(x,y)_s > S(x,y)_{maxd} \end{aligned} \tag{3}$$

3 In the next step, actual pixel P(x, y) is added to the proper range. Actual pixel value V is key to determining the range to which the pixel will be added. If pixel value V is lower than V_{maxd}, then the pixel is considered to be a *dark range* pixel. If pixel value V is higher than V_{minb}, then pixel is considered to be a *bright range* pixel.

$$\begin{cases} P(x,y) \in bright\ range,\ if\ P(x,y)_v > V(x,y)_{minb} \\ P(x,y) \in dark\ range,\ if\ P(x,y)_v < V(x,y)_{maxd} \end{cases} \tag{4}$$

4 If the actual pixel belongs to *bright range*, then the H and V values of *bright range* are updated according to the following equation:

$$\begin{cases} H(x,y)_{minb} = P(x,y)_h,\ if\ P(x,y)_h < H(x,y)_{minb} \\ H(x,y)_{maxb} = P(x,y)_h,\ if\ P(x,y)_h > H(x,y)_{maxb} \\ V(x,y)_{minb}\ not\ changed \\ V(x,y)_{maxb} = P(x,y)_v,\ if\ P(x,y)_v > V(x,y)_{maxb} \end{cases} \tag{5}$$

5 Similar to point 4, if the actual pixel belongs to the *dark range*, then the H and V values of the *dark range* are updated as follows:

$$\begin{cases} H(x,y)_{mind} = P(x,y)_h,\ if\ P(x,y)_h < H(x,y)_{mind} \\ H(x,y)_{maxd} = P(x,y)_h,\ if\ P(x,y)_h > H(x,y)_{maxd} \\ V(x,y)_{mind} = P(x,y)_v,\ if\ P(x,y)_v < V(x,y)_{mind} \\ V(x,y)_{maxd}\ not\ changed \end{cases} \tag{6}$$

6 If $P(x,y)_v$ is equal to $V(x,y)_{maxd}$ and $V(x,y)_{minb}$, then the H values for both ranges are updated using the first two equations from Eqs. 5 and 6.

The V_{maxd} and V_{minb} pixel values are equal from the first step of model creation. These values do not change in the model creation.

The light intensity changes model is only the additional utility for correcting foregrounds detected by background estimation algorithm. Comparing the model and foreground images allows for the removal of pixels assigned to foreground owing to their change of intensity. After background subtraction is performed, every pixel that is attached to the foreground image is checked according to the following equation:

$$\begin{cases} P(x,y) = 0,\ when\ P(x,y) \in B \vee P(x,y) \in D \\ P(x,y) = 1,\ otherwise \end{cases} \tag{7}$$

In the event of a small similarity between ball and background colours, all final model ranges can be increased by adding small values to them. Subsequently, the light changes model more effec-

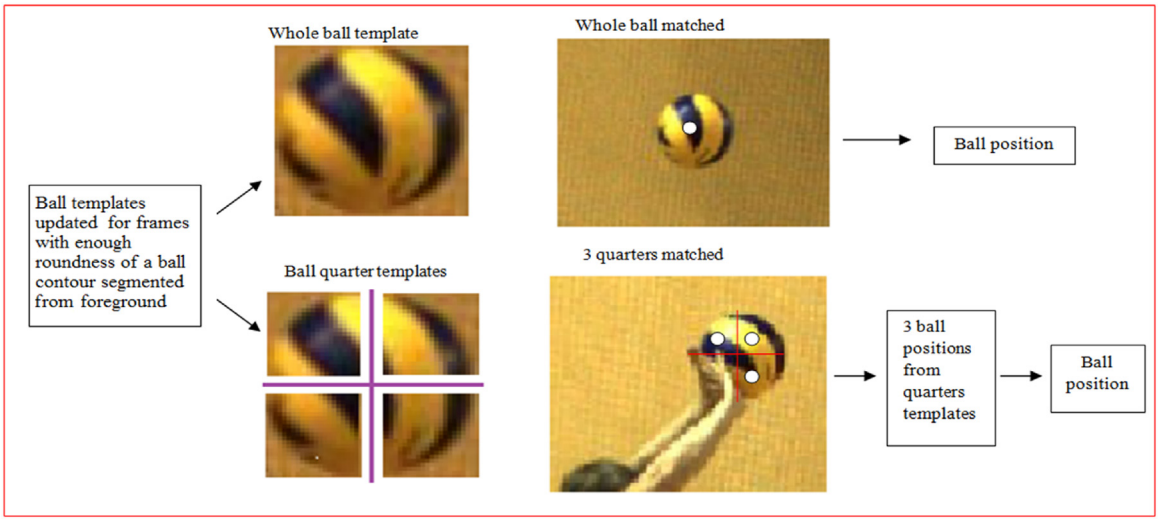

Fig. 4. Template matching example.

tively removes pixels in foregrounds that were added owing to their change of intensity.

The first N frames of videos are treated as training images for every new sequence. For videos with volleyball actions, N = 20 was adopted. After the training stage, the light intensity changes model is not updated to shorten processing time. For short volleyball videos N = 20 is enough to create an effective light changes model.

3.1.1. Ball position

Ordinarily, a few moving objects can be found on determined foreground images. Ball recognition is necessary to define ball position, therefore filters based on shape, size, and colour are employed to filter out segmented moving objects other than the ball. Only one ball position is determined for every frame, as this reduces analysis time.

