Player Tracking in Sports Videos

Burić, Matija; Ivašić-Kos, Marina; Pobar, Miran

Source / Izvornik: **2019 IEEE International Conference on Cloud Computing Technology and Science (CloudCom), 2020, 334 - 340**

Conference paper / Rad u zborniku

Publication status / Verzija rada: **Accepted version / Završna verzija rukopisa prihvaćena za objavljivanje (postprint)**

https://doi.org/10.1109/CloudCom.2019.00058

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:195:183966

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2025-01-27**

Sveučilište u Rijeci
Fakultet informatike i digitalnih tehnologija

Repository / Repozitorij:

Repository of the University of Rijeka, Faculty of Informatics and Digital Technologies - INFORI Repository

DIGITALNA KNJIŽNICA

DIGITALNI AKADEMSKI ARHIVI I REPOZITORIJI

Player Tracking in Sports Videos

Matija Burić
SIT PS Rijeka
Hep d.d.
Rijeka, Croatia
ORCID: 0000-0003-3528-7550
matija.buric@hep.hr

Marina Ivašić-Kos
Department of Informatics
University of Rijeka
Rijeka, Croatia
ORCID: 0000-0002-1940-5089
marinai@uniri.hr

Miran Pobar
Department of Informatics
University of Rijeka
Rijeka, Croatia
ORCID: 0000-0001-5604-2128
mpobar@uniri.hr

This paper considers the problem of tracking the players in handball videos using a single video source. Tracking of handball players in the video is a difficult task as they can frequently leave and re-enter the camera field of view, often change directions quickly and occlude each other. Players wear similar team uniforms, play indoor under artificial illumination, with the background than can vary greatly as the handball court is often painted in various colors. The continually improving accuracy of CNN-based object detectors makes tracking-by-detection methods an attractive choice for tracking players in sports videos as they can perform online and with low computational requirements on top of object detection. Here we consider the use of three tracking-by-detection methods in conjunction with the YOLO object detector, namely the standard Hungarian assignment algorithm, the Simple Online and Real-time Tracking (SORT) algorithm that adds a motion model, and its extension Deep SORT. The methods are tested on a custom dataset of handball video scenes.

Keywords— Object Detection, Yolo, Deep SORT Tracking, Action Recognition, Sports, Hungarian, computer vision, object tracking

I. INTRODUCTION

Video footages conceal more information than still images about how objects and surroundings change over time. This requires more space for storage and higher computational power than simple object detection but could prove to be essential in certain cases where action recognition is desired [1]. With a wide spread of applications such as monitoring of traffic flow [2], robotics vision [3], controlling autonomous vehicles [4][5], medical diagnosis [6] and action recognition [7] object tracking become the subject of many computer vision researches.

In the case of action recognition in sport videos, some activities can be better described with information about the trajectory of the certain object. Objects are tracked in order to perform object extraction, object recognition and tracking or provide decision about activity performed.

There are many factors, which influence how successful tracking is, and what parameters influence robustness of the build tracker such as: variation of illumination, out-of-plane and in-plane rotation, sudden change of object trajectory, out-of-view object, deformation and motion blur. All mentioned problems should be usually addressed simultaneously, which rises the level of complexity giving a rough picture of challenges tracker has to be able to handle.

Some of mentioned challenges are still too demanding for a computer vision and some like illumination and blur are being handled by everyday improving hardware. Multiple camera sources from a different point of view could help overcome occlusion, scale and even earlier mentioned issues like out-of-view and out-of-plane rotation but the focus of this research is on finding solution using only one camera. The problems, which authors have partly explored, are mostly related to scale of objects being tracked and occlusion of the same. The depth dimension that can make the object too small to detect or too large which than falls into domain of occlusion, covering other interesting objects and making them difficult to track influence object size, Fig. 1.

Figure 1. Tracking situation example with frequent occlusion

To begin tracking process one should first determine the boundaries of desired object. This can be done manually but in correspondence to recent researches of object detection, it is more convenient to perform selection automatically. Numerous machine learning algorithms were already developed with a motivation to extract important information in order to detect certain object classes, e.g. faces [8] or pedestrians [9]. In fact, authors of [10] proven that feature extraction has a significant impact on tracking, where for example trackers based on Histogram of oriented gradients (HOG) features performs better than those based on Haar-like features. Lately, deep learning approaches founded on convolutional neural networks (CNN) have been more successful in feature extraction [11] applied in object detection without need for specific class-dependent hand coded features [7]. It is reasonable to believe that with continuous progress of neural networks these results will be even better.

