Scientia Iranica D (2015) 22(3), 1031–1044

Sharif University of Technology
Scientia Iranica
Transactions D: Computer Science & Engineering and Electrical Engineering
www.scientiairanica.com

Discrete particle swarm optimization for player trajectory extraction in soccer broadcast videos

M. Manafifard[a,*], H. Ebadi[a] and H. Abrishami Moghaddam[b]

a. *Department of Photogrammetry and Remote Sensing, K.N. Toosi University of Technology, Tehran, Iran.*
b. *Department of Electrical Engineering, K.N. Toosi University of Technology, Tehran, Iran.*

Received 15 July 2012; received in revised form 12 March 2014; accepted 2 September 2014

KEYWORDS
Track;
PSO;
Adaboost;
Soccer;
Video.

Abstract. Broadcasted soccer videos demand automatic semantic representation for event and tactic analyses. Player tracking is an important step which can be further processed and analyzed by sport experts to evaluate player or team performance. A novel scheme for player tracking in soccer broadcast videos is proposed in this research. Following player detection using Adaboost, player labeling, occlusion handling and mosaic construction, player tracking is formulated as an optimization problem allowing player trajectories to be extracted using Particle Swarm Optimization (PSO). PSO is an optimization method inspired by the flocking behavior of birds, which was originally customized for continuous function value optimization. In this paper, a new application of discrete particle swarm optimization for player tracking in soccer videos is proposed. Updating equations for the particle swarm optimization algorithm are modified based on problem characteristics and discrete variables to extract the player trajectory. Experimental results show that the modified PSO is promising in solving soccer player tracking problems.

1. Introduction

Semantic analysis aims at detecting and extracting information that describes facts in a video. An approach for automatic semantic annotation of soccer videos was presented by Assfalg et al. [1]. The annotation engine detected the principal highlights of the soccer by checking a limited number of observed visual cues against highlight models. A review of state-of-the-art soccer video analysis systems, including player segmentation and tracking, was addressed by D'Orazio et al. [2]. Moreover, a tactic analysis approach for attack events in a broadcasted soccer video was proposed by Zhu et al. [3]. In this direction, an adaptive Gaussian mixture model was used in playfield detection, and a filtering based tracker, called a Support Vector Regression (SVR) particle filter, kept track of the players in the frames. During the tracking process, the color histogram of the target region was employed to identify the team affiliation of the tracked player. The feasibility of a real-time visual system for offside detection in soccer sequences taken from six fixed cameras was investigated by D'Orazio et al. [4]. A background subtraction algorithm allowed the detection of moving areas. Then, player team classification was carried out using an unsupervised clustering algorithm, and a tracking algorithm followed the players. To eliminate the occlusion problem, a linear model was employed to predict the new state configuration, and two or more blobs, whose predicted positions fell close in the image, were treated as occluding. A splitting procedure evaluated the occluding blob and searched for sub-regions considering color features and predicted states. However, the proposed scheme was not applied for broadcast images.

*. *Corresponding author.*
E-mail addresses: mmanafifard@mail.kntu.ac.ir (M. Manafifard); ebadi@kntu.ac.ir (H. Ebadi); moghadam@eetd.kntu.ac.ir (H. Abrishami Moghaddam).

Playfield extraction and player detection are important steps in many applications including tracking and highlight detection. The playfield is extracted by mainly two approaches in most applications: parametric and nonparametric. The nonparametric methods are widely used, based on histograms, which usually exploit a dominant color in the HSV color space [5] due to insensitivity to illumination changes. On the other hand, in parametric methods, usually, a Gaussian Mixture Model (GMM) is learnt to represent the playfield models [6]. Following playfield segmentation, player detection is performed more precisely. Color information, shape, edge, size and other properties were usually used to improve the player detection performance [7-10]. In the case of static cameras, difference images [11] and background subtraction [12] are other approaches to retrieve players.

Recently, some methods, such as template matching [13], Kalman filters [14], snake [15], graph representation [16], particle filtering [17], Markov chain Monte Carlo [5], Multiple Hypothesis Tracker (MHT) [18], meanshift associated with color information [19], and temporal spatio velocity [20], have been presented to deal with the player tracking problem. A dual-mode, two-way Bayesian inference approach was also presented by Xing et al. [21] that switches between single isolated player tracking and multiple occluded player tracking. Kalman filtering, incorporated with contextual knowledge and multiple cues, was also applied to track skaters [22]. It has recently been shown by Berclaz et al. [23] that multi-object tracking can be formulated as a global optimization problem, which could be efficiently solved using the K-Shortest Path algorithm (KSP) belonging to those classes that work on graphs of all potential locations over time. Its main limitation was that, by completely ignoring appearance, it could not identity switches when people came close to each other. A modified version of the KSP that includes appearance information from frame to frame for tracking multiple people problems was addressed by Shitrit et al. [24]. To sum up, most previous approaches to player tracking in the literature can be divided into two categories:

1. The first group of player tracking algorithms are mainly based on a player detection step in which successfully detected players between consecutive frames are associated (e.g., graph based tracking).
2. Another category of player tracking algorithms takes advantage of searching around the previously tracked player in the next frame with an arbitrary detection step (e.g., particle filter, meanshift).

