Low cost player tracking in field hockey

Henrique Duarte Moura, Leonid Kholkine, Laurens Van Damme, Kevin Mets, Christiaan Leysen, Tom De Schepper, Peter Hellinckx, Steven Latré

University of Antwerpen - imec, IDLab - Department of Computer Science

Abstract In the paper, we describe the technical details of a multi-player tracker system using tracking data obtained from a single low-cost stationary camera on field hockey games. Analyzing the tracking data of the players only from the transmitted video opens a multitude of applications that allows the cost of technology to be reduced. This method does not depend on the cooperation of the players (by using sensors) or their teams (by sharing data with a third party). The approach taken in this paper uses a variety of computer vision and tracking techniques. Making player tracking data more accessible lowers the barrier to entry for sports research and increases the period during which advanced analysis methods can be applied. The proposed system runs the full pipeline at 3 fps on a computer with a simple graphics card.

Keywords: Computer vision, field hockey video analysis, player tracking

1 Introduction

Artificial Intelligence (AI) impacts many aspects of sports. Its application can provide pre- and post-game reports, online summaries, and performance analysis personalized to specific targets, such as athletes, coaches, media, and fans. Many challenges still exist involving data collection (e.g., merging sensor and video data) and processing (e.g., large volumes of high-quality video data). Tracking players is key to analyze game strategies, player and team performance. It involves two steps: identification of in-field players and tracking them in time.

AI is widely used to track players using video data. Most studies consider football [1], soccer [2, 3], basketball [4, 5], and ice hockey [6] games. Most of them require expensive hardware around the pitch, sensors attached to the players or the ball, and manual annotation. Sensors could simplify tracking. However, there are some challenges using sensors: (a) not all teams wear them, (a) there is no suitable sensor to use in the ball, in most cases, (c) its use needs approval from the sports federation, (d) the sensor may not have an open API for online access, and besides that (e) the opposing team has little interest in sharing its data.

Multiple-object tracking (MOT) solutions benefit from the high quality of state-of-art object detection algorithms, popularizing the tracking-by-detection approach. This type of approach allows very simple, but accurate tracking methods to be applied, and depends mainly on the use of bounding boxes or detection masks as input to high-speed trackers. However, tracking players remains challenging due to frequent occlusions, abrupt movements, and environmental conditions (lighting, shadows, etc.). Moreover, players in one team wear the same uniform, making them more difficult to distinguish. In addition, players' body shape variations and poses, motion blur, and the presence of the reserve players and spectators alongside the pitch make the players hard to be tracked reliably. Thus, many existing tracking methods fail in this field.

We show in this paper a player detection and tracking method inspired by recent progress in deep learning (DL). Our method operates on a sequence of frames from a stationary camera. We develop the full field hockey video analysis pipeline. The different modules are designed with performance in mind to allow efficient processing of high-definition video. In this paper, we make three contributions. First, we present a complete pipeline that can (i) identify which area of the image corresponds to the field; (ii) detect the people in the image and select only the players on the pitch; (iii) exclude goalkeepers and referees because they are not tracked; (iv) identify which players correspond to which team; and finally (v) track the movement of these players on the pitch. Second, a comparison of two different approaches for each module described before. Third, a technique that combines the tracking by similarity with a merging stage. In contrast to existing approaches that solely focus on one aspect of the pipeline, our proposal yield insights into which points need improvement in the current state-of-the-art algorithms.

The paper is organized as follows. Related work is listed in Section 2, while the architecture is presented in Section 3. The experimental setup and results are discussed in, respectively, Sections 4 & 5. Finally, Section 6 concludes this work.

2 Related work

Several works on player detection and tracking make use of computer vision algorithms, tracking heuristics, and DL. Here, we discuss the main approaches.

Detection methods: Originally, feature maps associated with classifiers are used to identify the players, such as the Deformable Part Model [7, 8] and the histogram of oriented gradients (HOG) features [9]. Gaussian mixture model (GMM) can separate background and foreground [10]. Those methods work well for offline processing but depend on pre-selected features to identify the objects. Newer methods profit from the advances in DL using detection networks to identify the player or the ball [11–14]. Other solutions identify the pitch using the dominant color of the background, morphological filtering [15, 16], or more complex color segmentation approaches [17] but suffer from shadows and lighting variation. One can also apply a deep semantic segmentation network using semantic cues (the field surface, lines, etc.) [18]. However, the hockey pitch has few semantic cues.