According to [11] visual tracking method consists of two segments: a motion model and observation model. A motion model makes a prediction of a state in which object will be based on description of the previous state. Kalman [12] and particle [13], [14] filters are examples of such a model. Observation model represents the info about appearance of the tracked object and verifies predictions in each frame [15]. Based on research in [10] observation model has more significance in visual tracking than motion model. Currently used observation models can be generally categorized as generative or discriminative methods [11]. Generative methods are trying to search for most similar regions to a desired object – template matching [12][16][17][18] while discriminative methods are more focused on classification trying to extract object from the background [19][20]. Which method should one choose depends on the purpose of the object tracker, for example, generative CNN based trackers

This research has been supported by the Croatian Science Foundation under the project IP-2016-06-8345 "Automatic recognition of actions and activities in multimedia content from the sports domain" (RAASS) and under project 7619 "A Knowledge-based Approach to Crowd Analysis in Video Surveillance" (KACAVIS).

XXX-X-XXXX-XXXX-X/XX/$XX.00 ©20XX IEEE

are usually faster but less accurate than discriminative CNN trackers [11]. Not only CNNs are used for deep visual tracking. Recurrent neural networks (RNNs) are quite suitable for sequence modelling due to their ability to store memory of previous states and to establish temporal connection between them. Intrigued by success in field of handwriting [21][22] and speech recognition [23][24] some authors propose combination of RNNs with correlation filters [25] or even adding features produced by RNN into CNN to achieve a robust feature representation [26].

Deep neural networks can be used to not only extract features but also in making a decision about the best candidate of the tracked object. Networks in this case can be divided into feature extraction networks (FENs) and end-to-end networks (EENs). FENs provide benefit of acquiring quality features, which are then used by traditional methods to learn model appearance where EENs handle both tasks. EENs can as a result give a probability map or, as a complete solution, provide object location in a form of bounding box or segmentation representation [27][28].

Due to nature of the sport, scenes where quick and fast detection is needed with constant multiple object movement in different directions authors have decided to focus on solution, which will provide fast solutions with optimal precision. Therefore, Yolo [29] was used to handle object detection and Simple online and real-time tracking (SORT) [30], Deep SORT [31] and Hungarian algorithm-based approach [32] as object trackers. All tracking methods along with Yolo detector will be briefly described and tested within experiment.

The paper is structured as follows. In the next chapter, a short description of Yolo method will be presented following the introduction into tracking methods used. Next section is reserved for the experiment on the handball sports data gathered by authors during practice and competition in order to make preliminary assumption for further implementation in the project of handball sport action recognition. After experiment analysis, short conclusion with most interesting observations will be presented.

II. DETECTION

A. Yolo Object Detector

The basic idea and difference to other object detectors, such as, Faster R-CNN [33], Mask R-CNN [34], SSD [35], etc. of "You Only Look Once" (YOLO) [36] method is that it uses a single-stage network architecture to predicts the class probabilities and corresponding bounding boxes in a single pass. The main benefit of this approach is its real-time detection capability [37] [36]. Since its first release there were few releases bringing new improvements until current version YoloV3 [38] published in 2018.

Under scope of this research YoloV2 [29] was used in order to keep the reference with prior research [37][39][40] [43] and comparison under project of action recognition in handball sport.

Original Yolo model network architecture has 24 convolutional layers and additional two fully connected layers. The YoloV2 network differs from its predecessor where certain number of convolution layers have been removed, including fully connected layers, and some were replaced with max-pooling layers. By doing so speed and accuracy has been optimized [29]. Additional techniques, which improved Yolo, include usage of Anchor-Boxes and Batch-Normalization of hidden layers. Concept of Anchor-Boxes helps determining shape and layout of certain object. If one takes, for example, shape of the box surrounding regular pedestrian and shape of the normal car by usage of Anchor-Boxes one could classify object better considering pedestrians are mostly in standup position with height-width ratio in favor to height, opposite to car box shape.