Most applied tracking approaches are not totally ideal, having both advantages and disadvantages. The limitation of snakes was their sensitivity to parameters, contour initialization, occlusion or non-smooth shape varying processes. Although a graph provides a beneficial tool for occlusion resolution, the number of look forward or backward frames should be increased in case of long term occlusions, which imposes high computational load and prevents its real time application. The MHT algorithm also led to large computation and memory resources which could be challenging for real time applications in some cases. Moreover, nonlinear and unpredictable player movements could be problematic for the Kalman tracker due to its linear system model with Gaussian noise. Although the particle filter has been applied to deal with nonlinear player tracking scenarios, its main drawback was its dependence on the number of particles. Another limitation of these trackers lies in tracking multiple teammates close to each other, since all the particles might migrate to one of the modes and ignore all other modes. An obvious advantage of the popular meanshift approach over standard template matching was in avoiding a brute force search and computing the player position in a smaller number of iterations. However, meanshift required that a portion of the players be inside the initial search window and it might fail when fast moving small players are totally outside the initial window. Moreover, meanshift was sensitive to its initial placement and might fail when two teammates are moving very close to each other, since only color information is used. Incorporating motion information could also be violated in the case of unpredictable nonlinear player motion. As the template matching and meanshift system does not inherently consider occlusions and changes in player posture, the estimated position precision might be reduced, e.g. when a player falls down. Camshift is also an extension of the meanshift for size adaptation. Although the color of player uniforms has been widely used by most previous player detection and tracking approaches (e.g., template matching, meanshift, camshift and color based PF), distinguishing distant and blurred players using just color information is an uncertain task. The low resolution of pixels blurs the colors and the changes in lighting conditions complicate the reliable use of color information. Moreover, tracking players with colors similar to the background are main problems of color based approaches. Although the decision about player tracks is straightforward for widely separated players, it is difficult for closely spaced players, since there might be multiple possible solutions for each player. In addition, difficult situations arise for teammates in similar clothes. There is also a lot of clutter due to the jitter of the cameras and the lack of pixel resolution on players (Figure 1), which makes the occlusion resolution task more difficult.

This paper concentrates on player trajectory extraction based on Particle Swarm Optimization (PSO).

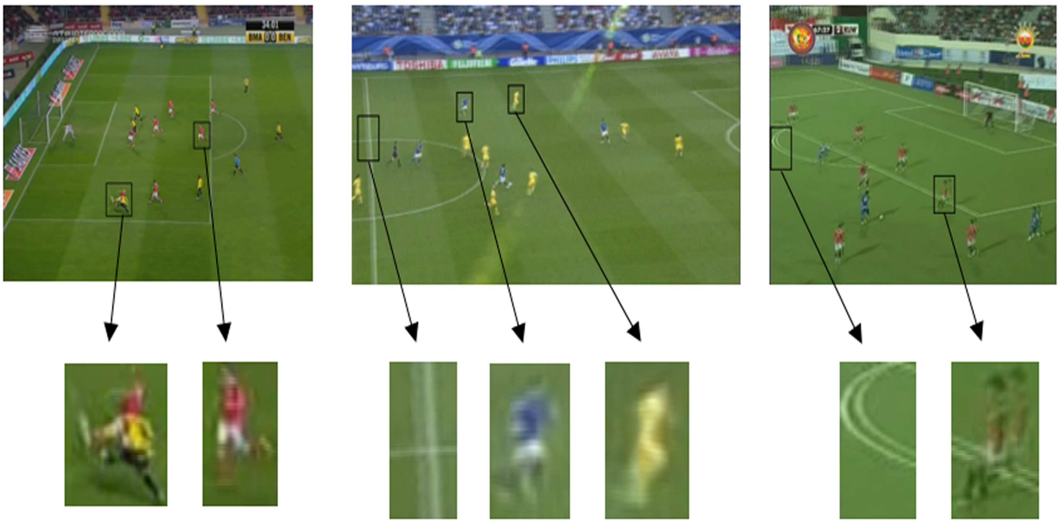

Figure 1. Scenes with blurred and duplicate players and lines.