Trackers: Most modern trackers follow the tracking-by-detection paradigm, where an object detector first finds all objects of interest in each individual frame before a tracking algorithm deals with the problem of associating the objects in consecutive frames [11, 19–26]. A common approach is to use a Kalman filter (KF) to forecast random variables (e.g. position or speed) at a specific timestamp [8]. Several proposals are based on this filter, such as Simple online and real-time tracking (SORT) [27], DeepSORT [11], and MF-SORT [24]. The association between frames can use appearance and motion features [13], the bounding boxes' intersection over union (IoU) in consecutive frames [28], visual aspects of players [29] and geometric cues [26] can be added in the matching between video frames. Particle filter [30] and k-shortest paths (KSP) tracking [31, 32] are alternatives to KF. To improve the tracking due to occlusions, ensemble methods are used. For example, simple trackers can be sampled from a set of trackers using Markov Chain Monte Carlo (MCMC) [33]. However, this method needs manual feature engineering and is slow to run online. Deep networks were also applied to solve the tracking problem. Long Short-Term Memory (LSTM)s can concatenate high-level visual features produced by a convolutional neural network (CNN)s with region information to track the players [12]. The long-term behavior of the players can be extracted using deep hierarchical networks [4]. Siamese network [34] composes a correlation filter used to discriminate between images and their translations in consecutive frames. Despite the improvements, the DL algorithms are slower than filters. When two players are close to each other, their identities can easily be switched. To address this problem, researchers have explored appearance models [35, 36], motion models [37, 38], or a mix [32].

Other approaches to detection Other tracking methods also still exist that do not require video input. For example, Yurko et al. [1] used RFID tags, but it requires a large amount of hardware to be installed on-site, as well as the cooperation of both teams for the installation of the RFID chips on their players.

Summary: Most of the current approaches focus only on one aspect: detection or tracking. Our solution provides a complete pipeline that works with images captured by a camera. These images are processed for the identification of players and the field and subsequent tracking of in-field players. Our approach, described in the following sections, uses tracking-by-detection, where we balance the system performance vs the result's quality to process high-quality video.

3 Proposed solution

In this section, we describe our modular video processing pipeline, shown in Figure 1. It identifies and tracks field hockey players using a low-cost camera as an input. The main components are *field detection*, *player detection*, *team labeling*, and *tracking* modules. The camera streams the video, which is captured by the system frame by frame. An image flows in the pipeline from left to right. Each frame is forwarded to a worker that detects the players. There are two processes depending on the execution stage of the system: (1) "Run on startup" module runs at initialization and are shown at the top of the figure, and (2) after startup, the system runs modules in "Run on multiple workers". These stages are explained below. "Run on startup" and "Dedicated worker" run in the same worker, because they don't compete for resources.

