3D Player Position Tracking via Multi-view Streams

Yuchang Jiang
D-BAUG, ETH Zurich
yujiang@ethz.ch

Ying Jiao
D-INFK, ETH Zurich
yijiao@ethz.ch

Yelan Tao
D-MAVT, ETH Zurich
yeltao@ethz.ch

Tianyu Wu
D-BAUG, ETH Zurich
wuti@ethz.ch

Abstract

Player tracking in soccer games provides abundant information to help coaches design optimal strategies as well as enhance players' performance. However, accurate player tracking is challenging due to frequent occlusions and the high moving speed of players. Based on several synchronous multi-view streams of a soccer game, we successfully construct a working pipeline for 3D soccer player position tracking. The proposed pipeline includes the multi-object tracking in each camera view, the projection of the 2D track onto the soccer pitch, and the multi-camera association. We experiment with the pipeline on two types of streams: the moving cameras, and the fixed cameras, and evaluate the tracking performance respectively. Additionally, we explore different setups of the cameras and study the sufficient number of cameras required to track all players. Our codes will be publicly available (`https://github.com/SherryJYC/3D-Tracking-MVS`).

1. Introduction

Player positions during a soccer match are one of the key performance indicators in the modern match analysis, which not only signal the physiological demands of individuals but also provide an overview of the team composite in a match (8). Player position tracking can be formulated as a multi-object tracking task and has been well studied in many computer vision works using single video streams. However, due to the fast movement and physical occlusions of players, the performance of continuous player tracking is generally limited, which leads to the solution of using multiple cameras for the tracking task. The goal of our project is to develop a robust system that identifies and tracks soccer players automatically from several synchronized multi-view streams and outputs the 3D trajectories of players on the soccer pitch.

Soccer player tracking is a complex and challenging task. The player's 3D localization is affected when players frequently interchange or when they gather together in a corner kick. Moreover, the re-identification of the same player can be difficult because of the similar appearance of the players as well as various illumination conditions. A common strategy to distinguish different players is to detect the color and the number of their jersey; however, there are usually serious deformations of the numbers encounter due to the movement of the players, the low resolution of the cameras, and the occlusions in some viewpoints.

The major contribution of this work is to achieve 3D player position tracking with a multi-camera network. This project also sets out to investigate the minimum number of cameras sufficient to track all players.

2. Related work

2.1. Object detection

Player detection is the first step in sports applications such as occupancy analysis, tracking, and performance analysis, etc (24). However, one should expect some noise caused by e.g. other moving objects, similar colors in the foreground and background, changing illumination, and shadows.

Though classic object detection algorithms such as AdaBoost have been widely applied for player detection tasks (e.g. (10), (12), and (16)), recent studies with neural-network-based detectors have demonstrated significant improvements in performance (15). One typical example is the two-stage detector (e.g. Region-based convolutional neural network (RCNN)), which generates region proposals in the first stage then refines the bounding boxes from those proposals in the second stage (22). Variants in the RCNN family of detectors have been applied to person tracking including Faster RCNN which passes object proposals for the object classification and the bounding boxes regression (1) and pose-guided RCNN which aimed at jersey number detection (14). Faster RCNN is used as the object detector in our pipeline.

2.2. Multi-object tracking in the single camera

Tracking players is difficult due to their high speed. A practical player tracker should track multiple people accurately for consecutive frames. For single-camera tracking, there are traditional methods and deep learning-based methods. Traditional tracking algorithms including the loopy inference on an underlying Bayes net (17), annealed particle filtering (7) and Gaussian Process Dynamical Models (GPDMs) (19). One can also build a model of appearance for each person and then track by detecting those models in each frame (18). Recently, deep learning methods have become increasingly popular, many of which follow the tracking-by-detection paradigm. This type of trackers uses object detectors such as Faster RCNN (20) to generate 2D detection and then feed the detection to a classic tracker (e.g. IoU tracker (3) or SORT (2)). One drawback of this approach is that the detection errors cannot be recovered in this two-stage pipeline. Further improvements can be achieved by combining detection and tracking into a single framework; for example, Tracktor (1), a unified framework that takes the detection from the previous frame as the region proposals to regress the detection in the current frame. Several extensions exist for Tracktor, which are designed to compensate for camera motions and support for re-identification. In this study, the variant Tracktor++ is adopted for single-camera multi-object tracking and is compared with the other two baseline tracking-by-detection methods.