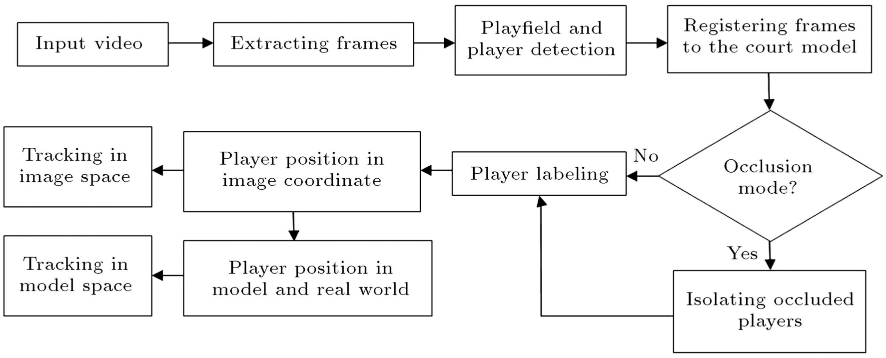

Figure 2. The architecture of our system.

PSO is a population-based search algorithm based on simulation of the social behavior of bird flocking and fish schooling, so, particles fly in the problem search space to find optimal solutions. The original version of optimizing continuous nonlinear functions was developed by Kennedy [25]. Later, Kennedy et al. [26] proposed a discrete version of binary PSO, defining particle trajectories and velocities in terms of changes of probabilities. Since 1997, some researchers have tried to extend PSO to discrete areas, such as solving the Travelling Salesman Problem (TSP). In this manner, a discrete PSO algorithm was used for TSP by Wang et al. [27]. The concept of swap operator and swap sequence was also addressed in their research. Focused on the TSP problems, a novel hybrid discrete PSO algorithm has been presented, by adding a heuristic factor, a crossover operator and an adaptive disturbance factor into the approach by Fan [28]. Furthermore, a modified Genetic Particle Swarm Optimization (GPSO) for TSP was addressed by Cheng et al. [29] in which both the genetic algorithm and PSO mechanisms were incorporated. In comparison with original PSO, this method has provided a much better performance. A hybrid algorithm which integrates PSO with a Simulated Annealing (SA) algorithm for TSP was also proposed by Fang et al. [30]. Thus, by integrating SA with the PSO, the new algorithm could escape from the local minimum trap. Furthermore, a novel polygonal approximation approach based on PSO and genetic reproduction mechanisms, namely crossover and mutation, to deal with combinatorial optimization, was proposed by Yin [31].

A unified framework for multi-player tracking in soccer broadcast videos is proposed in this paper (Figure 2). Since 1995, PSO has been proven to succeed in continuous problems. However, few reports can be found on its application to discrete problems. In this paper, an extended discrete PSO algorithm is proposed for player tracking in soccer videos. In order to gather basic information, including player positions, they are detected using Adaboost. Following image registration and mosaic construction, players are labeled and occluding players are isolated using color information. Finally, the estimated positions are plotted on a constructed mosaic to extract player trajectory using the proposed modified PSO. Since players are often motion-blurred, due to fast camera motion, their feet are missed in many cases, which reduces the precision of localization. Missing feet also lead to several meters error in player localization, which shows the importance of player detection without missing feet. In most research, the precision of player localization is not calculated in a real coordinate system, which requires image to model registration. In this research, this issue is considered in order to make our results comparable with different localization methods. Moreover, isolating occluding players is another crucial factor influencing player localization precision up to several meters. In our paper, the isolation of occluding players is considered in order to overcome the drift in localization. Compared to the scheme proposed by D'Orazio et al. [4], our approach is implemented on a broadcast video (monocular case) and a new combination of neighborhood graph, number of parents and children for each node in the graph, size of non-occluded neighboring bounding box, blob dimension and color features are employed to predict and split occluding blobs. Also, player localization in occluding blobs is improved considering the size of non-occluded neighboring bounding boxes. The proposed approach can cope with missed player feet and extra regions (grass, lines etc.) merged with players using color information. The novelty of our proposed approach relies mainly on applying PSO for player tracking in the far-view shots. The proposed PSO based player tracking is an extension of the detection based tracking methods. Nodes in the graph represents the segmented player region, while each edge represents the distance between nodes by linking players in consecutive frames, assuming that player positions in any two adjacent frames are quite close. The tracking of each player is usually performed by searching the optimal path in the graph. In this direction, a minimal path search using the distance between the blobs was applied by Figueroa et al. [11] to track isolated players, since there was only one edge at each step to be considered. Although the decision about the assignments is straightforward for widely separated players, it is difficult for closely spaced players, since there might be multiple possible edges for each player. The main advantage of the proposed approach, over tracking by searching the shortest path for each player, is inserting the association into the graph based approaches for solving ambiguities in case of closely spaced players (i.e., multiple possible edges for each player). For this purpose, edges are simultaneously assigned to all players in several frames and the fitness value is defined based on all tracks which require an appropriate search method for extracting trajectories. In fact, deciding about all trajectories at a time step could be postponed to the later time step at which subsequent frames could help resolve uncertainty for true trajectory estimation. In an exhaustive brute search, all edges should be explored for the players. Therefore, the search space required to adequately track all players might be huge according to the different combination of edges, which also increases as the number of potential edges grows. A possible solution to this computationally expensive problem would be performing a partial exploration of the image by the proposed PSO and yet achieving a satisfactory tracking result. Therefore PSO is an appropriate solution to the simultaneous tracking of all players in a big solution space, due to the large number of combinations for edges. In this regard, a neighborhood graph with nodes representing detection is built for multi-player tracking, and the specific structure of the graph is exploited to reach the optimum using the PSO algorithm. Moreover, two moves, namely, switch move and changing an edge, are proposed to generate suggestions for new solutions based on matrix representation and a neighborhood graph. The state of a missing player between successful frames is also recovered by linear interpolation.