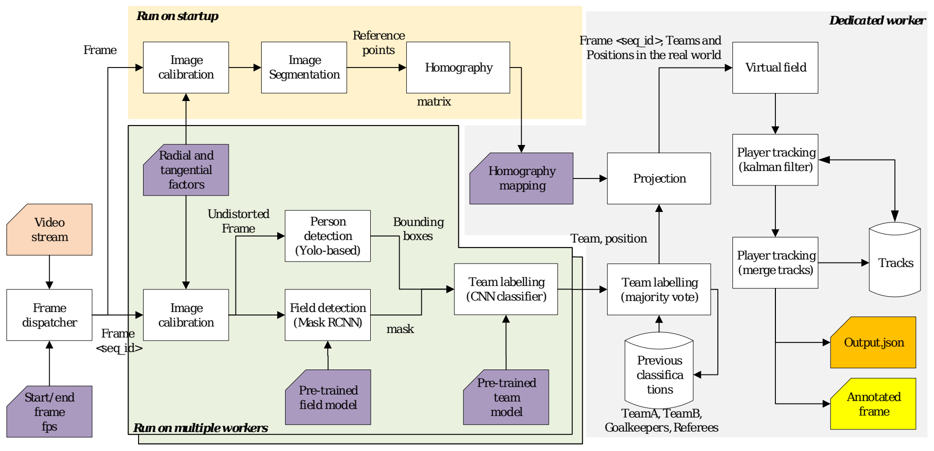

Figure 1. Proposed pipeline

During start-up, the first frames are used to generate the homography matrix, which is used to convert the pixel coordinates (image) in coordinates in the virtual field, i.e., coordinates measured in meters in a Cartesian system [39, § 2.1.2]. As there are few reference points in a field hockey pitch, several frames are captured to compute a robust position of the reference point, improving the homography's quality. After this step, the workers that perform the detection are activated. A dispatcher selects the worker (e.g., via round-robin) to receive a frame, which has a unique sequential identification number that allows the results to be sorted in the tracking step. The image calibration corrects the received image (e.g., removes distortions), which depends on parameters of the camera, lens type, and resolution. Next, a module detects people in the (corrected) image, which, combined with the field detector, identifies the players inside the pitch. In-field players are also tracked. We will explain below the main modules.

Field detection: We tested two ways to detect the field: (1) using Mask region-based CNN (R-CNN) network [40, 41], which obtain a polygon that represents the limits of the pitch; and (2) using instance segmentation based on Mask R-CNN and Hough transform to detect the pitch lines, the goals, and the penalty mark. With the lines identified by the segmentation, the system can say which are the external lines of the pitch that demarcate the limits of the field.

Player detection: We tested two different approaches: (1) based on Yolo v5, which outputs the bounding box of the detected players; and (2) based on Mask R-CNN [40] that segments the image on a pixel-to-pixel basis. The former is less complex, thus it smaller and faster. However, with the finer granularity of the second solution's output, it can show better results in occlusion cases (due to being able to detect a person based on only a part).

Team labeling: The images of all the people in the field are sent to a module that classifies them into two teams, referees, keepers, and unknown. We also tested two approaches. The first uses visual cues (characteristics). As the images of some players are very small, like one can see in Figure 2, we cannot use more

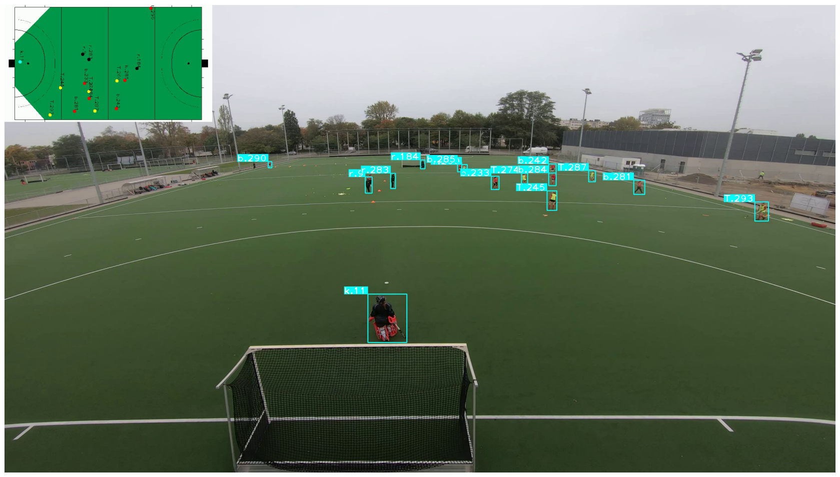

Figure 2. Result from the pipeline

sophisticated methods such as identifying faces or reading texts on the player's jersey. A CNN is used for feature extraction, *i.e.*, find the important characteristics of the players, and classify the detected people into both teams. The second method is based on a clustering method and separates players belonging to team 1 from those from team 2 using density-based spatial clustering of applications with noise (DBSCAN) [42, 43]. The labels outputted by DBSCAN are used to split the player detection into two teams and non-players (the outliers). We use DBSCAN for two reasons: (a) few hyperparameters need to be configured and (b) the inherent ability of the method to deal with outliers (referees and keepers).

