





# FOOTBALLTrace: An AI-Based System for Football Player Tracking with Occlusion Detection and Trajectory Correction

Abdelrahman H. Mostafa<sup>1</sup><sup>a</sup>, Muhammad A. Rushdi<sup>1,2</sup><sup>b</sup>, Tamer A. Basha<sup>1</sup><sup>c</sup> and Khaled Sayed<sup>3</sup><sup>d</sup>

<sup>1</sup>Department of Systems & Biomedical Engineering, Faculty of Engineering, Cairo University, Giza, Egypt

<sup>2</sup>Department of Computer Science, New Giza University, Giza, Egypt

<sup>3</sup>Department of Electrical & Computer Engineering and Computer Science, University of New Haven, West Haven, CT, U.S.A.

**Keywords:** Multiple Object Tracking, YOLO, Soccer Player Tracking, Football Analytics.

**Abstract:** Data analytics have had a significant impact on tactical and workload planning in football. Football data is divided into two categories: event data, which captures on-the-ball events like passes and shots, and tracking data, which captures off-the-ball movements. However, traditional methods of collecting tracking data are expensive and inconvenient. Recently, AI solutions have emerged as low-cost and user-friendly alternatives to track players' movements from video streams. This paper introduces FOOTBALLTrace, an end-to-end AI system for tracking multiple football players from a panoramic game view. The system also incorporates a novel algorithm that detects potential occlusion events and ensures trajectory continuity for occluded players. The workflow involves five stages: panoramic view creation, player detection, player ID association, occlusion detection, and trajectory correction. The system utilizes YOLOv7 for multiple object detection and employs a pre-trained deep affinity network to assign unique IDs to players throughout the game. Occlusion detection and trajectory correction are achieved by extracting geometric features from discontinuous player trajectories. The system's performance was evaluated on full-length video data of a football game, with occlusion events manually extracted for training and testing the occlusion detection and trajectory correction algorithm. The system achieved an 87.5% trajectory correction rate for occluded trajectories.


## 1 INTRODUCTION


Football industry has been one of the most emerging industries in the world, with significant impact on economies of both developed and developing countries (Zhang, 2018; Dejonghe, 2007). For example, the English Football Premier league has made revenues of around 5.7 Billion GBP in season 2022/2023 (Premier League Clubs Revenue By Stream, 2023). This emerging importance of the football industry has grabbed the attention of the scientific community to utilize data in performance management, training-workload management, and optimizing the tactical training of professional players (Arnason, et al., 2004).


Data-driven football analytics require tools to collect data from football games efficiently, to be


used in tactical analysis and training-load planning (Pifer, 2019). Football data is classified according to the player's ball possession into event data and tracking data (Rein, 2016). Event data describes "on-the-ball" events such as passes, shots, duels, and goals. Event data is collected by manual operators with the aid of a special tagger software such as TAGG-MAKER (Tagg Maker, 2023) and ONCE (Once- Video Analysis, 2023).

On the other hand, tracking data includes all other "off-the-ball" events and players' positional data, encompassing all the necessary information for analysts and coaches to design training loads and tactical plans (Forcher, 2018; De Silva, 2018). Conventional tracking data acquisition is done by using wearable technologies such as GPS vests or wrist-bands with GPS modules (Tierney, 2019).

<sup>a</sup> <https://orcid.org/0009-0007-1338-3556>

<sup>b</sup> <https://orcid.org/0000-0001-9869-0270>

<sup>c</sup> <https://orcid.org/0000-0003-4431-8646>

<sup>d</sup> <https://orcid.org/0000-0002-1252-3170>

However, those conventional tracking approaches are facing some challenges related to their cost, players' safety, and players' acceptance to wear them during the game (Edgecomb, 2006). These challenges have demanded the development of cost-effective, contactless alternatives to acquire tracking data during football games.

Filming football games with high-resolution cameras paved the way towards computer-vision-based approaches to track players without the need for extra wearables (Thomas, 2017). Vision-based tracking data acquisition from the football game feed has shown promising results. However, certain in-game scenarios are currently hindering the practical use of this approach. Occlusion is one of the in-game-scenarios that occurs frequently during a game whenever players are in a duel or they move across each other (Sabirin, 2015). Detecting occlusion events and correcting the trajectories of occluded players will improve the performance of vision-based tracking system and output more usable tracking data.