The rest of this paper is organized as follows. Section 2 describes field segmentation and player detection. Registration and mosaic construction is described in Section 3. Section 4 is devoted to the labeling and occlusion problems. In Section 5, player trajectories are extracted using modified PSO. The experimental results and analyses are given in Section 6, and conclusions are drawn in the last section.

2. Playfield extraction and player detection

The playfield is the dominant background in soccer video frames, and the green color of grass in the field varies under different conditions, such as light, air conditions and shadows. GMM associated with morphological operations is applied to playfield segmentation [6] and the field convex hull is computed to retrieve the binary mask of the field. Meanwhile, one step has to be taken to remove extra parts near the field border. Therefore, the convex hull is eroded by a few pixels. Following playfield detection, Adaboost is applied to detect players. The Adaboost method for frontal view face detection was proposed by Viola et al. [32], and can learn a strong classifier based on a large set of weak classifiers by reweighting the training samples. It is also a strategy for learning classifiers by combining simpler ones. Moreover, a cascade technique [32] is applied here to handle the huge number of sub-images. Since, in typical soccer frames, most sub-images are non-player patterns, the cascade technique attempts to discard these windows as much as possible. Every stage of the cascade is trained to detect almost all objects (players in our example), while rejecting just a fraction of the background. So, every position of the image that is not rejected is given to the next classifier stage.

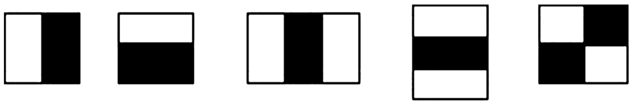

Figure 3. Haar wavelet features.

Three stages are trained to detect players, while each stage is trained with the Gentle Adaboost algorithm, the Haar feature set (i.e., features of Figure 3) and stumps as the weak classifiers. One feature from our feature pool, in combination with a simple binary thresholding decision, is used by the weak classifier. Finally, the strong classifier is constructed by a weighted combination of weak classifiers. In our case, 703 positive patterns and 712 negative patterns, two thirds as the training set and one third as the validation set are used to train the cascade. Moreover, the classifier may be improved by spending a lot more time and using larger training sets. During detection, sub-windows, typically sized between 32 $\times$ 32, are passed through the cascade, and sub-windows passing through all the three stages are classified as players. Since multiple players are detected at nearby locations, multiple nearby detection results are merged. The playfield and player detection results are shown in Figure 4.

3. Registration

To estimate real world player positions on the ground, it is required to compensate for camera motion (pan, zoom, tilt and rotations) by applying planar homography, which is a 3×3 matrix applied to coordinates in a homogeneous form [22]. A homography is computed by 4 matched points or lines in the image plane (x, y) and field plane (X, Y), but more than 4 points are usually used for registration. Accordingly, the field model is constructed considering FIFA regulations concerning court dimensions at a ground resolution of 0.5 meters/pixel. There are also some characteristic points (penalty points, line intersections with each other or circles and the end of two poles) on the field usually used for matching. For the images without enough corresponding points, reference frames are used to solve this problem by registering those low textured images to reference images, as more matched points can be found between frames. In order to avoid accumulating errors, several reference frames are chosen. In this manner, the rich textured frames are selected as the reference frames. Finally, the mosaic is registered to the model and the resulting mosaic is used for player tracking in the image space.