2.3. Multi-object tracking in multiple cameras

Multi-object tracking in multiple cameras has been an active research area in recent years, many new approaches have been proposed to solve this challenge. The key idea is to associate the 2D detection in each view with a multi-way matching algorithm. On the one hand, the detection can be clustered using several matching algorithms such as fast greedy algorithm (4), iterative greedy matching (23), etc. Each resulting cluster encodes 2D positions of the same player across different views and the consistent correspondences across the views make the inference of the player's 3D position possible (9). On the other hand, by exploiting the known poses of the cameras and the planar movement of the objects (as prior knowledge), the data association problem can be tackled by projecting the 2D positions from different camera views onto the same plane and solving the pairwise assignment in 2D with a cost matrix (5).

Nevertheless, the heavy occlusions in each view can bring up the challenge of the ambiguity in the cross-view association and have a negative effect on the accuracy of the matching. To improve the accuracy, one can exploit the temporal consistency in videos to match the 2D inputs with existing 3D positions instead of associating 2D poses among all pairs of views from scratch at every frame (6).

3. Problem Statement

In this study, we considered the 3D soccer players tracking task in the multi-view streams as a multi-target multi-camera tracking problem. Mathematically, video sequences $V = \{V_1, ..., V_k\}$ from k cameras in a synchronised calibrated camera network with overlapping views are treated as inputs and l trajectories with 3D positions, $T = \{T_1, ..., T_l\}$, are defined as outputs. The goal of this work is to design and evaluate a full pipeline to generate 3D tracks from multi-view videos.

4. Method

The main pipeline developed in this work is divided into two stages: the single-camera multi-object tracking and the multi-camera multi-object tracking. The detailed methods are explained in sequence.

4.1. Multi-object tracking in the single camera

We conduct the single-camera multi-object tracking based on Tracktor++ proposed by (1). The tracking pipeline consists of two main processing steps: i) use the detection of the existing tracks in the previous frame as object proposals to regress new positions in the current frame and leverage the object classification scores to kill potentially occluded tracks; ii) initialize a new track if no substantial Intersection over Union is found with the active tracks. Tracktor++ extends this vanilla Tracktor by integrating a re-identification algorithm (21), which reserves the deactivated tracks for a fixed number of frames to achieve continuous tracks when players are temporally out of the camera view or occluded. We further added an alignment function based on the provided camera calibration results to compensate for large camera motions. Both extensions are used for improving identity preservation across frames.

The regression-based object detector is the core element of this tracking pipeline. To fit our task better, we fine-tuned a Faster-RCNN (20) with ResNet-50 (11) and Feature Pyramid Networks (FPN) (13) pre-trained on COCO using the self-annotated data of the moving camera Left in timestamp 0125-0135.

4.2. Multi-object tracking in multiple cameras

To associate multiple cameras, this work leverages both the visual and the spatial constraint. In the following section, the re-identification (ReID) model that provides information for the visual constraint is discussed, followed by the description of the projection (2D-to-3D) procedure which accounts for the spatial constraint. Later, the algorithm of across-camera association built upon the two will be explained.

4.2.1 Team classification

The re-identification model in this study is based on the team classification results. A Support Vector Machine (SVM) is trained to classify players into different team based on the extracted color features. We obtained the training data by cropping bounding boxes from self-annotated videos. Team labels are generated by K-means clustering and manually correction.

4.2.2 Projection

With the 2D tracking results, the position of a player on the pitch (x_{pitch}) can be recovered by back-projecting his/her 2D position (x_{cam}) in the camera frame to acquire the 3D position in the world coordinate system (X) represented as a ray and finding the intersection of the ray with the ground plane (π). This can be achieved using the homogeneous coordinates together with the Plücker matrix of the line.

During a football match, players are assumed to run on the pitch most of the time; hence, it is natural to use the bottom center of the bounding box as the representation of players' position. Additionally, given the intrinsic and extrinsic of the camera, X can be solved using Equation 1. With X_cam and X, the line passing through these two points can be represented using the Plücker matrix as $L = X^T X_{cam} - X_{cam} X^T$. As the world coordinate system is realized in such a way that its origin lies at the center of the pitch, and the X, Y, and Z axis are aligned to the long side, up direction and short side respectively, the ground plane can be defined using its point-normal representation. Lastly, the players' positions on the pitch are computed with Equation 2.