Tracking: The pixel coordinates obtained by the player detection module are converted to the coordinates of the virtual field. We tested two methods to track the players. The first uses bipartite matching [44, 45], which joins elements from two sets (subsequent images) while keeping the cost of the edges (similarity metric) as low as possible. A simple metric is a Euclidean distance between the coordinates on frames $t-1$ and t. The results are improved using the predicted position on t of detection on $t-1$. This prediction uses simple motion equations. The second method uses KF as a corrective predictor filter by estimating the process's hidden state that predicts where the player is in the next frame. Ideally, in both cases, if those predictions are accurate, the distance will be zero. The changes between consecutive frames are small. For example, using a 4000x3000 pixels image at 30 frame-per-second (fps), one meter corresponds to 72 pixels in our video, and a person running at 12 m/s moves 29 pixels between two consecutive frames. Thus, the similarity can be (1) the distance between the center of the predicted and the detected bounding box or (2) the IoU of these two boxes, as there will always be some overlap. For any of the two forms, the assignment problem can be written, respectively, as the minimization of the sum of the distances or the maximization of the sum of the IoU. Due to occlusions, player overlapping, and conflicts, the tracking algorithm can break the track (it considers that a new player has been identified, spawning a new track). To reduce this problem, an additional module has been added to combine old tracks with newly created tracks, which is based on [32, 46] but adapted to run online.

4 Experimental setup

The pipeline runs on a machine with 32 MB of RAM, 4 Intel(R) Xeon(R) Silver 4108 CPU @ 1.80GHz, and one GeForce GTX 1080 Ti. The machine also runs Ubuntu 20.04 LTS with Python 3.8 and PyTorch 1.5.0. The camera is a Go Pro Hero7 Black running with 1080p at 30 fps.

Player detection: The first player detector is based on Yolo v.5 [47], with pre-trained parameters trained with 5000 COCO val2017 images [48]. We trained the two-class classifier (person or background) on a private dataset (900 annotated images taken from field hockey videos filmed with a GoPro Hero 8). The second detector uses Mask R-CNN, trained with 900 annotated frames. The network parameter was initialized with pre-trained values of 50-layer residual nets.

Field detection: The field detector based on Mask R-CNN was trained using the same dataset described. To assist the detection mechanism, we superimpose each frame with a 5-pixel-wide blank border. We calculate the convex hull of the network's output. The second method uses image segmentation with fully convolutional networks for semantic segmentation (FCN) [49] based on the implementation in [50]. We added new backbone models based on layer residual nets (RESNET)-18, -34, -50, and -101. The best results were obtained using RESNET-101, which also showed good results in [51]. The network was trained using 120 images annotated by the authors.

Team labeling: The classifier requires that labeled data about the teams is available before the match. The classifier is a multi-layer network composed of two convolution layers (CL) and four fully connected layers (FC), which use ReLU as activation. The CL are followed by Max Poll layers with $kernel = (2, 2)$ and $stride = 2$. The second method does not need previous training. It assembles clusters based on visual characteristics. The outliers are the goalkeepers and the referees, who use jerseys of different colors from both teams. Notice that DBSCAN cannot distinguish the referees from the goalkeepers.

Tracking: We consider only the last-second window for prediction. The KF's hidden state corresponds to the position, speed, and acceleration of the bounding box's coordinates. Our approach models its transitions as constant acceleration.

5 Experimental results

Figure 2 shows an output generated by the pipeline. We clearly see that (1) the identified players are marked with bounding boxes in the image, with players and others who are out of the field being excluded by the tracking, and (2) the positioning of the players in the image is converted into a virtual field at the top left of the figure. In the virtual field, the area indicated in green corresponds to the visual field captured by the camera. Below we discuss each module.

Player detection: We tested two flavors: *v5s* and *v5m*. The results for *v5m* are, in general, better than *v5s* (F1 score 81.1% vs. 79.3%). This is expected because *v5m* has a bigger network, which can capture more nuances. However, we opt to use *v5s* due to its faster running time. Thus, from here onward, we only show results with Yolo *v5s*. The detection module takes on average $60.590 \pm 0.944ms$ to run with the Yolo-based player detection, while the Mask R-CNN module runs in 2.445 ± 0.009 seconds. Thus the Yolo-based module is 40x faster than the Mask R-CNN module, which is a much bigger network. In both methods, one important parameter that controls the number of detected people is the threshold used to select if detection is a person or not. Choosing the right threshold is hard. If it is too high, it may cause a lot of people in the image to not be detected. The current threshold is set to 0.5 for both methods. Most of the cases where detection problems occur refers to players that were positioned far away from the camera. We also argue that using two cameras, one behind each goal, solves the detection problems in our footage, except occlusion.