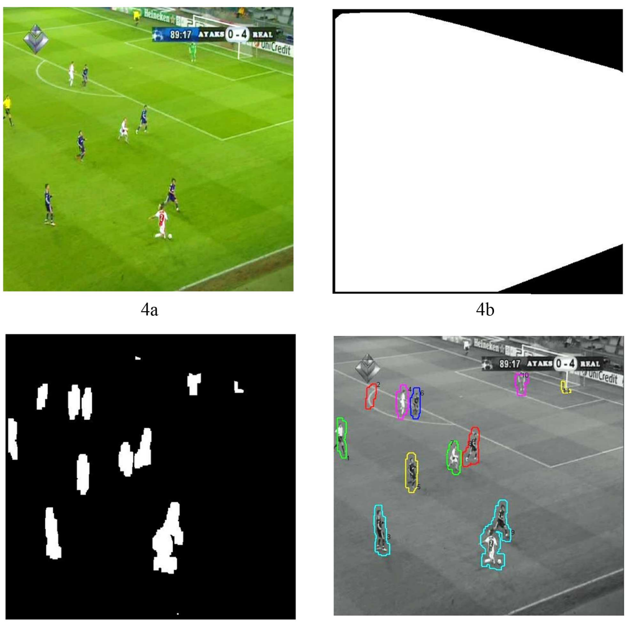

Figure 4. (a) Original frame. (b) Playfield segmentation. (c) Player segmentation. (d) Players in the original frame.

4. Solving occlusions and labeling players

The main difficulty that the player detection procedure has to face is dealing with occlusions, which remove player-position one to one correspondence. Since players shirts often appear partially in the occluding blob, shirt color is the most efficient feature to deal with occlusions caused by competitors. Moreover, a predefined color model is beneficial to most soccer broadcast videos, as teams use their official uniforms in different matches. Discriminative color selection is employed here to model soccer players and each pixel in a blob is classified into classes considering each team and referee. In cases where one team color is present in the player bounding box, it is labeled, considering the color. Otherwise, the occlusion is handled. The graph modeling is used as the initial trajectory for occlusion handling, where the temporal and spatial relationship between the successive trajectory segments is considered (Section 5). Each node of the graph represents one player location at a given time instant, and edges between neighboring nodes represent motion between locations. The neighboring players in the previous and next framed are the parent and the child of the player in the current frame, respectively. Afterwards, occluding blobs, including competitors, are considered, while the blob maintains the information about different colors identifying each team. Accordingly, the splitting procedure evaluates the occluding blob and searches for regions having the same color features. Each single blob related to the player's clothes is enlarged considering the size of non-occluded neighboring bounding boxes. Occlusion between teammates is also detected by comparing the bounding box size of the player to the average bounding

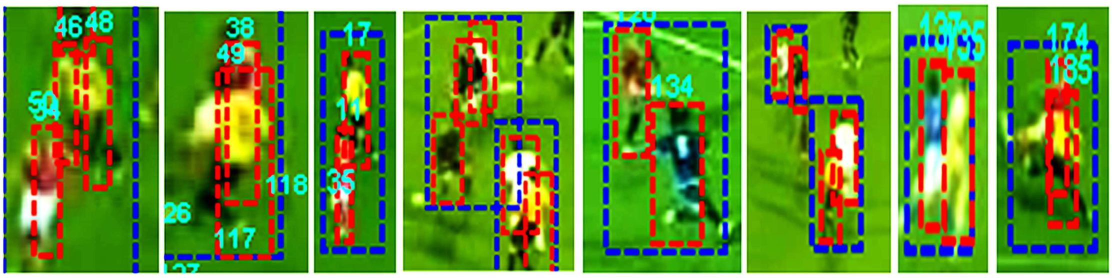

Figure 5. Separating players (red bounding boxes) in occluding blobs (blue bounding box).