3.1.1.1. Size filter. Owing to changes in distance between ball and camera, ball size differs in each frame. Ball size also changes when the ball is merged on a foreground image with another moving object, i.e., a player. Segmented moving objects are filtered out if their size is within the range $\langle S_{min}, S_{max} \rangle$, where S_{min} is the minimum and S_{max} is the maximum size. Range values were determined during test sequence processing.

3.1.1.2. Shape filter. A shape filter is based on the maximum threshold of the ratio between minimal area of circle defined by the contour of an object and a segmented moving object area. The threshold value was set as $T_R = 1.6$.

In most papers authors encounter the problem of the distorted shape of a ball caused by rapid motion. In our tests, a high-speed camera with frame rate of 180 frames per second and low exposure time is used. This virtually eliminates motion blur of a moving ball.

3.1.1.3. Colour filter. This filter is based on the HSV colour space. Colour calibration of the ball is determined based on a test sequence using a flying ball. Hue, Saturation, and Value maximum and minimum values are obtained as a result of this calibration. Each segmented moving object is checked by counting pixels with HSV values beyond the threshold range.

If, after the filtering process, there is more than one ball candidate object, then the contour with the smallest ratio described in the *Shape Filter* will be chosen as the ball. Other objects' positions are not tracked.

3.1.2. Template matching

On certain frames, the ball position cannot be determined. This can be caused by noise in foreground image or by another moving object which distorting the ball contour. In these situations template matching is employed. In the proposed algorithm, two types of templates are used. The first type is a whole ball template, and the second is comprised of templates of each quarter section of the ball, separated by horizontal and vertical lines. All templates are updated for every frame on which ball shape maintains the ratio threshold described in the *shape filter* below. Templates in this algorithm are determined as a rectangle fragment of images with ball in them.

The whole ball template is sought in an actual frame using the normalized cross correlation method. Ball position is determined as coordinates of pixels with the highest match value, and this value must be higher than the established threshold. Alternatively, every ball quarter template is matched in a frame. Only those quarter templates for which match value is higher than threshold are chosen. The ball center for each quarter template is calculated by using appropriate displacement of coordinates of the pixel with the highest match value. If there are less than two quarter templates for further analysis, then the ball position is not found in the actual frame. If there are more viable quarter templates, then the algorithm is checks for similarities between coordinates of the ball center calculated for every accepted template. If these similarities are sufficiently numerous, then ball position is calculated as the average value of the coordinates for the accepted quarter templates. This template matching example is shown in Fig. 4. Template matching is used only when ball positions are predicted for an actual frame, and this process is described in next paragraph.

4. Trajectory update

Trajectory is updated with each new ball position. Second degree polynomials are created using the least square method to approximate the flight path of the ball and the x and y coordinates of frame number (additional y(x) is used in block detection). The x coordinate corresponds to the column index of the camera image, and the y coordinate corresponds to a row number. Based on this, a polynomial ball position prediction is made. Because only one ball candidate is tracked, the amount of data from the frames to be analysed is reduced by using the region of interest in the images. The size of the region of interests is dependent on the average ball size in previous frames.

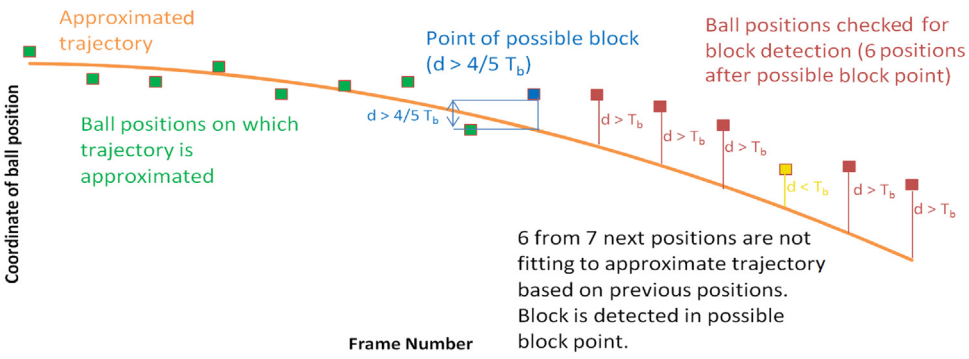

Fig. 5. Block analysis example. Blue point is considered as a block touch point.

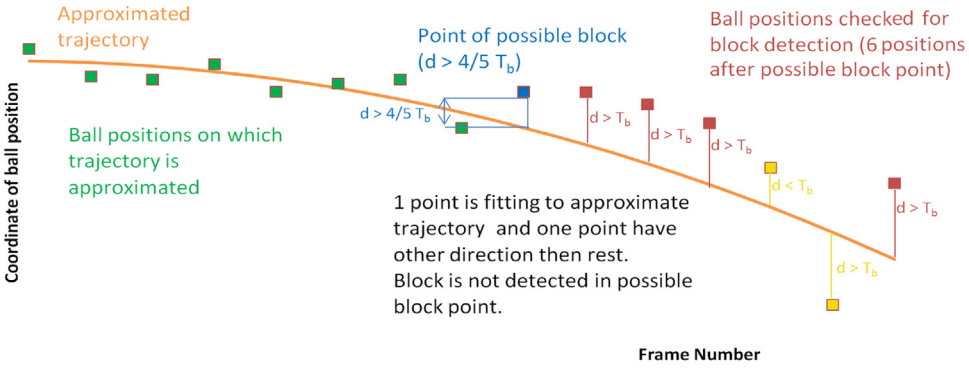

Fig. 6. Block analysis example. Blue point is not considered as a block touch point.