$$X = P^{+}x \tag{1}$$

$$x_{pitch} = L\pi = (X^T\pi)X_{cam} - (X_{cam}^T\pi)X \tag{2}$$

4.2.3 Across-camera association

After the single-camera tracking and the projection of the results, the 3D positions of the tracked objects are matched and combined across cameras. Similar to (5), the targets are first initialized in the 3D coordinate frame with the first frames of the two cameras. Then more targets are reconstructed incrementally. Starting from two cameras, the across-camera matching is achieved incrementally using both spatial and visual features. More specifically, we focus on the distance of the 3D positions for the defined spatial and visual constraints. For the spatial constraints, we split the whole soccer pitch into four regions and constrained that the same targets must lie in the same region. For the visual constraints, we leverage the team ID extracted from ReID process and assign more penalties to a pair of objects with different team ID. As ReID results may contain more mistakes comparing to 3D positions, we put a more relaxed constraint on it. The goal of the across-camera matching is to perform track-to-track and track-to-target associations that minimize the distance cost, which can be formulated as:

$$argmin_x \sum_r \sum_c c_{r,c} x_{r,c} \tag{3}$$

$$c_{r,c} = dist(D_r, D_c) \tag{4}$$

where $c_{r,c}$ is the distance cost between two objects. If these two objects lie in different regions, the cost is infinity; if they belong to different teams, the distance cost is multiplied by a penalty coefficient, α. Besides, when performing track-to-target associations, the predicted 3D positions based on the previous position and estimated velocity is used ($D_{pred} = D_{last} + V_{est}(t_{pred} - t_{last})$). To handle the case of unmatched tracks (e.g. objects only observed in one camera), trash bin nodes are added to the cost matrix to help perform optimization.

With the formulated problem, we start with the track-to-track association to initiate the targets and then perform the track-to-target association to merge or add targets. For example, the main pipeline of the simplest case (i.e. matching between two cameras) can be divided into three steps: 1) At the initial frame, use Algorithm 1 to initialize the targets. 2) For each frame in the remaining frames, use Algorithm 1 to create temporal targets for this frame. 3) Compare the 3D positions of the temporal targets to existing targets to merge or add new targets.

To associate more than two cameras, a binary tree structure is considered to match the cameras incrementally, as sketched in Figure 1. For example, a two-camera association is firstly performed on the camera 1-camera 2 and the camera 3-camera 4 pairs. Then the same association is conducted on the previous result. In the end, all 4 cameras are associated together.

5. Experiment results

To test and evaluate the performance of our multi-camera tracking pipeline on 3D player position tracking tasks, we conducted several experiments on different variations of data as well as compared our results with the two off-the-shelf tracking methods: IoU tracker and SORT. We designed and carried out our experiments focusing on the following two aspects: 1) evaluate the pipeline of 3D player

Algorithm 1: Algorithm of across-camera matching with spatial and visual features

```
Input: 3D tracking results, $D_{t,c,lid}$, in camera $c$
        ($c \in C$) frame $t$ with local object ID $lid$
Output: New 3D targets $T_{t,gid}$ at time $t$ with global
         ID $gid$
$costmat_{NxM} \leftarrow$ inf;
$T \leftarrow \emptyset$ ;
for each $D_i in D_{t_0,c1}$ do
    for each $D_j in D_{t_0,c2}$ do
        if $D_i$ and $D_j$ in same region then
            $costmat[i][j] = dist(\mathrm{D}_i, D_j)$
            if not $D_i$ and $D_j$ in same team then
                $costmat[i][j]* = \alpha$ ;
            else
                continue
            end
        else
            continue
        end
    end
end
add trash bin nodes to $costmat$ ;
do Hungarian algorithm on $costmat$ ;
$T \leftarrow$ each matched pair as a new target ;
```

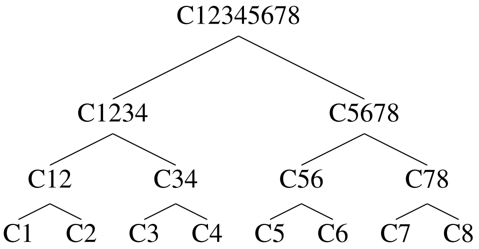

Figure 1. Tree-structured across-camera association

position tracking in multi-camera networks, including the evaluation of detection, single-camera tracking, and multi-camera tracking performances; 2) investigate the minimum number of sufficient cameras in a multi-camera network to track all soccer players. In the experiment, data from fixed cameras and moving cameras in the time range 0125 to 0135 are used.