Many multiple object tracking (MOT) paradigms have been introduced over the last two decades such as the tracking by detection paradigm (Andriluka, 2008) which starts with object (e.g., player) detection in a specific video frame, and constructs a feature vector for each detection that describes each detected object. Object detection is the input stage for any MOT pipeline, where the choice of the detector that has the best accuracy and highest speed is the key for efficient detections (Luo, 2021). Recently, deep neural networks architectures have shown significant performance in learning features and detecting objects, and are now the base of any state of the art object detector (Zhao, 2019). Detection is repeated for all objects in each frame, then frame-to-frame features are compared using similarity computation algorithms and objects with highly similar features are joined by a unique ID between frames. The process of associating objects sharing the same ID over multiple frames leads to the construction of an object's trajectory (Sun, 2019).

Effective tracking algorithms rely on accurate modeling of detected objects in each frame. There are two main approaches for modeling: appearance modeling and motion modeling (Luo, 2021). Appearance modeling is an object-descriptive approach that efficiently models objects in local regions. However, when similar objects appear in local regions, appearance modeling does not achieve the best performance. On the other hand, motion modeling adopts probabilistic approaches to model the dynamic behavior of the player (object) and predicts player's future locations according to a

statistical model. Both appearance and motion modeling approaches lack the ability to detect and/or correct the trajectories of occluded objects (Luo, 2021).



Figure 1: FOOTBALLTrace Workflow.

In this paper, we adopt the appearance modeling approach to propose an end-to-end AI-based system (Figure 1) called FOOTBALLTrace for acquiring low-cost, contactless tracking data for football analytics. Our system films the game with three fixed cameras with 25 frames per second, stitching the three camera views to create a panoramic view of the game, utilizes a state-of-the-art real-time object detector, YOLOv7, and a deep affinity estimation network to assign unique IDs for detected players throughout the entire game. Additionally, we introduce a novel algorithm for occlusion detection and trajectory correction to enhance the practicality of our system.

## 2 DATASET DESCRIPTION



Figure 2: Camera setup for panoramic view creation.

We used video data of a football game recorded with three 4K resolution Go-Pro cameras with wide view and 30% overlap between each camera view as seen in Figure 2. The 3 views are then stitched together using an image registration algorithm (Szeliski, 2007) to create a panoramic view of the pitch (depicted in Figure 3). The game video recording and panoramic view creation are carried out by KoraStats, Egypt (KORASTATS, 2023). The game is filmed at the Air Defense Stadium in Cairo, Egypt. The tracking coordinates are mapped to a pitch top view using Homography transformation between the pitch coordinates in the camera view and the corresponding coordinates in the pitch top view.



Figure 3: Panoramic View of the Pitch after Stitching.

Additionally, we created a dataset of 40 occlusion cases comprising 80 discontinued trajectories. We created this dataset by visually examining the tracking output on the panoramic view video. An occlusion case is defined by start and end timestamps, during which the trajectories of both the occluding and occluded players are plotted in 2D, and visually inspected as depicted in Figure 4. In the occlusion case shown in Figure 4, the trajectories of the occluded and occluding players exhibit sharp turns. Other occlusion scenarios may result in the occluded player’s ID disappearing and a new ID appearing for the same player after the occlusion ends.



Figure 4: (Left): Occluded Player with ID = 292. (Right): Player 292 Trajectory with a sharp turn at the blue point which show the time of occlusion.

To create a structured data frame for our trajectory correction algorithm, we split each occlusion case into two records: one for the occluded player and one for the occluding player as illustrated in Table 1. Table 1 includes the start and end timestamps of the occlusion case, the ID of the occluded/occluding player (before occlusion), and the coordinates (X and Y) of the bounding box that indicates a detected object (i.e., player). The range of upper and lower bounds in both X and Y directions are set to be 15% of the width of pitches’ tactical zones (Kim, 2019).

Table 1: Sample occlusion data record.