The base network used by YoloV2 trained on The PASCAL Visual Object Classes (VOC) dataset [41] takes as input images size 416x416 px with depth value 3 for each color of RGB and outputs 3-dimensional array with a shape 13x13xD, where D depends on number of classes and Anchor-Boxes. According to original paper [29] 5 Anchor-Boxes are the trade of between speed and accuracy giving optimal result. Under the experiment in this paper, in order to achieve better accuracy network has been changed to input frames sizes 1024x1024 ending up in 32x32x30 array based on single class and five Anchor-Boxes as shown in Table 1.

TABLE I. CNN USED IN EXPERIMENT WITH ADJUSTED PARAMETERS FOR BETTER DETECTION OF SMALLER/DISTANT OBJECTS

Customized YoloV2 network			
no.	***Layer***	***Parameters***	***Output***
1	Convolution	size=3, stride=1, pad=1, filters=32, bn=1, act=Leaky ReLU	1024 x 1024 x 32
2	Maxpooling	Size=3, stride=2	512 x 512 x 32
3	Convolution	size=3, stride=1, pad=1, filters=64, bn=1, act=Leaky ReLU	512 x 512 x 64
4	Maxpooling	Size=2, stride=2	256 x 256 x 64
5	Convolution	size=3, stride=1, pad=1, filters=128, bn=1, act=Leaky ReLU	256 x 256 x 128
6	Convolution	size=1, stride=1, pad=1, filters=64, bn=1, act=Leaky ReLU	256 x 256 x 64
7	Convolution	size=3, stride=1, pad=1, filters=128, bn=1, act=Leaky ReLU	256 x 256 x 128
8	Maxpooling	size=2, stride=2	128 x 128 x 128
9	Convolution	size=3, stride=1, pad=1, filters=256, bn=1, act=Leaky ReLU	128 x 128 x 256
10	Convolution	size=1, stride=1, pad=1, filters=128, bn=1, act=Leaky ReLU	128 x 128 x 128
11	Convolution	size=3, stride=1, pad=1, filters=256, bn=1, act=Leaky ReLU	128 x 128 x 256
12	Maxpooling	size=2, stride=2	64 x 64 x 256
13	Convolution	size=3, stride=1, pad=1, filters=512, bn=1, act=Leaky ReLU	64 x 64 x 512
14	Convolution	size=1, stride=1, pad=1, filters=256, bn=1, act=Leaky ReLU	64 x 64 x 256
15	Convolution	size=3, stride=1, pad=1, filters=512, bn=1, act=Leaky ReLU	64 x 64 x 512
16	Convolution	size=1, stride=1, pad=1, filters=256, bn=1, act=Leaky ReLU	64 x 64 x 256
17	Convolution	size=3, stride=1, pad=1, filters=512, bn=1, act=Leaky ReLU	64 x 64 x 512
18	Maxpooling	size=2, stride=2	32 x 32 x 512
19	Convolution	size=3, stride=1, pad=1, filters=1024, bn=1, act=Leaky ReLU	32 x 32 x 1024
20	Convolution	size=1, stride=1, pad=1, filters=512, bn=1, act=Leaky ReLU	32 x 32 x 512
21	Convolution	size=3, stride=1, pad=1, filters=1024, bn=1, act=Leaky ReLU	32 x 32 x 1024
22	Convolution	size=1, stride=1, pad=1, filters=512, bn=1, act=Leaky ReLU	32 x 32 x 512
23	Convolution	size=3, stride=1, pad=1, filters=1024, bn=1, act=Leaky ReLU	32 x 32 x 1024
24	Convolution	size=3, stride=1, pad=1, filters=1024, bn=1, act=Leaky ReLU	32 x 32 x 1024
25	Convolution	size=3, stride=1, pad=1, filters=1024, bn=1, act=Leaky ReLU	32 x 32 x 1024
26	Route 16 layer		64 x 64 x 512
27	Convolution	size=1, stride=1, pad=1, filters=64, bn=1, act=Leaky ReLU	64 x 64 x 64
28	Flattening and Reorganization [27,24]		32 x 32 x 1280
29	Convolution	size=3, stride=1, pad=1, filters=1024, bn=1, act=Leaky ReLU	32 x 32 x 1024
30	Convolution	size=1, stride=1, pad=1, filters=30, bn=0, act=Linear,	32 x 32 x 30

YoloV2 network consists from 23 convolution layers and 4 Maxpooling layers. All Convolution layers has Leaky ReLU activation function and batch normalization except last one where linear activation function is used. The convolution layers are the building blocks of any CNN and their primary function is to extract features from the input images. Maxpooling layers are used to down-sample an input representation in order to reduce dimensionality keeping information relative to object detection. Route layer simply takes as input result of the certain layer, without any processing, bringing finer grained features in, from earlier in the network. Reorganization layer reshapes feature map - decreases size and increases number of channels, without changing elements. In this case, it concatenate layers 27 and 24.