5.0.1 Moving cameras data

Videos from three moving cameras (Left, Right, Cam1) are mainly used to evaluate the tracking pipeline, including 2D detection and tracking. Data of moving cameras include the video sequence, ground truth of 3D positions on the pitch (from *Tracab*), and ground truth of 2D tracking results (labeled by ourselves). In the experiment, data from camera Left is treated as the training data for Faster RCNN and data from other cameras are considered as testing data.

5.0.2 Fixed cameras data

Videos from eight fixed cameras (1 to 8) are employed to investigate the minimum number of cameras required to track all players. Therefore, we focus on the video sequences and ground truth of 3D positions on the pitch (from *Tracab*). With the trained model, different numbers and combinations of fixed cameras are leveraged to generate 3D tracking results and compared with ground truth.

5.1. Implementation details

Faster RCNN settings: the model is constructed with a ResNet-50-FPN backbone from torchvision [1] pre-trained on the COCO dataset. It is fine-tuned on the self-annotated Left video. Tracktor++ settings: the threshold of object classification score for killing a trajectory is 0.5. A new trajectory is initialized if the IoU with any of the active ones is below 0.3. Multi-camera tracker settings: the score of the dustbin node is 5, and the maximum frame number allowed for the disappearance is 5.

5.2. Results

In this section, each step of the tracking pipeline is evaluated, including the detection, single-camera tracking, and multi-camera tracking. Then the minimum number of sufficient cameras and its relationship with camera poses are investigated.

5.2.1 Evaluation of detection

Table 1 demonstrates the performance of the object detector on two moving cameras, Right and Cam1, during 0123-0135. Figure 2 and 3 plot their precision-recall curves. They show that our object detector performs slightly better on Right compared to Cam1, in terms of a higher AP (Average Precise) at an IoU of 0.5, less FP, and maintaining a higher precision when the recall is increasing. One possible reason for this result is that Right has a smaller camera motion and thus less blurring, which contributes to a more accurate detection.

	Total_TP	**Total_FP**	**Total_P**	**AP@.5**
Right	5928	3701	6028	97.87%
Cam1	2905	4443	2934	93.83%

Table 1. Object detector performance on Right and Cam1

[1] https://pytorch.org/vision/stable/_modules/torchvision/models/detection /faster_rcnn.html

5.2.2 Evaluation of single-camera tracking

The evaluation of the single-camera tracking results is based on the accuracy of the 2D bounding box. Specifically, five standard metrics for tracking evaluation are computed and analyzed. To better demonstrate the performance of the designed pipeline, we further compare our results acquired using Tracktor++ with the two baselines trackers, SORT and IoU tracker, based on the detections of the Faster RCNN in the previous stage. The tracking results from the two moving cameras (Cam1 & Right) in the period of 0125-0135 are used for this experiments.

Table 3 and 2 summarize the computed metrics of Cam1 and Right respectively. Compared with the two classic trackers, our approach with Tracktor++ consistently outperforms the other two for both cameras. Especially, Tracktor++ shows significant improvement on MOTA for Cam1 and a large drop of IDSW for Right. The performance of the pipeline on Cam1 is worse than the camera Right, which may occur due to the faster movement of the camera during the sample sequence leading to less reliable camera calibrations.

	IDF1↑	MOTA↑	MOTP↑	IDSW↓
IoU	61.8%	26.9%	0.336	46
SORT	58.3%	22.0%	0.340	27
Tracktor++	72.6%	62.8%	0.322	21

Table 2. Results (2D) of single-camera tracker on Right

	IDF1↑	MOTA↑	MOTP↑	IDSW↓
IoU	42.6%	-44.4%	0.355	53
SORT	43.8%	-28.2%	0.360	36
Tracktor++	49.3%	-29.0%	0.344	17

Table 3. Results (2D) of single-camera tracker on Cam1

Nevertheless, the pipeline is less confident in handling the occlusions of players, as shown in Figure 2, where one player is covered by another in the viewing direction, resulting in only one tracked bounding box. The situation can be typical, especially during a tight sport event as a football match, which also implies the need for multi-camera tracking pipelines that can mitigate the effect of occlusions by leveraging multiple viewing angles.