box size of all players and neighboring players, as follows:

$$\text{Height}(i) > \frac{1}{N}\sum_{i=1}^{N}\text{Height}(i) + 3\times\sigma(\text{Height}),$$

$$\text{Area}(i) > \frac{1}{N}\sum_{i=1}^{N}\text{Area}(i) + 3\times\sigma(\text{Area}),$$

$$\text{Height}(i) > \frac{1}{n}\sum_{i\in n}\text{Height}(i) + 1.5\times\sigma_n(\text{Height}), \quad (1)$$

where Height(i) and Area(i) denote the height and area of the ith player, respectively. The standard deviation of all players and neighboring players are also represented by σ and σ_n. N and n also denote the number of all players and neighboring players, respectively. So, occluding blobs are detected, but a splitting procedure can be difficult, especially when two or more teammates are very close to each other. Our system deals with occlusion among multiple players and two teammates (Figure 5) using color information and the size of the nearest non-occluded adjacent bounding box. According to the number of player clothes regions in the occluding blob, two situations may occur: In the first case, the occluding blob is split by considering one piece of the player's clothes (e.g., shirt) and each single blob related to the players' clothes is enlarged considering the size of non-occluded neighboring bounding boxes. In the second case, morphological operation is first applied to split occluding blob (distinguished by Eq. (1)) with more than one parent and one child. They are only applied when 2 split regions, by more than 90 percent overlap with occluding blob, are present, after applying morphological operation. As long as doing so cannot split all occluding regions, the occluding blob is split into two from the upper and lower pixels, considering the bounding box size of neighboring players. Moreover, the occluding blob (distinguished by Eq. (1)) with just one parent and one child, referring to player segmentation error, is modified using color information. Illegal player size is caused by playfield pixels merging with the player blob during player detection. Inequality is also applied to detect missed feet during segmentation. In cases where the inequality is true, the player bounding box is recovered, considering the bounding box size of neighboring players and color features, in order to compensate for the player's feet:

$$\text{Height}(i) < \frac{1}{N}\sum_{i=1}^{N}\text{Height}\ (i) - 3\times\sigma(\text{Height}),$$

$$\text{Area}(i) < \frac{1}{N}\sum_{i=1}^{N}\text{Area}(i) - 3\times\sigma(\text{Area}). \quad (2)$$

Furthermore, occluding blobs including more than two teammates (usually occurring when play is stopped) are split by considering one piece of the players' clothes (e.g., shirt) and each single blob related to the players' clothes is enlarged considering the size of non-occluded neighboring bounding boxes. However, this method fails when total occlusion occurs between more than two teammates. This rarely happens during the match because players chase opponents due to the nature of the game and the opponent players usually stay between teammates. So, the decision is taken for each blob, considering dimensions, color features and number of parents and children. At the end of this step, the state of each segmented blob is maintained by updating its position in the occluding blob.

The above approach can cope with missed player feet and extra regions (grass, lines etc.) merged with players using color information, and non-labeled blobs corresponding to blurred players and false alarms (line, grass etc.) are resolved in the tracking stage. Occlusion handling and player labeling are also solved simultaneously. However, comprehensive research on suitable color features is crucial to represent team uniforms.

5. Player tracking

To track players, the center bottom pixel of the bounding boxes are plotted, as the feet positions in the same coordinate system, and players are tracked by a tracking algorithm in the camera plane or the field plane. In order to initialize the tracking process, the

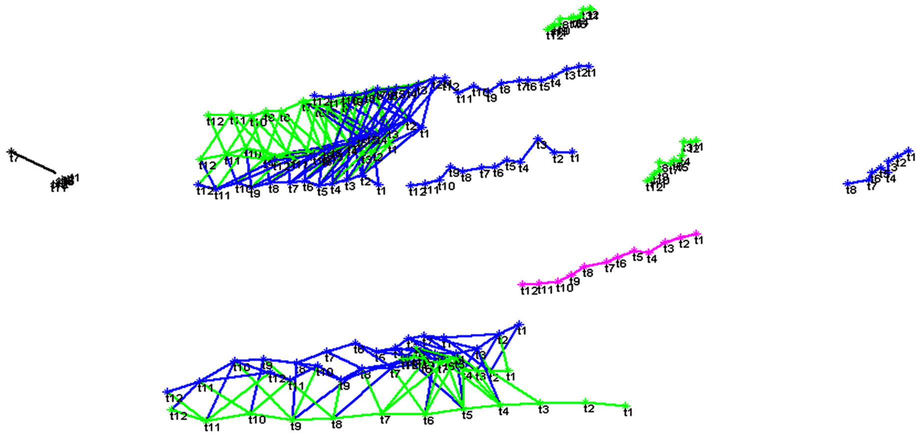

Figure 6. Neighborhood graph. The dots indicate estimated positions of the players and their colors indicate team A (blue), team B (green), goal keeper (black) and referee (pink).