Timestamp		Player ID	X-axis		Y-axis	
Start	End		Lower Bound	Upper Bound	Lower Bound	Upper Bound
00:02	00:07	17	264	269	201	206

### 3 METHODS

Our proposed player tracking system, FOOTBALLTrace, consists of five main stages: panoramic view creation, player detection, player ID association, occlusion detection, and trajectory correction. The panoramic view creation step is performed at KoraStats (KORASTATS, 2023) where real-time stitching between the three camera views is executed during game filming.–We then apply the object detection algorithm, YOLOv7, to generate a bound box surrounding each detected player in every frame. The player ID association is performed using a Deep Affinity Network (DAN) (Sun, 2019) that accepts two frames as an input to associate objects appearing in both frames.

However, due to frequent interaction among players during the game, occlusion events can occur, leading to inaccurate player ID associations and disrupted trajectories. For instance, when players A and B are in close proximity and occlude each other, their IDs may get swapped (e.g., Player “A” gets assigned the ID “B” and player “B” gets assigned the ID “A”). Our novel algorithm detects occlusion events and utilizes geometrical trajectory analysis to identify sharp trajectory turns, indicating the occurrence of occlusion. Finally, it performs trajectory correction by re-connecting the corrected tracks

#### 3.1 Player Detection

In this study, we utilized one of the state-of-the-art real-time detectors, YOLOv7 (You Only Look Once) (Wang, 2023). YOLOv7 surpasses all known object detectors in terms of both speed and accuracy when applied to videos with a frame rate from 5 to 160 FPS. Additionally, it exhibits the highest accuracy of 56.8% Average Precision (AP) among all known real-time object detectors (Wang, 2023). YOLOv7 was trained by Wang et. al (Wang, 2023) on the widely recognized Microsoft Common Objects in Context (MS COCO) dataset (Lin, 2014). The learned weights

on the MS COCO were used in this study to detect players in game videos with a frame rate of 25 FPS. The output of YOLOv7 consists of bounding boxes for each detected player in each frame, which are employed by the association network to construct continuous trajectories.



Figure 5: Sample of Players' Detection.

### 3.2 Player ID Association

To construct a player trajectory throughout the entire game, we utilized a Deep-Affinity-Network (DAN) (Sun, 2019) to associate the player's ID across consecutive frames. DAN takes two consecutive frames as input and performs object modeling to associate similar objects based to their affinity estimation score; this score is computed using dot product of the estimated feature vectors for detecting bounding boxes as described in (Sun, 2019). The DAN architecture is divided into two deep network components: the feature extractor network and the affinity estimator network.

#### 3.2.1 Feature Extractor Network

The feature extractor network is based on the Visual Geometry Group (VGG) architecture (Simonyan, 2014). It models the appearances of objects (players) by utilizing geometric-based convolutions. VGG extracts appearance features of players at multiple levels of abstraction (i.e., multiple scales). The feature extractor produces a 520-dimensional feature vector for each detected object in each frame. These feature vectors are then concatenated with feature vectors from other frames to form a frame-to-frame feature matrix for player ID association (Sun, 2019).

#### 3.2.2 Affinity Estimator Network

The feature matrices from the two input frames are then fed into the affinity estimator network, which performs exhaustive pairing permutations of features to estimate the similarities and relationships between

players across different frames. The DAN incorporates a specialized loss function for affinity prediction, which ensures a consistent low error in data association (Sun, 2019). The association algorithm is robust to short-term occlusions and capable of accurately tracking players entering and exiting the scene.

### 3.3 Occlusion Detection

In occlusion events, the object detector may fail to distinguish between the two occluded players, resulting in association failure and potentially associating the occluded player ID with the occluding player ID.

False ID associations can be classified into two types; 1) Swapping of IDs between the occlud(ing) and occlud(ed) players, or 2) Assigning new ID for the occluded player as if it were a new player entering the scene. The type of false association during occlusion depends on the duration of the occlusion. The parameter that defines the duration of a long occlusion is denoted as  $\delta_b$ . This parameter is defined in the affinity estimation part of the network, specifying the number of previous frames the network considers during association (Sun, 2019). This means that if the occlusion lasts for a duration longer than  $\delta_b$  (which is set to 15 frames in our study), then the occluded player will be considered as a new player and will be assigned a new ID. If the occlusion lasted less than  $\delta_b$ , then the occluded player either retains the same ID before swapping, or the IDs are swapped between the occluded and the occluding player, and that depends on the networks ability to compute affinities.