Field detection: Using the CNN, we obtained a 98.9% accuracy with a mean IoU of 72.8% for the entire test set. Analyzing the average IoU values per class, we see the worst result is the identification of the shooting circle (curved line), followed by the lines inside the pitch. The shooting circle is actually composed of a line parallel to the goal line closed by curved edges. The algorithm confuses the straight part with the rest of the lines in the field. On the other hand, the internal field lines do not have visual cues capable of distinguishing one from the other. The distinction between them is purely based on distance. We leave for future work the incorporation of factors capable of distinguishing these lines in the segmentation. The CNN-based field detection on average takes 0.217 ± 0.021 s to run, while the image segmentation runs in 1.384 ± 0.005 s. It is 6x faster than the segmentation method, as it uses a smaller network and the segmentation involves more complex operations. It is key that the identified field mask produced by the detector follows the lines that define the pitch as close to perfection as possible, to exclude players who wait near the outer lines. Both algorithms tested in this section showed good adherence to these lines in the plane close to the camera. However, the more distant lines, particularly the bottom line of the pitch, are not well recognized because they are quite blurred, making the task difficult even for the authors who made the filming. The use of two cameras positioned on opposite sides of the field will again improve the result since the algorithm only has to deal with the nearest lines.

Team labeling: The main advantage of the CNN-based method is that it has higher accuracy (98% vs. 69.2%). We ran this method over a video with 2800 frames for 5 different videos, from which we created a dataset with about 10,000 annotated images of people in the pitch. The dataset was divided into train and test sets in a proportion of 6:4. The second method uses clustering based on DBSCAN. To configure the hyperparameter ϵ, DBSCAN searches the suitable value during the startup. The running time during clustering averages 1.812 ± 0.0230 ms. This method is about 20x faster than using CNN. However, this method does not work well due to the assumption that there are always two

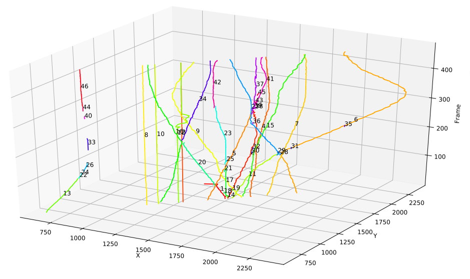
Figure 3. Only the simple tracking

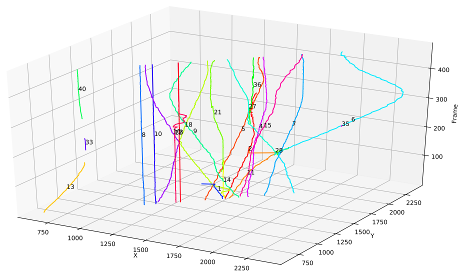
Figure 4. Merged tracks

teams in the image and that the number of players is much greater than that of non-players. If these conditions are not met, the algorithm fails completely. Another problem is related to the features used by DBSCAN. In our case, we use color histograms to characterize the players. These histograms can present great variations due to lighting, making the similarity metric values quite different for players of the same type, leading to classification errors. This is aggravated if the teams use uniforms with similar colors. The use of more complex identification methods with DBSCAN increases the runtime that makes it similar to the use of identification via CNN, which showed higher accuracy than DBSCAN.