At first stage, after the input image is sliced into cells, in this case its 32x32 grid, for each grid cell certain number of bounding boxes, related to AnchorBox value, is designated. During the training, bounding box prediction has no information about the object so the confidence value is set to zero. Once it holds an object and detection occur, the

confidence is calculated with a use of the intersection-over-union (IOU) score of prediction and ground truth (GT) boxes.

If the IOU score is high enough it is presumed that certain box contains an object and the grid cell in the center of the object is picked as container of prediction of a particular object. To handle situations where there are many bounding boxes with high probability Non-Max Suppression can be used.

III. TRACKING

The problem of player tracking is an instance of Multi-Object Tracking (MOT) problem which entails estimating from observations the state of the objects – their position, configuration and identity over time. Ideally, the tracker should in every video frame detect all the present players, correctly determine their position and assign a unique ID for each player that stays the same throughout the video sequence.

With recent improvements in performance of object detectors, tracking-by-detection has become a leading paradigm for MOT due to the ability to deal with challenges such as cluttered scenes or dynamics of tracked objects. In the tracking-by-detection paradigm, the tracking algorithm relies on the object detector to detect objects on the scene in each frame and to determine their location, while the role of the tracking algorithm is reduced to the problem of associating the detections across frames that belong to the same object.

Here we consider three methods that perform this association in online fashion i.e. that can assign track IDs for each frame as it becomes available, considering only detections in current and last frame, as opposed to methods that have to consider the whole video at once. These are the Munkres' version of the Hungarian algorithm, the Simple Online and Realtime Tracking [30] method (SORT) and the SORT with a deep association metric [31] (Deep SORT) method. To inform the association process, the trackers use different models of motion or appearance of objects in the video. The SORT method uses the Kalman filter [12] to model the linear motion on top of the base Hungarian assignment algorithm [32], while Deep SORT adds an appearance model based on deep neural network features.

A. Hungarian algorithm-based approach

In the case of tracking using the Hungarian algorithm, only the parameters of the bounding boxes detected by the YOLO object detector are used for assigning the detected objects to tracks, and no visual features extracted from the video frames are used after detection. The algorithm finds the globally optimal assignment with respect to the cost function whose value depends on the Euclidean distance of each detected object's bounding box centroid from the predicted centroid of an object in the considered track and on the size difference of the bounding box and the last assigned bounding box to the same track.

The assignment cost $\mathrm{d}(b,k)$ of assigning a bounding box b with the centroid C_b and area P_b to the k-th track with the predicted centroid ${C_{k+1}}'$ is:

$$\mathrm{d}(b,k) = w \cdot d_2(C_b, {C_{k+1}}') + (1-w)|P_b - P_{k-1}| \quad (1)$$

$$w \in [0,1], k \in N$$

where P_{k-1} is the area of the last bounding box assigned to the track with the ID k.

For the prediction of the centroid location ${C_{k+1}}'$, last known position of object centroid in the track k is used, so ${C_{k+1}}' = C_{k-1}$.

The initial assignment of bounding boxes to tracks simply assigns a unique track ID to each detected bounding box whose detector confidence is higher than a set threshold, which was set to 0.5.

New tracks are created whenever the number of detected objects exceeds the number of currently active tracks, and the unassigned object's bounding box is used to initialize the new track. An existing track is considered inactive when no detections are assigned to that track for a number of frames, T_inactive, here set to 20 frames. After a track is noted to be inactive, no further detections are added to it. If an object reappears in frame or is detected again after T_inactive frames, it will get a new track ID and will be considered a new object. The full frame rate of the source video is used, and players are tracked rather well even when move fast and switch the positions.

The use of a Kalman filter to predict box location was considered in the preliminary experiment but showed uneven performance and was not used in the further experiment.

B. SORT Tracker

Like the described Hungarian assignment approach, the SORT method associates the detected objects across video footage solely based on the bounding box parameters of the detection results. The Hungarian algorithm is again used for the assignment of bounding boxes to tracks, while a Kalman filter is now used to model the motion of the objects.