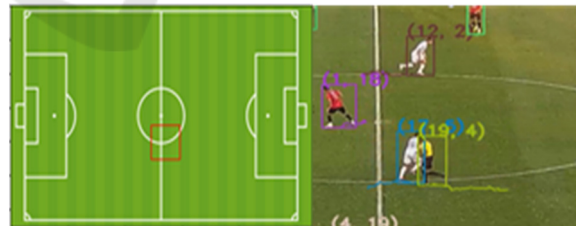


Figure 6: Occlusion suggestion for player ID 19 (green box) and player ID 17 (blue box) with the suggestion bounding box drawn on the pitch zone where occlusion case has occurred.

We have developed an occlusion detection algorithm to aid in identifying possible occlusions during the game. The algorithm utilizes a sliding window of 50 frames (2 seconds) to the algorithm with a 50% overlap between consecutive windows where each window contains the coordinates and IDs

of the detected players. The algorithm then computes a mutual Euclidean distance matrix for players in each frame within the window. A possible occlusion is identified if the distance between two or more players is less than or equal to a proximity threshold ( $\tau$ ). In this work,  $\tau$  is set to 4 for the purpose of reproducibility. Optimization of this parameter is left for future work.

If a possible occlusion occurs for a duration equal to or more than 50% of the window length, the IDs of these two players are marked as possible occlusions. A bounding box is drawn around the pitch zone where the possible occlusion case took place, along with printing the time stamp (in minutes and seconds) corresponding to this case (Figure 6). An operator can refer to the original game camera feed and confirm the possible occlusion before it is input to the trajectory correction algorithm.

### 3.4 Trajectory Correction

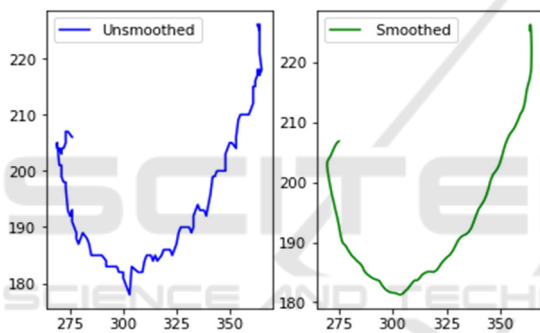


Figure 7: The effect of Savitzky-Golay smoothing filter on a trajectory. (Left): Trajectory before smoothing. (Right): Smoothed trajectory.

We introduce a novel trajectory correction algorithm to re-connect disrupted trajectories in confirmed occlusion events (Algorithm 1). The algorithm utilizes geometrical analysis of trajectories (tracks) within a 5-second window around the occlusion event. However, this step is preceded by a trajectory pre-processing phase aimed at smoothing out the trajectory by eliminating noise in the player’s movement and unwanted trajectory breaks, as depicted in Figure 7.

To smooth out the trajectory, we employed the Savitzky-Golay (SG) filter (Schafer, 2011). The SG filter applies a polynomial of order ( $n$ ) to a subset of trajectory points within a defined window ( $w$ ). Smoothed trajectories are then resampled using a step ( $s$ ). The smoothed and resampled trajectory is used for the trajectory correction step described in Figure 8. We choose a point ( $P$ ) on the trajectory, which is

**Data:** Disrupted trajectories of the players involved in a manually confirmed occlusion case.

**Result:** Corrected trajectories of the players involved in a manually confirmed occlusion case.

**For point  $P(i)$  in the disrupted trajectory array  $T$  do**

calculate  $v1$  and  $v2$  and angle between them

$v1 = P(i) - P(i-1)$ ;

$v2 = P(i) - P(i+1)$ ;

$Angle = \text{dot}(v1, v2) / (\text{norm}(v1) * \text{norm}(v2))$ ;

Append calculated angle to angles array

Find index of minimal angle in Angles array ( $I_{min}$ )

Find point  $P(I_{min})$

Divide Trajectory ( $T$ ) at  $P(I_{min})$

$T\text{-before} = T(0 \text{ to } P(I_{min}))$ ;

$T\text{-after} = T(P(I_{min}) \text{ to } \text{len}(T))$ ;

Connect  $T\text{-before}$  of the occlud(ed) player to  $T\text{-after}$  of the occlud(ing) player

Algorithm 1: Trajectory Correction algorithm.