Tracking: We tested two methods to track the players: bipartite matching and Kalman-based tracker. The first approach takes $383.447 \pm 0.170\ ms$, while the second takes $3.466 \pm 1.010 ms$ (110x faster). The last stage of the pipeline's processing path consists of merging tracks. In Figure 3, we see in the 3D representation of each of the tracks (as a curve) generated by the tracking process. The X and Y axes represent the coordinates in the virtual field (relative to the width and depth of the pitch), while the Z-axis corresponds to the frame where the detection was made. Over time, each curve grows upwards on the Z-axis and the position (x, y) consists of the location of the player for a given z frame. Figure 4 shows the tracks after merging. Thirty-two tracks were altered out of 46. The method took $11.4ms$ on average per loop (the standard deviation was $229\mu s$). The merging mechanism presents problems for joining tracks with a large gap in the detection (*e.g.* tracks 15, 33, and 40 correspond to the same player in Figure 4). However, they were not joined because the interval without detection was higher than the limit defined in the system. If this threshold is too small, a short period where the player is not detected or is occluded disconnects the tracks. However, if it is too large, tracks from different players can be connected. In our example, the merging modules fail in only one case due to a lack of player detection. In the other cases, occlusion and collision were the causes that lead to the disconnected tracks. This problem may be reduced (or even solved) using temporal correlation via attribute matching as proposed in [2], which can be tested in future work.

System's Accuracy: We analyze each frame in a 01:30 minute video, comparing it with human-provided ground truth. The results obtained with Yolo are better than with Mask R-CNN, where the accuracy is 93.8% and 70.9%, respectively. Both methods show two problems: (1) detect fewer players; or (2) detect more players than the total amount in the pitch. Those occur because: (a) the detection algorithm cannot recognize the player; (b) occlusion; and (c) duplication, *i.e.*, detection generates two bounding boxes for the same player. Yolo shows more over-detection cases, and because Mask R-CNN uses segmentation to generate the detection mask, it produces fewer duplication errors since the mask generation unites the parts into a consistent whole. In Yolo, it is necessary to regulate the overlapping of bounding boxes manually. The main error found in our studies is the algorithm provides two bounding boxes for different parts of the player's body or, when two players collide, the algorithm produces multiple detection. Some collisions can be solved by the tracking module. Offline methods provide better results than online methods because offline can use future tracking points [11, 27]. Online can be improved by enhancing tracking, using multiple cameras with different fields of view, or using a small delay in the transmission. However, the problem posed by occlusion is still an open topic.

6 Conclusion

Our solution uses the tracking-by-detection paradigm: the object detector first finds all objects of interest in a frame, and the tracking algorithm associates these objects in consecutive frames. We created a full pipeline that was tested on field hockey videos. Two approaches were tested for each of the main modules, and their performance is compared and discussed in Section 5. The player detection module directly influences the pipeline output's quality. The better the detection, the easier the tracking module work, as fewer players are missing or occluded. However, some problems cannot be solved by the detection module alone, *e.g.*, if a player is hidden by another player, the detection module is incapable of solving this problem. Thus, it is necessary that the system has memory and is capable of predicting the existence of this hidden player, which is a hard online task. Missing detection causes a large number of ID switches and fragmentation, which significantly degrades the tracking quality, especially during occlusion, collision, and crossings. Future work will consider better methods for movement patterns forecasting and also consider the detection and tracking of the ball, which is much harder than in tennis (little obstruction/high contrast) and soccer (larger ball). Also in our footage the players don't wear their number on the jerseys. This can be used to improve the player identification. Also, the extracted features from players (used in team labeling) can be to improve the identification.

Acknowledgment

This work was funded by the DAIQUIRI project, cofunded by imec, a research institute founded by the Flemish Government. Project partners are Ghent University, InTheRace, Arinti, Cronos, VideoHouse, NEP Belgium, and VRT, with project support from VLAIO under grant number HBC.2019.0053.

Bibliography

[1] R. Yurko, F. Matano, L. F. Richardson, N. Granered, T. Pospisil, K. Pelechrinis, and S. L. Ventura, "Going deep: models for continuous-time within-play valuation of game outcomes in american football with tracking data," *Journal of Quantitative Analysis in Sports*, vol. 1, no. ahead-of-print, 2020.

[2] H. Sabirin, H. Sankoh, and S. Naito, "Automatic soccer player tracking in single camera with robust occlusion handling using attribute matching," *IEICE TRANSACTIONS on Information and Systems*, vol. 98, no. 8, pp. 1580–1588, 2015.

[3] D. Linke, D. Link, and M. Lames, "Football-specific validity of TRACAB's optical video tracking systems," *PloS one*, vol. 15, no. 3, p. e0230179, 2020.