The assignment cost $d_J(b,k)$ of assigning the detected object's bounding box b to the k-th track with the predicted bounding box b_k is defined as the Jaccard distance of boxes b and b_k, so that the boxes with the highest intersection-over-union (IOU) values have the smallest cost:

$$d_J(b,k) = 1 - \frac{|b \cap b_k|}{|b \cup b_k|} \quad (2)$$

A minimum IOU threshold is used to prevent assignments to tracks when the overlap between the predicted and detected box is smaller than a predefined value, in this case set to 0.3.

To predict the bounding box position and size in the current frame, a linear constant velocity model is used for each tracked object. The state of each tracked object is modeled using a vector whose components are the bounding box centroid coordinates, the bounding box area, its aspect ratio, and the velocities of the centroid coordinates and area. The aspect ratio is considered constant. After assigning a bounding box to a track, the assigned box is used to update the state vector, where the velocity components are computed using the Kalman filter. When no detection is assigned to a track, the state is updated simply by computing the predicted position of bounding box using the velocity components.

As in the previous case, new tracks are created whenever more objects are detected in a frame than there are active tracks. A new track is also created if the IOU of the detected object is smaller than minimum IOU threshold for all active tracks. The new track is initialized with the detected bounding box coordinates and with velocities set to zero and with velocity covariance with large number to signify low

confidence in that measurement. In addition, tracks are considered inactive after T_inactive frames, so objects that reappear after T_inactive frames are tracked with a new ID. As above, T_inactive is set to 20 frames. This behavior causes objects that are under occlusion longer and thus not detected to be constantly reevaluated as new objects.

C. Deep SORT

Deep SORT specifically aims to improve tracking of objects under occlusion in comparison to the SORT method by using appearance information to re-identify objects that appear in the scene after they have not been tracked for some frames. In particular, the assignment cost is augmented with an appearance distance measure based on feature descriptors that are extracted with a convolutional neural network (CNN). The cost of assigning the j-th detection to the i-th track ($c_{i,j}$) is given by:

$$c_{i,j} = \lambda d^{(1)}(i,j) + (1-\lambda) d^{(2)}(i,j) \qquad (3)$$

The $d^{(1)}$ measure is the squared Mahalanobis distance between the predicted Kalman state of the object in track i and the detected bounding box j:

$$d^{(1)}(i,j) = \left(d_j - y_i\right)^T S_i^{-1} \left(d_j - y_i\right) \qquad (4)$$

where y_i and S_i are the mean and covariance matrix of i-th track's bounding box observations, and d_j is the j-th bounding box detection.

The $d^{(2)}$ measure is the visual appearance metric calculated as the cosine distance between the j-th detection and i-th track:

$$d^{(2)}(i,j) = \min\left\{1 - r_j^T r_k^{(i)} \mid r_k^{(i)} \in \mathcal{R}_i\right\} \qquad (5)$$

where r_j is the appearance descriptor with $|r_j| = 1$ extracted from the part of the image within j-th bounding box, and $\mathcal{R}_i$ is the set of last 100 appearance descriptors r_k^i associated with the track i.

To calculate the appearance descriptors, a convolutional neural network is used that was pre-trained on a person re-identification dataset [31] that contains over a million images of 1261 pedestrians. The CNN is a wide residual network with two convolutional layers followed by six residual blocks that calculates a 128-element vector in the dense layer. The features are normalized to fit within the unit hypersphere to be compatible with the cosine distance.

The influence of $d^{(1)}$ and $d^{(2)}$ is controlled by the parameter λ, which is in practice often set to 0 [31], so that only the visual appearance metric is used for association. The $d^{(1)}$ is however still used to filter out detections that are too far away from the predicted locations, so a detection whose distance $d^{(1)}$ to *i*-th track is greater than a set threshold cannot be assigned to that track regardless of visual similarity. Similarly, a visual similarity threshold, determined experimentally[31], is used to prevent assignment of a detection to a track when the $d^{(2)}$ measure is too high.

IV. Experiment and Analysis

A. Experiment description

An experiment is performed to demonstrate how well the SORT and Deep SORT tracking-by-detection methods, as well as the basic Hungarian assignment method, can handle tracking of players during practice and gameplay of typical indoor team sport. All three methods were tested with the identical object detector on a custom handball dataset.