neither the first nor the last point, is chosen as a pivot point. Two vectors ( $V1$  and  $V2$ ) are then calculated from the pivot point to a point prior to  $P$  on both sides (e.g.,  $P(i-1)$  and  $P(i+1)$  in Figure 8). The angle ( $\theta$ ) between the two vectors  $V1$  and  $V2$  is calculated using the dot product formula in Equation 1:

$$\theta = \frac{V_1 \odot V_2}{\|V_1\| \cdot \|V_2\|} \tag{1}$$

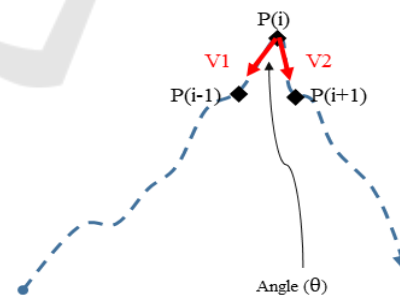


Figure 8: Geometrical Trajectory Analysis. Player moving left to right, where  $P(i)$  is the pivot point on the re-sampled trajectory.

At each point  $P_i$ , the angle  $\theta_i$  is calculated and stored in an array of angles. The index of the minimal angle in the array corresponds to the index of the point  $P_i$  where the breakpoint occurs, marking the moment of occlusion. We then divide the trajectory at this point into two trajectories ( $T_{\text{before}}$  &  $T_{\text{after}}$ ). The

process of finding the breakpoints is applied to the trajectories of both the occluded and occluding players. Trajectory correction is then applied by re-connecting the trajectories such that  $T_{\text{before}}$  of the occluded player is connected to  $T_{\text{after}}$  of the occluding player and vice versa (Figure 9). Corrected trajectories are then assigned to the correct player ID before the occlusion event occurrence.

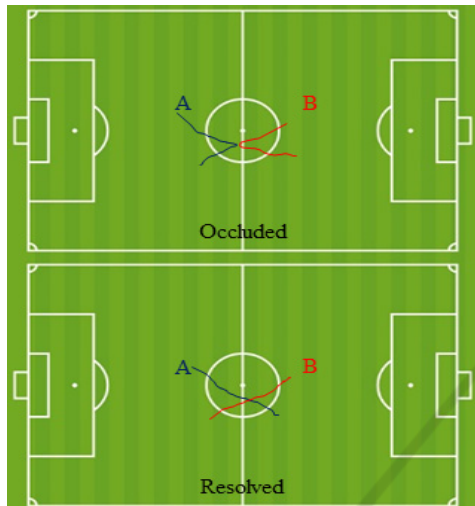


Figure 9: Trajectory Correction Example. (Upper): trajectories of two occluded players before correction. (Lower): corrected trajectories using the proposed trajectory correction algorithm.

### 3.5 Parameters Optimization

The performance of the trajectory correction algorithm depends on three hyperparameters: The Savitzky Golay filter window size ( $w$ ), which determines the number of data points used for curve fitting, the polynomial order ( $n$ ), which specifies the order of the polynomial used to fit the data within the window ( $w$ ), and the step size ( $s$ ), which determines the interval at which the trajectory is resampled for geometrical analysis and vector calculations.

We divided our occlusion dataset into 80% (64 trajectories) for training and 20% (16 trajectories) for testing. To select the best set of parameter values, we performed a grid-search with 5-fold cross validation on the training dataset. Our grid search was limited to window sizes of [5,7,9], polynomial orders of [2,3,4], and step sizes of [5,7,9,10] to expedite the results. It is worth mentioning that those parameters were constrained so that the polynomial order must be smaller than the window size in order to fit the polynomial using the data points within the window.

## 4 RESULTS AND DISCUSSION

The player detection and association steps were executed on an Nvidia GTX 2080 GPU with 8 GB V-RAM. The YOLO object detector was configured to detect a single class, “Person”, and it successfully detected the players in the scenes with 87% accuracy during non-occlusion events as seen in Figure 8. The detection accuracy is calculated by comparing the number of detected persons in non-occlusion events to a reference value of 23 persons, which represents the expected number of individuals on the pitch (22 players+ 1 Referee). However, in cases where there are failed detections in certain frames, these missed detections are often detected in later consecutive frames. Since the DAN does not rely on feeding adjacent frames, it can effectively compensate for these dropped detections during the association process. The average runtime for the detection algorithm is 0.25 second per frame to detect all players.