[4] S. Zheng, Y. Yue, and P. Lucey, "Generating long-term trajectories using deep hierarchical networks," *arXiv preprint arXiv:1706.07138*, 2017.

[5] B. Macdonald, "Recreating the Game: Using Player Tracking Data to Analyze Dynamics in Basketball and Football," *Harvard Data Science Review*, vol. 2, no. 4, 2020.

[6] V. Vovk, S. Skuratovskyi, P. Vyplavin, and I. Gorovyi, "Light-Weight Tracker for Sports Applications," in *2019 Signal Processing Symposium (SPSympo)*. IEEE, 2019, pp. 251–255.

[7] P. Felzenszwalb, D. McAllester, and D. Ramanan, "A discriminatively trained, multiscale, deformable part model," in *2008 IEEE conference on computer vision and pattern recognition*. IEEE, 2008, pp. 1–8.

[8] W.-L. Lu, J.-A. Ting, J. J. Little, and K. P. Murphy, "Learning to track and identify players from broadcast sports videos," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 7, pp. 1704–1716, 2013.

[9] E. Cheshire, C. Halasz, and J. K. Perin, "Player tracking and analysis of basketball plays," in *European Conference of Computer Vision*, 2013.

[10] G. Csanalosi, G. Dobreff, A. Pasic, M. Molnar, and L. Toka, "Low-cost optical tracking of soccer players," in *International Workshop on Machine Learning and Data Mining for Sports Analytics*. Springer, 2020, pp. 28–39.

[11] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *2017 IEEE international conference on image processing (ICIP)*. IEEE, 2017, pp. 3645–3649.

[12] G. Ning, Z. Zhang, C. Huang, X. Ren, H. Wang, C. Cai, and Z. He, "Spatially supervised recurrent convolutional neural networks for visual object tracking," in *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2017, pp. 1–4.

[13] G. Khan, Z. Tariq, and M. U. G. Khan, "Multi-person tracking based on faster R-CNN and deep appearance features," in *Visual Object Tracking with Deep Neural Networks*. IntechOpen, 2019.

[14] J. Komorowski, G. Kurzejamski, and G. Sarwas, “Footandball: Integrated player and ball detector,” *arXiv preprint arXiv:1912.05445*, 2019.

[15] X. Tong, J. Liu, T. Wang, and Y. Zhang, “Automatic player labeling, tracking and field registration and trajectory mapping in broadcast soccer video,” *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 2, pp. 1–32, 2011.

[16] L. Gu, X. Ding, and X.-S. Hua, “Online play segmentation for broadcasted american football tv programs,” in *Pacific-Rim Conference on Multimedia*. Springer, 2004, pp. 57–64.

[17] M.-H. Hung, C.-H. Hsieh, C.-M. Kuo, and J.-S. Pan, “Generalized playfield segmentation of sport videos using color features,” *Pattern Recognition Letters*, vol. 32, no. 7, pp. 987–1000, 2011.

[18] N. Homayounfar, S. Fidler, and R. Urtasun, “Sports field localization via deep structured models,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5212–5220.

[19] L. Leal-Taixé, C. Canton-Ferrer, and K. Schindler, “Learning by tracking: Siamese CNN for robust target association,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2016, pp. 33–40.

[20] S. Schulter, P. Vernaza, W. Choi, and M. Chandraker, “Deep network flow for multi-object tracking,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6951–6960.

[21] S. Sharma, J. A. Ansari, J. K. Murthy, and K. M. Krishna, “Beyond pixels: Leveraging geometry and shape cues for online multi-object tracking,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 3508–3515.

[22] K. Fang, Y. Xiang, X. Li, and S. Savarese, “Recurrent autoregressive networks for online multi-object tracking,” in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2018, pp. 466–475.

[23] Y. Xu, X. Zhou, S. Chen, and F. Li, “Deep learning for multiple object tracking: a survey,” *IET Computer Vision*, vol. 13, no. 4, pp. 355–368, 2019.

[24] H. Fu, L. Wu, M. Jian, Y. Yang, and X. Wang, “MF-SORT: simple online and Realtime tracking with motion features,” in *International Conference on Image and Graphics*. Springer, 2019, pp. 157–168.