The ball flight path is divided to several polynomial approximations separated by sudden changes in trajectory. If new ball position lags behind calculated prediction coordinates, then this position is attached to new trajectory. If two trajectories have been incorrectly separated, they are compared and merged. When in any trajectory there are gaps, the ball positions are interpolated based on the determined polynomials. Impact coordinates are calculated as a point of polynomial intersection.

4.1. *Block touch detection*

Block touches can be detected from 2D trajectories. To determine whether the ball touched the blocking player, changes in ball trajectory are examined. A search for these changes begins only when the ball is near the net. Whole block analysis is carried out on original ball positions from foregrounds without interpolated and changed coordinates. A block touch is detected when more than α next ball original positions differ (in the same direction) from predicted positions with a value greater than the established threshold. For our tests we set the value of α as 7. For lower values, block detection is less accurate and repeatable. Higher values do not change accuracy of block detection. Extrapolated ball positions for frame gaps greater than 7 are sometimes inaccurate, even for actions without block touch. First, trajectory is created based on 7 original ball positions, as this is the minimum value for which approximated trajectory is accurate and stable enough to detect blocks. Next, each subsequent original position is compared with predicted positions from an extrapolation of this trajectory. If the original ball position fits this trajectory it is added as a continuation of flight of a ball. After adding this new (continuous) ball position to the current trajectory, new polynomials are calculated. If the new ball position does not conform to the original trajectory, it is checked as a possible instance of a block touch. Next 6 original positions are checked to fit with the original trajectory polynomials. If 5 of them do not fitting the polynomials, the conclusion is that a block touch has occurred. These original positions are compared to the positions extrapolated from the y(x) and y("frame number") polynomials. Only those polynomials were used, as the y axis of the camera was analogous to gravity in a real world scenario. Block touches change ball trajectory in this direction (in most cases). The estimated threshold T_b for the difference between predicted and real positions is set to 1.5 pixels. The first point of a block search has smallest threshold, equalling $\frac{4}{5}T_b$.

If, for a new ball position, one of coordinate functions [x(frame number) or y(frame number)] changes its derivative sign (and direction), then this point is considered an impact point and not a block point. In these cases new trajectory is created for the following 7 positions.

Block analysis is presented in Figs. 5 and 6. Green points are points on which the trajectory of the ball is approximated and then extrapolated. The orange line represents the approximated trajectory. Blue points are points at which the distance between one coordinate of the actual and extrapolated (predicted) points is higher than $\frac{4}{5}T_b$, and is therefore the point of a suspected block touch. Next, points are checked without trajectory approximation. In Fig. 5 five next points have distance greater than T_b value, yet only one lies near the approximated trajectory. This indicates as the occurrence of a block touch.

In Fig. 6, one point lies near the approximated trajectory and one has a distance higher then T_b, but is directionally opposite to the approximated trajectory when compared to other points. Two points are negating a block touch, therefore this point cannot be considered as the point of a possible block touch. As such, it is added to the ball trajectory, which is approximated anew and the next point is analysed.

Table 1
Types of noise simulated in each synthetic sequence.

Sequence number	Type of noise
1	No noise
2	Ball obscuration
3	Motion blur
4	Light intensity changes
5	Moving background
6	All type of noise listed above

5. Results and discussion

In order to assess the accuracy of the ball tracking method proposed in this paper, the authors compared it with other methods based on background subtraction. Algorithms based on frame differencing [10–14,16] and median filters [19] have been chosen as most accurate and commonly used solutions. We have tested both methods based on frame differencing, namely simple frame differencing [10–12,16] three-frame differencing [13,14]. The results for these methods are similar, however frame differencing has been proved to be more accurate. For this reason, we will only include results for this method. Our comparison is divided using two datasets, one real and one synthetic. Algorithms proposed by Chen *et al.* [11] and Ekinci *et al.* [19] have been chosen for comparison. For each tested algorithm, the filters described in the Ball Position section were used to filter segmented moving objects other than the ball. Similar filters are used in all methods.

Both algorithms chosen for comparison generate several potential ball trajectories, therefore the trajectory closest to the optimal ball trajectory is chosen. This approach is a result of many foreground contours potentially being recognized as the ball. For each test sequence, the only round moving object is the ball. There should be only one possible ball candidate in the foreground image. Therefore, an algorithm identifying the true ball trajectory from possible trajectories was not included in this comparison.

5.1. Synthetic data

In first test stage, the authors sought to confirm the accuracy of determined ball positions. Real sequence is lacking accurate reference ball positions, and manual determination of ball positions on images is encumbered with an error rate +/- 2 pixels. To solve this problem, 6 synthetic sequences with a flying ball were prepared. For frames of these sequences every ball position (in pixels) is known. On each sequence there are various types of noises, which are presented in Table 1.