On the other hand, the association step had an error rate of 20%, which is calculated as number of wrong player ID associations following an occlusion event divided by the total number of occlusion events in the game.

The occlusion detection algorithm detected 211 potential occlusion events in the game, refined to 40 events (80 trajectories) which were manually confirmed. There is a noticeable difference between the number suggested occlusion events and the number of manually confirmed events because the occlusion detection algorithm only considers the Euclidean distance proximity between the detected players. However, this proximity could occur when two players get close for a limited number of frames. In such cases, the DAN is capable enough to overcome the occlusion and successfully associate those players. Meanwhile, the occlusion detection algorithm may still indicate this situation as a possible occlusion due to spatial proximity.

The trajectory correction algorithm successfully corrected 87.5% (70 trajectories) of the 80 disrupted trajectories from the manually confirmed occlusion events. The values of the hyperparameters ( $w$ ,  $n$  and  $s$ ) used for the trajectory correction algorithm were chosen through majority voting among the values that yielded the highest validation accuracy (Table 2), as described in section 3.5. The majority voting process resulted in choosing a window of length 7, a polynomial order of 2, and a step size of 7. The average validation accuracy recorded was 90.5% on the training data.

Table 2: Cross Validation Results.

Fold #	s	w	n	Validation Accuracy
0	7	7	2	0.923
1	9	5	2	0.846
2	7	7	2	1.0
3	7	7	2	0.923
4	9	5	4	0.833

The 10 discontinued trajectories are corrected manually to generate complete and usable low-cost, contactless player tracking data. The complete tracking data can be visualized on 2D maps as shown in Figure 10 and used to plot player movements as in heatmap in the Figure 11.

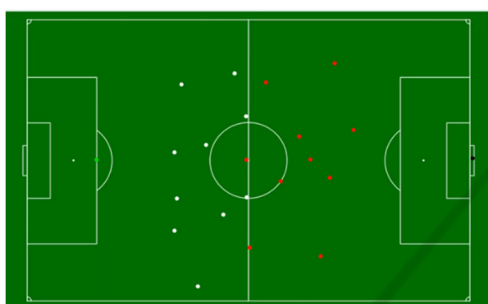


Figure 10: A 2D visualization of the generated tracking data.

Overall, our proposed system is capable of providing a semi-automated tool to collect tracking data through entire game using the recorded match feed. To the best of our knowledge, FOOTBALLTrace is the first AI-based system to produce complete tracking data that can be used in lieu of the GPS modules.

## 5 LIMITATIONS AND FUTURE WORK

Although FOOTBALLTrace is an important step towards the practical application of AI and Computer vision-based tracking systems, manual interventions are still required to generate usable AI-based football tracking data. To develop fully automated and accurate systems, more domain specific training datasets are needed. In this study, we were limited by the lack of publicly available football tracking data. Therefore, we utilized an object association network (i.e., DAN) that is trained on pedestrian tracking data. It is worth noting that pedestrian tracking data exhibit less similarity between tracked objects, whereas football tracking data exhibit a higher degree of similarity between players, leading to more unresolved occlusion events.

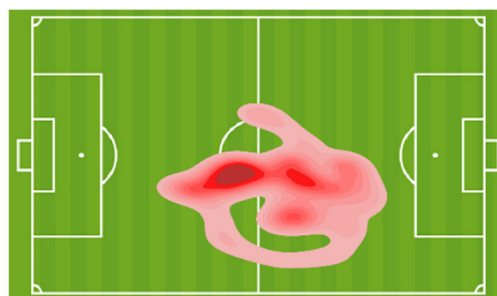


Figure 11: An example player movement over a specific game segment of 5 min.

As a future solution, we plan to use FOOTBALLTrace with minimal human interventions to construct labeled football tracking data and utilize transfer learning to improve the performance of the DAN while reducing the number of human interventions. Furthermore, we intend to investigate the impact of increasing the track memory parameter  $\delta_b$  on association performance during long-term occlusions. Additionally, we will explore optimization techniques such as grid search, coupled with other types of distance metrics (i.e. Manhalobi's distance), to identify the optimal proximity metric and threshold ( $\mathbb{T}$ ) for improving performance of the occlusion detection algorithm.

Finally, we aim at reducing the runtime of our system to generate a real-time tracking data. Currently, the runtime of FOOTBALLTrace on a full game (90 minutes) is approximately one day to generate complete tracking trajectories. While real-time tracking data collection may not be essential for tactical and performance analysis, we plan to enhance the runtime by implementing the entire system using C++ and efficient GPU programming.