[25] G. Ciaparrone, F. L. Sánchez, S. Tabik, L. Troiano, R. Tagliaferri, and F. Herrera, “Deep learning in video multi-object tracking: A survey,” *Neurocomputing*, vol. 381, pp. 61–88, 2020.

[26] M. H. Nasseri, H. Moradi, R. Hosseini, and M. Babaee, “Simple online and real-time tracking with occlusion handling,” *arXiv preprint arXiv:2103.04147*, 2021.

[27] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, “Simple online and realtime tracking,” in *2016 IEEE international conference on image processing (ICIP)*. IEEE, 2016, pp. 3464–3468.

[28] E. Bochinski, V. Eiselein, and T. Sikora, “High-speed tracking-by-detection without using image information,” in *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. IEEE, 2017, pp. 1–6.

[29] M. Manafifard, H. Ebadi, and H. A. Moghaddam, "A survey on player tracking in soccer videos," *Computer Vision and Image Understanding*, vol. 159, pp. 19–46, 2017.

[30] S. Murray, "Real-time multiple object tracking-a study on the importance of speed," *arXiv preprint arXiv:1709.03572*, 2017.

[31] J. Berclaz, F. Fleuret, E. Turetken, and P. Fua, "Multiple object tracking using k-shortest paths optimization," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 9, pp. 1806–1819, 2011.

[32] Q. Liang, W. Wu, Y. Yang, R. Zhang, Y. Peng, and M. Xu, "Multi-Player Tracking for Multi-View Sports Videos with Improved K-Shortest Path Algorithm," *Applied Sciences*, vol. 10, no. 3, p. 864, 2020.

[33] J. Kwon and K. M. Lee, "Tracking by sampling trackers," in *2011 International Conference on Computer Vision*. IEEE, 2011, pp. 1195–1202.

[34] J. Valmadre, L. Bertinetto, J. Henriques, A. Vedaldi, and P. H. Torr, "End-to-end representation learning for correlation filter based tracking," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2805–2813.

[35] H. B. Shitrit, J. Berclaz, F. Fleuret, and P. Fua, "Tracking multiple people under global appearance constraints," in *2011 International conference on computer vision*. IEEE, 2011, pp. 137–144.

[36] T. Kang, Y. Mo, D. Pae, C. Ahn, and M. Lim, "Robust visual tracking framework in the presence of blurring by arbitrating appearance-and feature-based detection," *Measurement*, vol. 95, pp. 50–69, 2017.

[37] J. Liu, P. Carr, R. T. Collins, and Y. Liu, "Tracking sports players with context-conditioned motion models," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1830–1837.

[38] Z. Li, S. Gao, and K. Nai, "Robust object tracking based on adaptive templates matching via the fusion of multiple features," *Journal of Visual Communication and Image Representation*, vol. 44, pp. 1–20, 2017.

[39] R. Szeliski, *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[40] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[41] W. Abdulla, "Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow," https://github.com/matterport/Mask_RCNN, 2017.

[42] E. Schubert, J. Sander, M. Ester, H. P. Kriegel, and X. Xu, "DBSCAN revisited, revisited: why and how you should (still) use DBSCAN," *ACM Transactions on Database Systems (TODS)*, vol. 42, no. 3, pp. 1–21, 2017.

[43] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *Kdd*, vol. 96, no. 34, 1996, pp. 226–231.

[44] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[45] J. Munkres, “Algorithms for the assignment and transportation problems,” *Journal of the society for industrial and applied mathematics*, vol. 5, no. 1, pp. 32–38, 1957.

[46] X. Zhou, V. Koltun, and P. Krähenbühl, “Tracking objects as points,” in *European Conference on Computer Vision.* Springer, 2020, pp. 474–490.

[47] Ultralytics, “YOLOv5 in PyTorch,” https://github.com/ultralytics/yolov5/tree/v4.0, January 2021.

[48] T.-Y. Lin, G. Patterson, M. R. Ronchi, Y. Cui, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, L. Zitnick, and P. Dollár, “Coco common object in context - 2017 dataset.” [Online]. Available: https://cocodataset.org/

[49] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[50] M. P. Shah, “Semantic segmentation architectures implemented in pytorch.” *https://github.com/meetshah1995/pytorch-semseg*, 2017.

[51] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.