The first synthetic sequence contains only a ball flying above the court. In the next sequence the ball is obscured either partly or completely by inserting rectangles in half of the frames. In to distinguish these rectangles from the background, they appear and disappear suddenly in sequence images. These rectangles obscure the ball in a manner similar to live action volleyball players. In the next video, motion blur is used to blur the shape of the ball. We use a motion blur equal to 12% of the diameter of the ball, and equivalent to a flying speed of 130 km/h on images from a live camera. In the fourth sequence, two light sources are added above the court. These sources change in intensity to mimic lighting conditions in a real sport hall. Changes of intensity were measured in a real sport hall and corresponding values were added to the synthetic sequence. Next, video with moving people is inserted as a background for the entire sequence. Another impediment in this sequence is background (video) colour that is very similar to one of three colours on the ball. For the final sequence, all noise types described above were added. Every prepared sequence has 200 frames and the ball is present on 170 of these frames. One frame of last synthetic sequence is shown in Fig. 7. In Tables 2 and 3,

Fig. 7. Example frame of synthetic sequence (grey rectangle is ball obscuration noise).

the results of these tests are presented. For all algorithms the light changes model was used and no of parameters were changed.

The results from synthetic data show that the algorithm proposed in this paper clearly has the highest rate of ball detection. Only in the second, fifth, and sixth sequences were some ball positions were not accurately determined from the images, and after trajectory analysis most of these were interpolated. In the second and sixth sequences, the ball was completely obscured in some images, making determination of its position impossible without interpolation. For the other compared methods fewer ball positions was determined, and this resulted in the inability to interpolate certain whole trajectories of ball flight. For the fifth and sixth sequences only, our algorithm determined enough ball positions to approximate whole ball trajectory.

Our results show that the differences between positional accuracy for tested algorithms were minimal, and our algorithm produced the smallest error average for both x and y coordinates in total. Determined ball positions for the sixth sequence produced the highest error values, and this was caused by the fusion of all noise types. The maximum error value of ball position for our algorithm was of 5.50, but these error values are rare and single. The highest error values for each sequence are for frames on which ball is near the image edge. It is caused by distortion of the modelled camera, and not a circular ball image due to camera perspective. The frame differencing method produces the smallest maximum error value. It is caused by very small number of determined ball positions for fifth and sixth sequences. Errors value for interpolated positions was higher in comparison to errors from ball positions determined directly from images, especially for bigger gaps between determined positions.

5.2. Real data: block detection

Block touch detection was checked in the second stage. Authors have chosen 12 real sequences where ball was fired with high speed from special cannon used in volleyball trainings. In front of the cannon man was standing with his hands up like in block action. This situation is presented in Fig. 8. On half of the sequences ball touches block and on the others ball flies near and above player fingers. At prepared sequence, if ball touched the player, it was only a slight contact with player fingers, not when a block touch occurs using the entire hand. 110 is the average amount of frames in these sequences. These sequences were recorded in sport hall where fluorescent lamps are used. A light intensity is differs for each frame. Block detection was carried out in forward and backward directions. For forward detection the beginning of block touch analysis

Table 2
Total results of tests for all synthetic sequences.

	Frame Differencing		Median Filter		GMM + Template Matching	
Ball positions determined	69.22%		96.27%		99.80%	
Ball detection from images	59.12%		66.96%		81.96% (17% from template matching)	
Coordinate	x coordinate	y coordinate	x coordinate	y coordinate	x coordinate	y coordinate
Average error [pixel]	0.79	0.50	0.90	0.45	0.85	0.30
Standard Deviation [pixel]	0.98	0.51	1.37	0.59	1.26	0.49

"Ball positions determined" is equal to percentage of ball positions determined from images or by interpolation and extrapolation.

Table 3
Results of tests of accuracy for each synthetic sequence.

Seq. Nr.		Frame Differencing		Median Filter		GMM + Template Matching	
	Coordinate	x coordinate	y coordinate	x coordinate	y coordinate	x coordinate	y coordinate
1	Avg. Er. / Max Er.	0.27 / 2.56	0.30 / 1.19	0.39 / 2.51	0.18 / 1.37	0.45 / 2.51	0.17 / 1.12
	B_{pd} / B_{pdi}	100 / 97		100 / 100		100 / 100	
2	Avg. Er. / Max Er.	0.59 / 3.26	0.81 / 3.59	0.69 / 3.69	0.78 / 3.57	0.49 / 2.71	0.36 / 3.19
	B_{pd} / B_{pdi}	100 / 51		100 / 51		100 / 90	
3	Avg. Er. / Max Er.	1.67 / 3.85	0.52 / 2.22	1.85 / 3.77	0.55 / 2.27	1.80 / 3.60	0.50 / 1.68
	B_{pd} / B_{pdi}	98 / 94		99 / 96		100 / 100	
4	Avg. Er. / Max Er.	0.34 / 2.59	0.31 / 2.44	0.44 / 2.75	0.26 / 2.15	0.41 / 2.59	0.21 / .27
	B_{pd} / B_{pdi}	100 / 94		100 / 96		100 / 100	
5	Avg. Er. / Max Er.	1.94 / 2.28	0.67 / 1.98	0.94 / 4.33	1.08 / 3.18	0.46 / 3.76	0.89 / 3.20
	B_{pd} / B_{pdi}	10 / 10		99 / 35		99 / 49	
6	Avg. Er. / Max Er.	3.33 / 3.59	0.44 / 1.48	2.20 / 6.49	0.87 / 3.06	1.88 / 5.50	1.12 / 4.26
	B_{pd} / B_{pdi}	7 / 5		78 / 24		99 / 53	

Avg. Er. – Average error in pixels; Max Er. – Maximal error in pixels; B_{pd} – Percentage of determined ball position; B_{pdi} – Percentage of determined ball position directly from images.