The custom dataset consists of handball sports video footages taken at 1920x1080 resolution and 30 frames per second. To collect footages 3 GoPro cameras were used, mounted at the 1.5m height and positioned at the edge of the field at the different relative positions. Scenes are occupied by youngsters with their trainers, occasionally using more than one ball and dressed casually. Indoors artificial lighting was used with some sunlight thought windows while outdoor scenes were illuminated with a bright daylight during midday with clear or with almost no clouds. This dataset was also used to train and test player and ball detectors in [37][42][43] and YoloV2 has been shown to achieve the best results. For this reason, Yolo V2 was also used in this study, but we focused only on tracking players so all other objects detection except person are discarded.

Implementation was made using Standard Azure virtual machine D8s v3, under Ubuntu 16.04 LTS Linux environment with a use of Tensorflow and Python programing. At this point of the research speed tests have lower priority than functionality so therefore speed test results are not taken into consideration.

B. Analysis

First tests were made using pretrained tiny-yolo weights without any alteration in order to get the picture how method works out-of-the-box (see Fig. 2.).

Tracking shown in Fig. 2 is unstable and inconsistent. The numbers above the bounding boxes represent the tracking session of each object. Due to constantly changing sizes or even complete absence of detected objects in the series of frames, trackers fail to follow movement of players. Also noticed are false positives (FP), bounding boxes (BB) 65 and 23, multiple tracking of same object, BB 35 and 9, and broken tracking sessions, BB pairs (40,57), (1,41), (42,60) and (31,58).

In the next test, full yolo weights were used with confidence threshold set to 0.5.

The Fig. 3 shows an example of a tracking situation when occlusions occurred – the player marked 240 in the bottom row moves behind and is for some time occluded by the person marked 238, while the player marked 223 in the bottom row for some time occludes the player 242 in the bottom row. The Hungarian algorithm (top row) successfully tracked one of

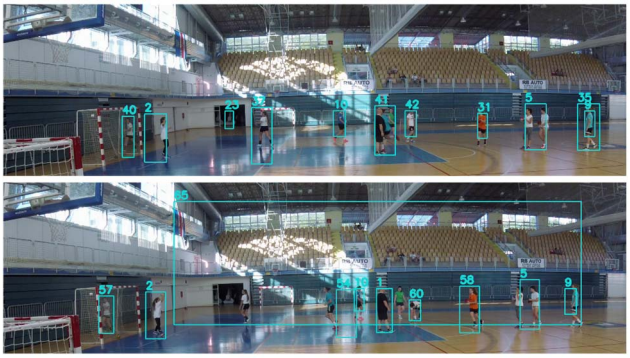

Figure 2. Influence of object detection on tracking

Figure 3. An example of tracking situation with occlusion. Top: Hungarian algorithm, middle: SORT, bottom: Deep SORT. The left and right frames are 1 second apart

four persons (the person marked with 802), two of the players got new IDs after occlusion, while one player has switched ID with another (807 got previously existing ID 812). The SORT algorithm (middle row) successfully tracked three persons (marked 219, 217 and 221), while one player incorrectly got a new tracking ID after occlusion (222 became 224). Deep SORT (bottom row) managed to track correctly all four players.

The last example (see Fig. 3.) shows the tracking result with Deep SORT on a sequence of six frames taken outside while most players were changing position and some changing the pose. The tracking is more stable, keeping track numbers for most players constant. Interesting to notice in this example is that player BB 2 have overtaken tracking session of BB 8 and had no problem keeping it after closest player BB 1 have completely blocked him from sight. In addition, players far back were detected most of the time with similar tracking score.

V. Conclusion and Discussion

The experiment demonstrates that the recent development of deep learning-based object detectors made tracking-by-detection approaches more attractive for the problem of tracking players in team sports. The tracker that additionally uses visual features to re-identify players after occlusions in particular shows promising results. The tracking is improved if more successful object detector is used. In this experiment, the custom dataset of handball footages was used where players did not wear clothing with numeration or team colors, so detection and tracking were more challenging. All the tested methods had problems with tracking players temporarily out-of-sight, therefore player re-identification is still an open issue. With object detectors specifically designed for individual players, tracking could possibly be improved but this requires unique weights for each player. Additional challenge, which should be addressed in future, is finding a mechanism how to preserve tracking in cases when trajectory of the player is rapidly changing. This is shown to be difficult for all tested trackers. It is interesting to notice that bounding boxes of tracked players easily adopt to the pose of each player but this shows to be closely related to the efficiency of object detector used. Further research should also include tracking of ball object, which is more difficult due to its size, occlusions and speed. Efficient ball tracker would be helpful in determining of active player and action recognition.