## ACKNOWLEDGEMENTS

We would like to express our gratitude to Salma Hazem, Doaa El Sherif, and Mohamed El-Badry at KoraStats, Egypt, for providing us with the necessary facilities and access to the video recordings of football games, and for their efforts in implementing the image stitching algorithm to create the panoramic view data.

## REFERENCES

(2023, June). Retrieved from Tagg Maker: <https://tagg-maker.com/home/>

- Andriluka, M. R. (2008). People-tracking-by-detection and people-detection-by-tracking. *IEEE Conf. on Computer vision and pattern recognition* (pp. 1-8). IEEE.
- Arnason, A., Sigurdsson, S., Gudmundsson, A., Holme, I., Engebretsen, L., & Bahr. (2004). Physical fitness injuries, and team performance in soccer. *Medicine & Science in Sports & Exercise*, 278-285.
- De Silva, V. C. (2018). Player tracking data analytics as a tool for physical performance management in football: A case study from Chelsea Football Club Academy. *Sports*. MDPI -Sports.
- Dejonghe, T. A. (2007). The popularity of football games in the world. Is there a relation with hegemonic power? *Van Christaller Tot Wallerstein*, 39-50.
- Edgecomb, S. J. (2006, May). Comparison of global positioning and computer-based tracking systems for measuring player movement distance during Australian football. *Journal of science and Medicine in Sport*, 9(1-2), pp. 25-32.
- Forcher, L. A. (2022). The use of player tracking data to analyze defensive play in professional soccer-A scoping review. *International Journal of Sports Science & Coaching*, 1567-1592.
- Kim, J. J. (2019, October 16). The attacking process in football: a taxonomy for classifying how teams create goal scoring opportunities using a case study of crystal Palace FC. *Frontiers in psychology*, 10(2202).
- KORASTATS. (2023, July). Retrieved from KORASTATS: <https://korastats.com/site/>
- Lin, T. Y. (2014). Microsoft coco: Common objects in context.. *Computer Vision—ECCV 2014: 13th European Conference Proceedings* (pp. 740-755). Zurich, Switzerland: Springer.
- Luo, W. X. (2021, April). Multiple object tracking: A literature review. *Artificial Intelligence*, p. 293.
- Once- Video Analysis. (2023, June). Retrieved from <https://once.de/>
- Pifer, N. D. (2019). Data analytics in football: Positional data collection, modeling, and analysis. *Journal of Sport Management*, 574.
- Premier League Clubs Revenue By Stream. (2023, June). Retrieved from Statista: <https://www.statista.com/statistics/556893/premier-league-clubs-revenue-by-stream/>
- Rein, R. &. (2016, August 24). Big data and tactical analysis in elite soccer: Future challenges and opportunities for sports science. *Springerplus*.
- Sabirin, H. S. (2015, August 1). Automatic soccer player tracking in single camera with robust occlusion handling using attribute matching. *IEICE TRANSACTIONS on Information and Systems*, 98(8), pp. 1580-1588.
- Schafer, R. W.-1. (2011, June). What is a Savitzky-Golay filter? *IEEE Signal processing magazine*, 28(4), pp. 111-117.
- Simonyan, K. &. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint* .
- Sun, S. A. (2019). Deep affinity network for multiple object tracking. *IEEE transactions on pattern analysis and machine intelligence*, pp. 104-119.
- Szeliski, R. (2007, January 2). Image alignment and stitching: A tutorial. *Foundations and Trends in Computer Graphics and Vision*, 2(1), pp. 1-104.
- Thomas, G. G. (2017). Computer vision for sports: Current applications and research topics. *Computer Vision and Image Understanding*, 3-18.
- Tierney, P. &. (2019, April 24). A Comparison of a Smartphone App with Other GPS Tracking Type Devices Employed in Football. *Exercise Medicine*.
- Wang, C. Y. (2023). YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 7464-7475). IEEE.
- Zhang, J. J. (2018). The sport industry in growing economies: critical issues and challenges . *International Journal of Sports Marketing and Sponsorship*.
- Zhao, Z. Q. (2019, January 27). Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems*, 30(11), pp. 3212-3232.