neighborhood (Figure 6) between players is defined as follows:

$$\begin{aligned} N =&\{|Y^i_{t_i} - Y^j_{t_j}| \leq |t_i - t_j| \times V_v^{\max} \\ &\text{and} \quad |X^i_{t_i} - X^j_{t_j}| \leq |t_i - t_j| \times V_h^{\max} \\ &\text{and} \quad |t_i - t_j| \leq \Delta t^{\max}\}, \end{aligned} \tag{3}$$

where $X^i_{t_i}$ and $Y^i_{t_i}$ represent the position of the ith player in the t_ith frame, and $V_v^{\max}$, $V_h^{\max}$ are the maximum speed of the player in vertical and horizontal directions, respectively. $\Delta t^{\max}$ indicates the time interval between consecutive frames. Furthermore, the maximum displacement constraint is assessed in both image and field model space. The neighboring players in the previous and next frames are also the parent and child of the player in the current frame, respectively. Finally, the PSO optimization is applied to the multi-player tracking problem in soccer videos.

5.1. *Particle Swarm Optimization (PSO) algorithm*

PSO is a population based stochastic optimization technique developed by Eberhart and Kennedy [25]. It is inspired by the social behavior of bird flocking or fish schooling. Suppose there are a group of birds randomly searching for food in an area, and there is only one piece of food in the area being searched. They do not know its location, but they know how far it is in each iteration. Each bird, called a particle, benefits from his own and other members of the swarm's experiences during the search for food. So, each bird flies through the space by following the nearest bird to the food. The PSO algorithm applies the concept of social interaction to problem solving, which contains a swarm of particles, each of which includes a potential solution. Each particle keeps track of its coordinates and flies through a multidimensional search space. Thus, the position of each particle is adjusted according to its own experience (local search), as well as the experience of other particles (global search). All particles also have fitness values, which are evaluated by the fitness function to be optimized. The best position each particle has ever achieved is called P_{id}. All the particles can also share their information about the search space, so, there is a global best solution, called P_{gd}. Moreover, a velocity is assigned to each particle in order to direct the flight through the problem's solution space. In each time step, calculation of the particle velocity through each dimension, d, is mathematically modeled, according to the following equation:

$$V_{id}^{\text{new}} = w_i V_{id}^{\text{old}} + c_1 r_1 (P_{id} - x_{id}) + c_2 r_2 (P_{gd} - x_{id}), \tag{4}$$

where c_1 and c_2 are acceleration constants, and r_1 and r_2 are random values drawn from $U(0,1)$. P_{id} is the best solution this particle has reached and P_{gd} is the global best solution of all the particles. V_{id}^{old} and w_i are the velocity of particle in the previous time step and the weighting function, respectively, and x_{id} is the current position of the particle. The following weighting function is utilized in this application:

$$w = w_{\max} - [(w_{\max} - w_{\min}) \times \text{iter}] / \max\_\text{iteration}, \tag{5}$$

where $w_{\max}$ and $w_{\min}$ are final weight and initial weight, set to be 0.01 and 0.5, respectively. *max_iteration* is the maximum iteration number and *iter* is the current iteration number. In this manner, the velocity value is used to update the particle's position in the next iteration by the following formula in the original version:

$$x_{id} = x_{id} + V_{id}. \tag{6}$$

The original PSO algorithm was customized for continuous optimization problems which exhibited effectiveness and robustness in many applications. In order to deal with discrete optimization problems, Kennedy and Eberhart developed a discrete binary version of PSO in 1997. In the binary version, trajectories are changes in the probability that a coordinate will take on a zero or one value [26].

5.2. Modified discrete PSO algorithm for player tracking

In this section, the basic PSO will be modified for extracting the best trajectories of players in soccer broadcast videos. The details are described in the following sections.

5.2.1. Position matrix (P)

First, solutions are represented by $n \times n$ matrices, where n denotes the number of players plotted on the mosaic. The neighboring players in the previous and next frames are also the parent and child of the player in the current frame, respectively. In this manner, each player is assigned a number and each row represents one player in the position matrix. In each row of the position matrix, value 1 is assigned to the column which is a child of the player. For example, if the ith player is a child of the jth player, it is represented by assigning value one to the jth row and the ith column ($P(j,i) = 1$), as shown in Figure 7.

5.2.2. Initialization of the swarm

The swarm is initialized with randomly generated particles. So, a position matrix is generated for each particle and one child is randomly assigned to each player using a neighborhood graph. In this manner, a population of particles is generated and each of them is represented by a single matrix. The two following properties are considered to initialize the position matrix of each particle:

a) One child is only selected by one player (parent) if possible.

b) For a player with only one child, the child is assigned to that player.