5.2.3 Evaluation of multi-camera tracking

To evaluate the absolute accuracy of the 3D players' positions, we compared predicted 3D coordinates on the pitch with the *Tracab* ground truth trajectories: the distance those two are calculated for each player. The errors with different experiment settings are summarized in Table 4. The mean error of the tracking predictions with the two moving cameras is around 5m, which is around the same level of accuracy achieved by the fixed camera 1 and 2. However, with other sets of fixed camera (i.e. fixed camera 3/4, fixed camera 5/6, and fixed camera 7/8), the mean errors of the tracking results are reduced to half. One possible attributing aspect can be the size of overlaps between the cameras. As can be seen in Figure 5, the overlapping area between the fixed camera 1 and 2 is relatively smaller than that of the fixed camera 5 and 6. Although the fixed camera 1 and 2 are physically mounted close to each other, they point to two different directions, whereas the overlapping area of the fixed camera 5 and 6 is larger and covers a larger portion of the pitch.

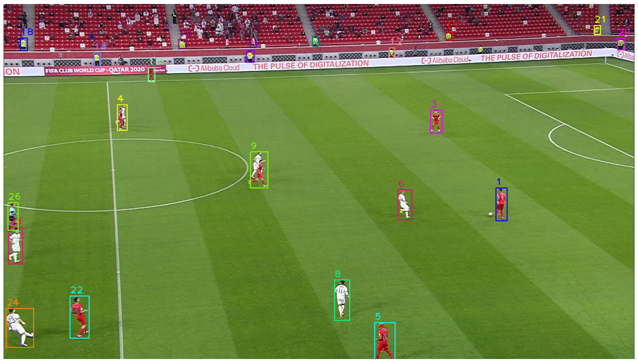
18
14
13
11
15
12
17
21
20
ON
FIFA CLUB WORLD CUP · QATAR 2020
Alibaba Cloud
THE PULSE OF DIGITALIZATION
Alibaba Cloud
THE PULSE OF DIGITALIZATION
4
3
9
26
23
6
1
24
22
8
5

Figure 2. An example of failure cases due to occlusions

	$Error_{mean}$	$Error_{std}$
2 Cam_{mov}	5.09	7.23
2 Cam_{fix} (1, 2)	5.78	10.52
2 Cam_{fix} (3, 4)	2.92	5.34
2 Cam_{fix} (5, 6)	2.80	5.60
2 Cam_{fix} (7, 8)	2.88	5.14

Table 4. Multi-camera tracking errors (meter) with two moving or fixed cameras

5.2.4 Investigation of the minimum number of sufficient cameras

The distinct performances of the different combinations of cameras naturally lead to the discussion of the optimal setup for a 3D player tracking task using a network of cameras and the minimum number of cameras to achieve a satisfactory result. Therefore, we conducted further investigations with the sample sequences taken by the eight fixed cameras during the same time frame of 0125-0135. The cameras used for tracking are combined incrementally following a tree structure. The effect of the camera number on the 3D tracking result from different combinations of cameras are displayed in Figure 3, and 4, where the y-axis is the 3D position error and the x-axis stands for the combination of cameras (e.g. 12 means associate camera 1 and 2). Here, two findings are observed. First, the 3D position error depends on the combination of cameras. The accuracy achieved us-

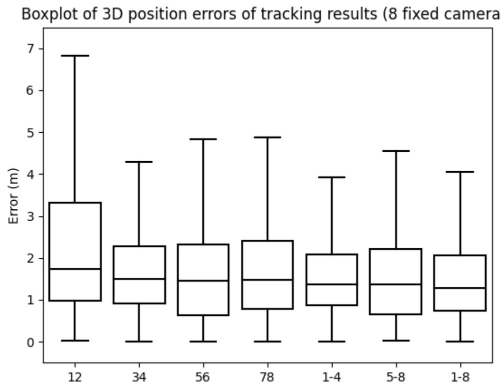

Figure 3. Boxplot of errors with 8 fixed cameras in sequence

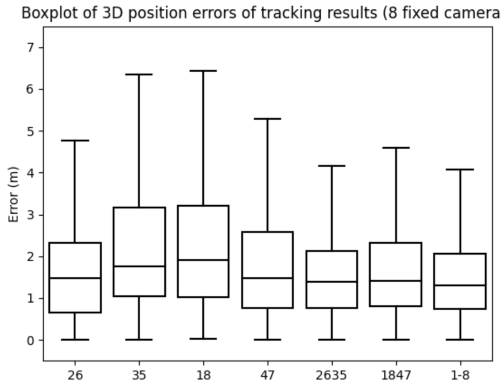

Figure 4. Boxplot of errors with 8 fixed cameras not in sequence

ing two out of eight cameras can be diverse depending on the positions of the cameras and the size of overlap. The other finding is the relationship between the number of cameras used and the position error. Although the performances with four cameras are better and more stable than two cameras, the performance using all eight cameras for tracking shows only marginally differences from a setup with four cameras, indicating incorporating more than four cameras may not be necessary in the case. As illustrated in Figure 7, the pitch region is already sufficiently covered in the four-camera setup; therefore, the benefit of further increasing the number of cameras is no longer significant.