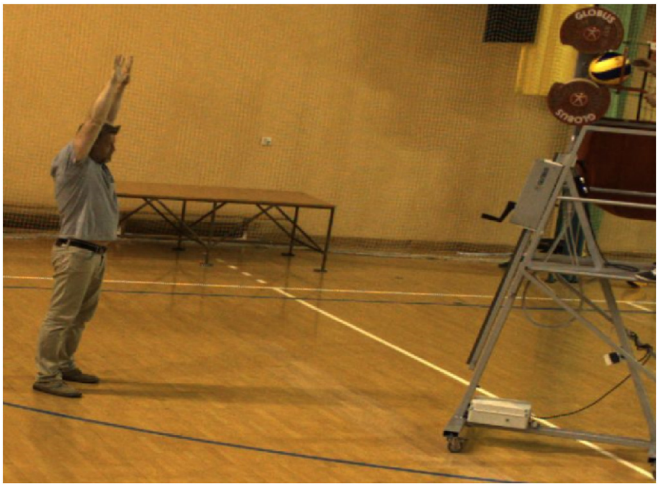

Fig. 8. Scene for block detection analysis.

is at the first determined ball position and for backward detection it is at the last ball position. A block touch is confirmed if both methods detect changes in trajectory. Situation where only one method of analysis detected a block touch are called *block suspicion*, which is regarded as non-detection of a block touch.

Results of block touch analysis for each algorithm are shown in Table 4. An additional light intensity changes model was not used for any method, including our own. Results with this model are presented in Table 5. We chose to use the additional light changes model for all algorithms. This allowed us to compare background subtraction methods as well as the light changes model.

Block detection for algorithms without the light changes model was ineffective, as the number of detected ball positions was too small. Only for our algorithm the number of determined ball coordinates on the image was high enough to begin block touch analysis. This was caused by using template matching. However, ball positions were too inaccurate to detect block touches correctly. After applying the light intensity changes model, the detection of both the ball and block touches greatly improved. Our method proved the most accurate, however the differences between the results for each method were minimal. Our method failed to detect a block touch in one sequence only.

The differences between predicted trajectory and determined trajectory for block detection were also quite small, and the average difference for the sixth position after block detection was equal to 2.8 pixels.

5.3. *Light changes' model*

To show the influence of the light changes' model, we presented results from the fourth synthetic sequence with and without the model in Table 6.

It is clear from comparing results that the light intensity changes model significantly increases detection of moving objects for short sequences. Most pixels that were added to foregrounds owing to the change of light intensity were removed. This improves ball detection efficacy and accuracy. Real moving objects are not removed from foreground images even when their colour is similar to the background colour.

For the frame differencing method, determination of threshold values for background subtraction was difficult. For small values, most pixels were added to the foreground images. For higher values, ball contour was very fragmented. In comparison, we have used small values.

Another problem in framed differencing was the imposition of a fragment of the ball with the same colour on the next two frames. This resulted in inaccurate determination of ball contour on foreground images for this method, as some pixels in the centre of the image were assigned as background pixels (pixel values change only slightly in the next two frames). For the algorithm presented in this paper and for the median filter method, fewer pixels were assigned to the foreground images owing to changes in their intensity. However, for frames with significant intensity changes most pixels were designated as a moving object. After light

Table 4
Results of block detection for algorithms without light intensity changes model.

	Frame Differencing	Median Filter	GMM + Template Matching	
Sequence with block touch	Correct √	To less detected ball positions to start block analysis	To less detected ball positions to start block analysis	4
	Miss × (block suspicion)	–	–	2 (1)
Sequence without block touch	Correct √	To less detected ball positions to start block analysis	To less detected ball positions to start block analysis	4
	Block Detection × (block suspicion)	–	–	2 (0)
Ball detection on images		6.57%	42.26%	76.37% (53.58)

"Ball detection on images" is equal to percentage of ball positive detection on images. The percentage of ball detection with participation of template matching is presented in brackets.

Table 5
Results of block detection for algorithms with light intensity changes model.

	Frame Differencing	Median Filter	GMM + Template Matching	
Sequence with block touch	Correct √	4	5	5
	Miss × (block suspicion)	2 (1)	1 (0)	1 (1)
Sequence without block touch	Correct √	5	5	6
	Block Detection × (block suspicion)	1 (3)	1 (2)	0 (3)
Ball detection on images		93.83%	94.58%	99.46% (2.76)

"Ball detection on images" is equal to percentage of ball positive detection on images. The percentage of ball detection with participation of template matching is presented in brackets.

Table 6
Percentage of determined ball positions from images for fourth sequence with and without light changes model.

	Frame Differencing	Median Filter	Mixture of Gaussians + Template Matching
With model	94	96	100 (4)
Without model	0	17	86 (52)

The percentage of ball detection with the participation of template matching is presented in brackets.

changes model filtering on foregrounds, only real moving objects remained visible. The effect of using an additional light changes model is shown in Fig. 9. The influence of each foreground creation step is presented in Fig. 10. These images were captured during tests on real sequences. For enhanced visibility, we showed only fragments of the original background and foreground images containing ball shape. After using region of interest, the number of pixels analysed decreased by 99,98%, from 1920 × 1080 to 66 × 66. However, the foreground image, after using region of interest, was still very noisy. Comparing the light changes model with the foreground image removed 1007 of the pixels added to foreground, reducing their number from 2496 to 1489. After using the light changes model ball shape became clear. It follows from this figure that using both region of interest and the light changes model improves the algorithm with respect to analysis time and accuracy of ball position.