REFERENCES

[1] Y. Wang, J. F. Doherty, and R. E. Van Dyck, "Moving object tracking in video," in Proceedings - Applied Imagery Pattern Recognition Workshop, 2000, vol. 2000-January, pp. 95–101.

[2] B. Tian, Q. Yao, Y. Gu, K. Wang, and Y. Li, "Video processing techniques for traffic flow monitoring: A survey," in IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC, 2011, pp. 1103–1108.

[3] J. M. B. Oñate, D. J. M. Chipantasi, and N. D. R. V. Erazo, "Tracking objects using Artificial Neural Networks and wireless connection for robotics," J. Telecommun. Electron. Comput. Eng., vol. 9, no. 1–3, pp. 161–164, 2017.

[4] M. Brown, J. Funke, S. Erlien, and J. C. Gerdes, "Safe driving envelopes for path tracking in autonomous vehicles," Control Eng. Pract., vol. 61, pp. 307–316, 2017.

[5] V. A. Laurense, J. Y. Goh, and J. C. Gerdes, "Path-tracking for autonomous vehicles at the limit of friction," in Proceedings of the American Control Conference, 2017, pp. 5586–5591.

[6] S. Walker et al., "Systems and methods for localizing, tracking and/or controlling medical instruments," 2017.

[7] M. Buric, M. Pobar, and M. Ivasic-Kos, "An overview of action recognition in videos," 2017 40th Int. Conv. Inf. Commun. Technol. Electron. Microelectron. MIPRO 2017 - Proc., pp. 1098–1103, 2017.

[8] P. Viola and M. Jones, "Managing work role performance: Challenges for twenty-first century organizations and their employees.," Rapid Object Detect. using a Boost. Cascade Simple Featur., 2001.

[9] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005, 2005.

[10] N. Wang, J. Shi, D. Yeung, J. J.-P. of the IEEE, and undefined 2015, "Understanding and diagnosing visual tracking systems," openaccess.thecvf.com.

[11] P. Li, D. Wang, L. Wang, and H. Lu, "Deep visual tracking: Review and experimental comparison," Pattern Recognit., vol. 76, pp. 323–338, 2018.

[12] A. Heidari and P. Aarabi, "Real-time object tracking on iPhone," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2011, vol. 6938 LNCS, no. PART 1, pp. 768–777.

[13] P. Pérez, C. Hue, J. Vermaak, and M. Gangnet, "Color-based probabilistic tracking," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2002, vol. 2350, pp. 661–675.

[14] Y. Li, H. Ai, T. Yamashita, S. Lao, and M. Kawade, "Tracking in low frame rate video: A cascade particle filter with discriminative observers of different life spans," IEEE Trans. Pattern Anal. Mach. Intell., vol. 30, no. 10, pp. 1728–1740, 2008.

[15] X. Li, W. Hu, C. Shen, Z. Zhang, A. Dick, and A. Van Den Hengel, "A survey of appearance models in visual object tracking," ACM Transactions on Intelligent Systems and Technology, vol. 4, no. 4. 2013.

[16] J. Kwon and K. M. Lee, "Tracking by sampling and integrating multiple trackers," IEEE Trans. Pattern Anal. Mach. Intell., vol. 36, no. 7, pp. 1428–1441, 2014.

[17] X. Mei and H. Ling, "Robust visual tracking and vehicle classification via sparse representation," IEEE Trans. Pattern Anal. Mach. Intell., vol. 33, no. 11, pp. 2259–2272, 2011.

[18] A. Adam, E. Rivlin, and I. Shimshoni, "Robust fragments-based tracking using the integral histogram," in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2006, vol. 1, pp. 798–805.

[19] C. Xu, W. Tao, Z. Meng, and Z. Feng, "Robust visual tracking via online multiple instance learning with Fisher information," Pattern Recognit., vol. 48, no. 12, pp. 3917–3926, 2015.