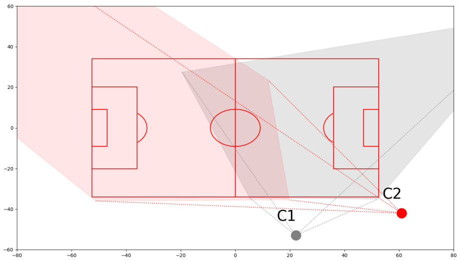

Figure 5. The positions of Camera 1-2

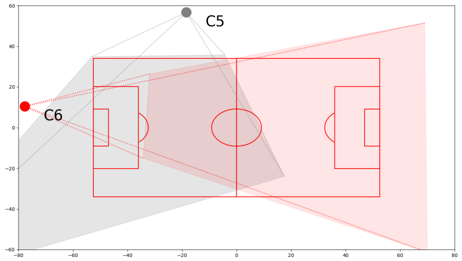

Figure 6. The positions of Camera 5-6

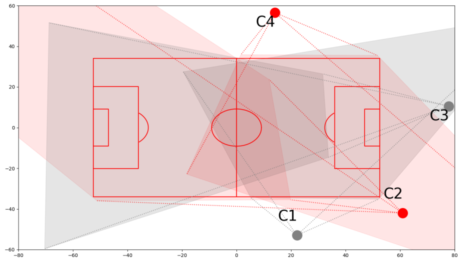

Figure 7. The positions of Camera 1-4

6. Conclusion

We successfully designed and implemented a multi-camera tracking pipeline for 3D player position tracking tasks in this project. The proposed method with Tracktor++ outperforms the two off-the-shelf tracking baselines (i.e. IoU tracker and SORT) in terms of better performance in several evaluation metrics. In our experiments, we observed that the quality of the camera calibrations is crucial to the overall performance of the pipeline. However, acquiring sufficiently good camera calibrations for those fast-moving cameras is a challenging and reoccurring task, especially for a fast-paced sport event. This issue may be of fewer troubles for a network of fixed cameras but the performance of fixed cameras heavily relies on the overlap views in the camera network. To achieve a reasonable level of accuracy, a certain number of cameras need to be installed (e.g. four cameras are needed based on our investigations), which may cause concern with the budget. On the other hand, utilizing moving camera sequences seems to be a more cost-effective choice as broadcasting streams are available for almost any sports events.

Moreover, we further explored the potential applications of our tracking pipeline. One example can be plotting the Voronoi of the tracked player positions to analyze further the attaching and defencing regions for each player and design a better winning strategy (Figure 8). In this regard, our project is of particular interest to players and coaches for training and instructing purpose. Additionally, the 3D tracking result on the pitch can also function as a guideline or reference for defining possession and events.

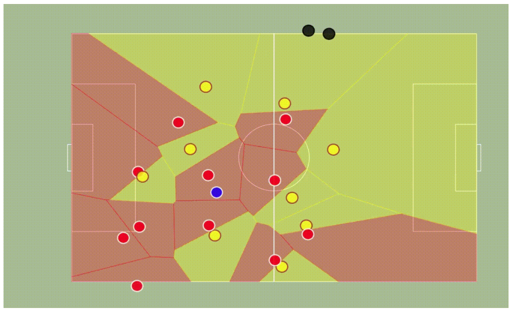

Figure 8. Application example of 3D tracking results

References

[1] P. Bergmann, T. Meinhardt, and L. Leal-Taixe. Tracking without bells and whistles. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 941–951, 2019.

[2] 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)*, pages 3464–3468. IEEE, 2016.

[3] 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)*, pages 1–6. IEEE, 2017.

[4] L. Bridgeman, M. Volino, J.-Y. Guillemaut, and A. Hilton. Multi-person 3d pose estimation and tracking in sports. pages 2487–2496, 06 2019.