5.4. Global system parameters

During our research we tested different system parameters. These parameters were: N – the number of training frames, T_R – the threshold of the ratio between minimum areas of circle defined as the contour of an object and segmented moving object, and T_b – the threshold of the difference between predicted and real positions in block detection.

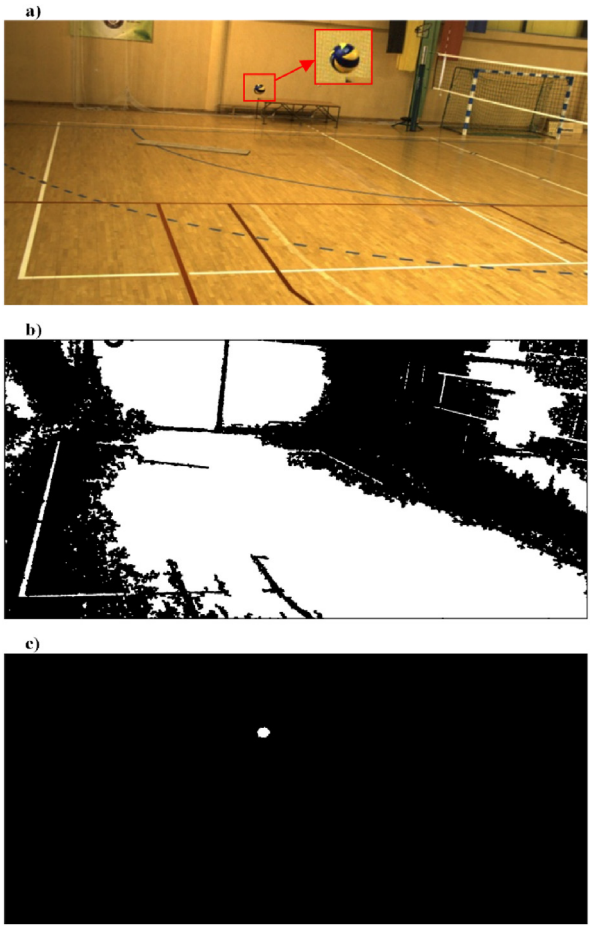

Fig. 9. Effect of light changes model use (a) Original frame from camera, (b) Foreground image after background subtraction, (c) Corresponding foreground image after background subtraction and analysis with light changes model - only ball shape is left.

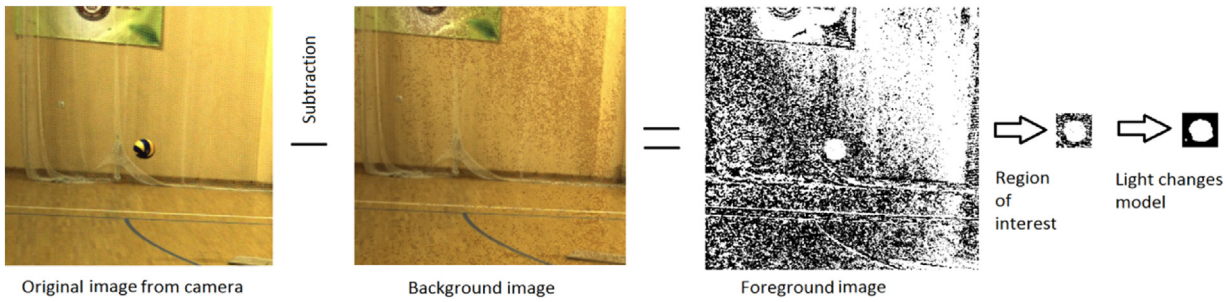

Fig. 10. Final foreground creation stages.

Table 7
Percentage of determined ball position directly from image for synthetic sequence with light intensity changes (sequence 4) for different number of training frames.

N	10	15	20	25	30
FD	64.7	90.6	93.5	93.5	93.5
MF	88.2	92.9	95.9	95.9	95.9
GMM	100(12.4)	100(7.1)	100(4.1)	100(4.1)	100(4.1)

FD – Frame Differencing algorithm, MF – Median Filter algorithm, GMM – Gaussian Mixture Model with Template Matching. Percentage of ball detection with participation of template matching is presented in brackets.

Table 8
Results of tests of accuracy for different TR values for algorithm based on Gaussian Mixture Model and template matching.

T_R	1.4	1.5	1.6	1.7	1.8
A. Er.	0.68	0.58	0.57	0.60	0.61
B_{pd}	71	87	100	100	100
B_{pdi}	61	77	82	86	89

A. Er. – Average error for both coordinates in pixels; B_{pd} – Percentage of determined ball position; B_{pdi} – Percentage of determined ball position directly from images.

Table 9
Percentage of correct block touch detection depending on Tb threshold for all of the tested algorithms with light changes model.

T_b	1.3	1.5	1.7
	69	83	75

Tests results for the influence of these parameters on the accuracy of ball (or block) detection are presented in Tables 7–9.

Results from Table 7 show that 20 is the optimal number of training frames. For smaller numbers of training frames, the light changes model removes fewer pixels added to foreground images, as previously mentioned. For training frame numbers of training frames and higher, there is no change in the percentage of ball detection.