[20] L. Zhang and P. N. Suganthan, "Robust visual tracking via co-trained Kernelized correlation filters," Pattern Recognit., vol. 69, pp. 82–93, 2017.

[21] D. Cireşan and U. Meier, "Multi-Column Deep Neural Networks for offline handwritten Chinese character classification," in Proceedings of the International Joint Conference on Neural Networks, 2015, vol. 2015-Septe.

[22] D. C. Cireşan, U. Meier, L. M. Gambardella, and J. Schmidhuber, "Convolutional neural network committees for handwritten character classification," in Proceedings of the International Conference on Document Analysis and Recognition, ICDAR, 2011, pp. 1135–1139.

[23] Z. Wu, C. Valentini-Botinhao, O. Watts, and S. King, "Deep neural networks employing Multi-Task Learning and stacked bottleneck features for speech synthesis," in ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings, 2015, vol. 2015-Augus, pp. 4460–4464.

[24] S. Kim, T. Hori, S. W.-2017 I. International, and U. 2017, "Joint CTC-attention based end-to-end speech recognition using multi-task learning," ieeexplore.ieee.org.

[25] Z. Cui, S. Xiao, J. Feng, S. Y.-P. of the IEEE, and U. 2016, "Recurrently target-attending tracking," openaccess.thecvf.com.

[26] H. Fan and H. Ling, "SANet: Structure-Aware Network for Visual Tracking," in IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2017, vol. 2017-July, pp. 2217–2224.

[27] D. Held, S. Thrun, and S. Savarese, "Learning to track at 100 FPS with deep regression networks," in Lecture Notes in Computer Science, 2016, vol. 9905 LNCS, pp. 749–765.

[28] G. Ning et al., "Spatially supervised recurrent convolutional neural networks for visual object tracking," in Proceedings - IEEE International Symposium on Circuits and Systems, 2017.

[29] Redmon, J. and Farhadi, A. 2017. "YOLO9000: better, faster, stronger," arXiv preprint.

[30] A. Bewley, Z. Ge, L. Ott, et. al., 2016, "Simple online and realtime tracking," ieeexplore.ieee.org.

[31] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in Proceedings - International Conference on Image Processing, ICIP, 2018, vol. 2017-Septe, pp. 3645–3649.

[32] H. W. Kuhn, "http://dx.doi.org/10.1002/nav.3800020109The Hungarian method for the assignment problem," Nav. Res. Logist. Q., vol. 2, no. 1–2, pp. 83–97, Mar. 1955.

[33] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no. 6, pp. 1137–1149, 2017.

[34] K. He, G. Gkioxari, P. D.-P. of the I., and U. 2017, “Mask r-cnn,” openaccess.thecvf.com.

[35] W. Liu et al., “SSD: Single shot multibox detector,” in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2016, vol. 9905 LNCS, pp. 21–37.

[36] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2016, vol. 2016-Decem, pp. 779–788.

[37] M. Buric, M. Pobar, M. Ivasic-Kos, and A. Yolo, “Ball detection using Yolo and Mask R-CNN,” 5th Annual Conf. on Computational Science & Computational Intelligence (CSCI’18), Las Vegas, USA

[38] J. Redmon and A. Farhadi, Yolov3: An incremental improvement,” arXiv preprint arXiv:1804.02767.

[39] Pobar, M., Ivasic-Kos, M. 2018. “Detection of the leading player in handball scenes using Mask R-CNN and STIPS” in 11th International Conference on Machine Vision (ICMV 2018), Muenchen, Germany

[40] Ivasic-Kos, M., Pobar, M. 2018. “Building a labeled dataset for recognition of handball actions using mask R-CNN and STIPS”, in 7th IEEE European Workshop on Visual Information Processing (EUVIP), Tampere, Finland, pp. 1-6

[41] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The Pascal Visual Object Classes Challenge: A Retrospective,” Int. J. Comput. Vis., vol. 111, no. 1, pp. 98–136, 2014.

[42] M. Buric, M. Pobar, and M. Ivasic-Kos, “Object detection in sports videos,” in 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2018 - Proceedings, 2018, pp. 1034–1039.

[43] M. Buric, M. Pobar, and M. Ivasic-Kos, “Adapting Yolo network for ball and player detection,” in ICPRAM 2019 - Proceedings of the 8th International Conference on Pattern Recognition Applications and Methods, 2019, pp. 845–851.