[5] L. Chen, H. Ai, R. Chen, Z. Zhuang, and S. Liu. Cross-view tracking for multi-human 3d pose estimation at over 100 fps. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3279–3288, 2020.

[6] L. Chen, H. Ai, R. Chen, Z. Zhuang, and S. Liu. Cross-view tracking for multi-human 3d pose estimation at over 100 fps. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3276–3285, 2020.

[7] J. Deutscher, A. Blake, and I. Reid. Articulated body motion capture by annealed particle filtering. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No.PR00662)*, volume 2, pages 126–133 vol.2, 2000.

[8] V. Di Salvo, R. Baron, H. Tschan, F. C. Montero, N. Bachl, and F. Pigozzi. Performance characteristics according to playing position in elite soccer. *International journal of sports medicine*, 28(03):222–227, 2007.

[9] J. Dong, W. Jiang, Q. Huang, H. Bao, and X. Zhou. Fast and robust multi-person 3d pose estimation from multiple views. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7784–7793, 2019.

[10] H. Faulkner and A. Dick. Afl player detection and tracking. In *2015 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, pages 1–8, 2015.

[11] 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*, pages 770–778, 2016.

[12] Z. Ivankovic, B. Markoski, M. Ivkovic, D. Radosav, and P. Pecev. Adaboost in basketball player identification. In *2012 IEEE 13th International Symposium on Computational Intelligence and Informatics (CINTI)*, pages 151–156, 2012.

[13] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

[14] H. Liu and B. Bhanu. Pose-guided r-cnn for jersey number recognition in sports. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2457–2466, 2019.

[15] J. Liu, X. Tong, W. Li, T. Wang, Y. Zhang, and H. Wang. Automatic player detection, labeling and tracking in broadcast soccer video. *Pattern Recognition Letters*, 30(2):103–113, 2009. Video-based Object and Event Analysis.

[16] Z. Mahmood, T. Ali, and S. Khattak. Automatic player detection and recognition in images using adaboost. In *Proceedings of 2012 9th International Bhurban Conference on Applied Sciences Technology (IBCAST)*, pages 64–69, 2012.

[17] D. Ramanan and D. Forsyth. Finding and tracking people from the bottom up. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, volume 2, pages II–II, 2003.

[18] D. Ramanan, D. Forsyth, and A. Zisserman. Tracking people by learning their appearance. *IEEE transactions on pattern analysis and machine intelligence*, 29:65–81, 02 2007.

[19] L. Raskin, E. Rivlin, and M. Rudzsky. Using gaussian processes for human tracking and action classification. In G. Bebis, R. Boyle, B. Parvin, D. Koracin, N. Paragios, S.-M. Tanveer, T. Ju, Z. Liu, S. Coquillart, C. Cruz-Neira, T. Müller, and T. Malzbender, editors, *Advances in Visual Computing*, pages 36–45, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

[20] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *arXiv preprint arXiv:1506.01497*, 2015.

[21] E. Ristani and C. Tomasi. Features for multi-target multi-camera tracking and re-identification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6036–6046, 2018.

[22] P. Soviany and R. T. Ionescu. Optimizing the trade-off between single-stage and two-stage deep object detectors using image difficulty prediction. In *2018 20th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*, pages 209–214. IEEE, 2018.

[23] J. Tanke and J. Gall. Iterative greedy matching for 3d human pose tracking from multiple views. In G. A. Fink, S. Frintrop, and X. Jiang, editors, *Pattern Recognition*, pages 537–550, Cham, 2019. Springer International Publishing.

[24] G. Thomas, R. Gade, T. B. Moeslund, P. Carr, and A. Hilton. Computer vision for sports: Current applications and research topics. *Computer Vision and Image Understanding*, 159:3–18, 2017. Computer Vision in Sports.

A. Appendix

A.1. Work distribution

	Yelan Tao	**Ying Jiao**	**Yuchang Jiang**	**Tianyu Wu**
Labelling	✓moving cameras			
Data preprocessing	✓	✓	✓	✓
Single-camera tracking		✓Tracktor, ReID	✓evaluation	✓evaluation
Multi-camera tracking			✓MC association	✓projection, visualization
Documentation	✓	✓	✓	✓

A.2. Acknowledgement

We would like to thank the following resources:

- Data labelling: Supervisely
- Single-camera trackers:
  - Tracktor++
  - IoU tracker
  - SORT
- Visualization of Voronoi: Football crunching (footyviz)
- Evaluation: Pymotmetrics