The most accurate results were for $T_R = 1.6$. This value is optimal with respect to average error of ball positions and ball determination percentage. The percentage of determined ball positions (with interpolation) was the same for $T_R = 1.6$, 1.7, and 1.8. For value higher than 1.6 there is a risk of determining ball position when ball is partly obscured without using template matching. It can cause inaccurate ball position determinations, i.e. when the ball is obscured by fingers of player during a block, which would disturb ball trajectory in a key moment for block detection. For T_R values lower than 1.6 results are less accurate, and this is due to a larger number of interpolated points, and a lower number of determined ball positions.

Results on the influence of T_R values for algorithms based on frame differencing and median filtering were similar.

Results from Table 9 show that the highest block detection percentage was for T_b value equal to 1.5.

Table 10
Time of analysis for algorithms without light intensity changes model.

	Frame Differencing	Median Filter	GMM + Template Matching
τ_1 [ms]	11.6	15.8	18.2
τ_2 [ms]	37.8	42.3	32.1
τ_3 [ms]	31.4	36.4	29.1

τ_1 – time of analysis of one frame at training stage (background model creation), τ_2 – time of analysis of one frame at ball tracking stage, τ_3 – average time of one frame analysis.

Table 11
Time of analysis for algorithms with light intensity changes model.

	Frame Differencing	Median Filter	GMM + Template Matching
τ_1 [ms]	28.8	33.9	35.6
τ_2 [ms]	19.9	26.1	22.4
τ_3 [ms]	22.1	28.1	25.4

τ_1 – time of analysis of one frame at training stage (background model creation), τ_2 – time of analysis of one frame at ball tracking stage, τ_3 – average time of one frame analysis.

Table 12
Time of analysis of one frame at ball tracking stage for algorithms with light intensity changes model and without using region of interest.

	Frame Differencing	Median Filter	GMM + Template Matching
No Region of Interest	40.7 ms	43.0 ms	44.1 ms
With Region of Interest	19.9 ms	26.1 ms	22.4 ms

5.5. Time of analysis

In this section the authors presented the time of analysis for each method. In Tables 10 and 11, time of analysis is shown respectively with and without the light intensity changes model. These results are taken from tests on 2500 frames (synthetic and real), run on a computer equipped with an Intel Core i7-5820 K 3.30 GHz and 64 GB of RAM.

Time of analysis was similar for all methods, with fastest being the frame differencing method, and the algorithm based on the median filtering method the slowest. Ordinarily, this method is faster than GMM background estimation algorithm method, however for our algorithm a greater number of ball positions were detected. This resulted in the frequent use of region of interest from predicted positions. In those cases, the number of pixels to analyse falls from 1920×1080 to about 70×70 for frame. To show how region of interest reduces the time of analysis times of ball tracking without position prediction are presented in Table 12.

In our method, the time of analysis for one frame using the light intensity changes model and background creation is longer than in the ball tracking stage. That being so, the training stage uses only

first 20 frames, so therefore this does not significantly affect the time of whole sequence analysis.

6. Conclusions

In this paper, we presented an algorithm for accurate ball tracking using the additional light changes' model which is based on the range of HSV values of every pixel of the image create from first frames. The proposed method was tested using both real and synthetic data. Results show that in sport halls with fluorescent lamps, algorithms without our light intensity changes model cannot accurately determine enough ball positions to approximate actual ball flight. This model significantly reduces the influence of light intensity changes and allows for accurate determination of most ball positions directly from frame images.

Comparisons on synthetic data have shown that our algorithm produces an average of ball position accuracy of approximately 0.57 pixels for all sequences. This accuracy is high, particularly taking into account effect of noise present in all sequences. Such accuracy should allow for precise support of referees in volleyball games.

Tests on real data show that block detection using only 2D images is possible based on few trajectory changes. Our algorithm correctly determined block touch presence or absence for 11 out of 12 sequences. High accuracy of determined ball positions is required to detect block touches, and our algorithm produced the highest efficacy and accuracy for block touch determination of all considered methods

The algorithm proposed in this paper has high percentage of accurate ball position determination. A combination of a Gaussian mixture model, the additional light changes model, and template matching allows for accurate determination of ball positions on 90% of images. Other positions are mostly interpolated. For all tests, our algorithm lost only 2 positions on start of the ball trajectory. Other algorithms do not produce such precise ball trajectory approximations.

Additionally, time of analysis was reduced by using region of interest on images based on predicted positions. Ball flight analyses for short volleyball videos takes acceptable amounts of time to support referees, even when using high-speed cameras.

In the future work, 3D coordinate calculations will be implemented to further increase accuracy. Trajectory in 3D will allow for the analysis of the coordinates of ball impact. This added capability would provide full support for referees in volleyball games.

References

[1] X. Tong, T. Wang, W. Li, Y. Zhang, A novel algorithm for effective ball tracking, Intern. J. Pattern Recognit. Artif. Intell. 24 (03) (2010) 359–379.
[2] X. Yu, H.W. Leong, C. Xu, Q. Tian, Trajectory-based ball detection and tracking in Broadcast Soccer Video, IEEE Ttans. Multimedia 8 (6) (2006) 1164–1178.
[3] X. Yu, C. Xu, Q. Tian, K.W. Wan, A novel ball detection framework for real soccer video, IEEE Int. Con. Multimedia. 2 (2003) 265–268.
[4] R.E. Kalman, A new approach to linear filtering and prediction problems, J. Basic Eng.-Ttans. Asme 80 (1960) 35–45.
[5] Y. Seo, S. Choi, H. Kim, K.S. Hong, Where are the ball and players? Soccer game analysis with color-based tracking and image mosaic, Lect. Notes Comput. Sci. 1311 (1997) 196–203.
[6] Y. Ji, J. Zhang, Z. Shi, M.H. Liu, J. Ren, Research on real time tracking of table tennis ball based on machine learning with low-speed camera, Syst. Sci. Control. Eng. 6 (1) (2018) 71–79, http://dx.doi.org/10.1080/21642583.2018.1450167.
[7] D. Speck, P. Barros, C. Weber, S. Wermter, Ball localization for robocup soccer using convolutional neural networks, Lect. Notes Artif. Int. 9776 (2017) 19–30, http://dx.doi.org/10.1007/978-3-319-68792-6_2.
[8] D. Liang, Y. Liu, Q. Huang, W. Gao, A scheme for ball detection and tracking in broadcast soccer video", Lect. Notes Comput. Sci. 3767 (2005) 864–875.
[9] T. D'Orazio, N. Ancona, G. Cicirelli, M. Nitti, A ball detection algorithm for real soccer image sequences, Proc. IAPR Int. Conf. Pattern Recogn. 1 (2002) 210–213.
[10] H. Chen, W. Tsai, S. Lee, J. Yu, Ball tracking and 3D trajectory approximation with applications to tactics analysis from single- camera volleyball sequences, Multimed. Tools Appl. 60 (3) (2012) 641–667.
[11] H. Chen, M. Tien, Y. Chen, W. Tsai, S. Lee, Physics-based ball tracking and 3D trajectory reconstruction with applications to shooting location estimation in basketball video, J. Vis. Commun. Image R 20 (3) (2009) 204–216.
[12] H. Chen, H. Chen, M. Hsiao, W. Tsai, S. Lee, A trajectory-based ball tracking framework with visual enrichment for broadcast baseball videos, J. Inf. Sci. Eng. 24 (1) (2008) 143–157.
[13] B. Chakraborty, S. Meher, A real time trajectory based ball detection and tracking framework for basketball video, J. Opt.-UK 42 (2) (2013) 156–170.
[14] M. Labayen, I. Olaizola, N. Aginako, J. Florez, Accurate ball trajectory tracking and 3D visualization for computer-assisted sports broadcast, Multimed. Tools Appl. 73 (3) (2014) 1819–1842.
[15] B. Sugandi, K. Hyoungseop, J.K. Tan, S. Ishikawa, Tracking of moving objects by using a Resolution image, in: Second International Conference on Innovative Computing, Information and Control, 2007, 408.
[16] Y. Ohno, J. Miura, Y. Shirai, Tracking players and estimation of the 3D position of a ball in soccer games, Proc. IAPR Int. Conf. Pattern Recogn. 1 (2000) 145–148.
[17] G. Gomez, P.H. Lopez, D. Link, B. Eskofier, Tracking of ball and players in beach volleyball videos, PLoS One 9 (11) (2014).
[18] Z. Zivkovic, F. van der Heijden, Efficient adaptive density estimation per image pixel for the task of background subtraction, Pattern Recogn. Lett. 27 (7) (2006) 773–780.
[19] B.D. Ekinci, M. Gokmen, A ball tracking system for offline tennis videos, in: 1st WSEAS International Conference on Visualization, Imaging and Simulation, 2008, pp. 45–48.
[20] Y. Ariki, T. Takiguchi, K. Yano, Digital camera work for soccer video production with event recognition and accurate ball tracking by switching search method, in: 2008 IEEE Int Con Multi, 2008, pp. 889–892.
[21] X. Cheng, X. Zhuang, Y. Wang, M. Honda, T. Ikenaga, Particle filter with ball size adaptive tracking window and ball feature likelihood model for ball's 3D position tracking in volleyball analysis, Lect. Notes Comput. SC –PCM 9314 (2015) 203–211.
[22] P.R. Kamble, A.G. Keskar, K.M. Bhurchandi, Ball tracking in sports: a survey, Artif. Intell. Rev. (2017) 1–51, http://dx.doi.org/10.1007/s10462-017-9582-2.
[23] X. Wang, V. Ablavsky, H. Shitrit, P. Fua, Take your eyes off the ball: improving ball tracking by focusing on team play, Comput. Vis. Image Underst. 119 (2014) 102–115.
[24] N.M. Oliver, B. Rosario, A.P. Pentland, A Bayesian computer vision system for modelling human interactions, IEEE Trans. Pattern Anal. Mach. Intell. 22 (8) (2000) 831–843.
[25] A. Maksai, X. Wang, P. Fua, What players do with the ball: a physically constrained interaction modeling, Proc CVPR IEEE (2016) 972–981.
[26] M. Hearst, S. Dumais, E. Osuna, et al., Support vector machines, IEEE Intell. Syst. Appl. 13 (4) (1998) 18–28.
[27] "FIVB Official Volleyball Rules 2017-2020" https://www.fivb.org/EN/Refereeing-Rules/documents/FIVB-Volleyball_Rules_2017-2020-EN-v06.pdf. (Accessed 06 June 2018).