Velocity is also initialized by a $n \times n$ zero matrix for each particle. Afterwards, the fitness is calculated for all particles and the swarm's best position (P_{gd}) is set to be the position matrix with the highest fitness.

	$P1$	$P2$	$P3$	$\cdots$	$\cdots$	Pn
$p1$	0	0	0	1	0	0
$P2$	1	0	0	0	0	0
$P3$	0	0	1	0	0	0
.	0	1	0	0	0	0
.	0	0	0	0	1	0
Pn	0	0	0	0	0	1

Figure 7. Position matrix (Pn denotes nth player).

5.2.3. Velocity updating

In order to extract the trajectories of players, the standard PSO velocity updating equation is modified to reasonably change the position of particles in our application. Given a particle position matrix, x_{id}, and the best local position matrix, p_{id}, the cognition term is defined, based on the subtraction of these two positions, as follows:

$$p_{id} - x_{id} = \begin{bmatrix} 0 & \cdots & 1 \\ \cdots & \cdots & \cdots \\ 1 & \cdots & 0 \end{bmatrix} - \begin{bmatrix} 1 & \cdots & 0 \\ \cdots & \cdots & \cdots \\ 0 & \cdots & 0 \end{bmatrix} = \begin{bmatrix} -1 & \cdots & 1 \\ \cdots & \cdots & \cdots \\ 1 & \cdots & 0 \end{bmatrix}. \quad (7)$$

According to the above equation, -1 represents an uncertain child of the current position. As the probability of changing the child is denoted by the final velocity matrix, values of -1 in the above matrix are replaced with 0.9, and other values are replaced with zero.

$$p_{id} - x_{id} = \begin{bmatrix} 0.9 & \cdots & 0 \\ \cdots & \cdots & \cdots \\ 0 & \cdots & 0 \end{bmatrix}. \quad (8)$$

The same procedure is implemented for calculating the social term, but p_{id} is replaced with p_{gd} in the above equation. The above matrices and the velocity matrix in the previous time step are summed according to Eq. (4), and the updated velocity is estimated. Then, the calculated velocity is limited to [0 1] using the sigmoid function.

5.2.4. Position updating

The position updating equation is modified according to our application. In each row of the position matrix, the cell with value 1 is considered. If its corresponding value in the velocity matrix is the maximum value in the same row, it will be selected as a candidate for position updating. In Figure 8, the cells with red colored values show the position updating candidates.

Following velocity updating and initial solution selection, the updated solutions are proposed using two

0	0	0	1	0	0
1	0	0	0	0	0
0	0	1	0	0	0
0	1	0	0	0	0
0	0	0	0	**1**	0
0	0	0	0	0	1

x_{id}

0.0	0.5	0.0	0.3	0.0	0.0
0.9	0.0	0.4	0.0	0.0	0.0
0.0	0.0	0.2	0.6	0.0	0.0
0.0	0.8	0.0	0.9	0.0	0.0
0.0	0.0	0.0	0.7	**0.8**	0.0
0.0	0.0	0.0	0.0	0.6	0.5

V_{id}

Figure 8. Position matrix and its corresponding velocity matrix to verify position updating candidates (red).

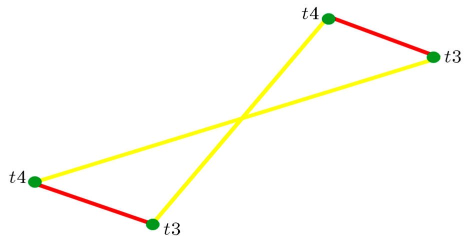

Figure 9. A switch candidate (old edges in yellow) and updated edges by switch move (in red).

types of moves, namely, switch move and changing an edge.

Switch move: If there exist two points, $s1$ and $s2$, such that the child of $s1$ is in the child set of $s2$ and the child of $s2$ is in the child set of $s1$, the pair of nodes will be considered as one candidate for a switch move. Then, one candidate is selected and two new edges are defined by changing their children (Figure 9). The following properties are also considered for switch candidates:

a) The maximum velocity of the candidate's row in the velocity matrix is assigned to the current candidate's child, as shown in Figure 8.

b) The candidates with zero maximum velocities assigned to their children are eliminated from the candidate list.

c) Four candidates whose children are assigned maximum velocities are selected from the candidate list to switch.

d) In cases where the velocity matrix is a zero matrix, one candidate is chosen randomly to switch.

Changing an edge: The sum of each column in the position matrix is calculated and a row vector is produced. If the sum of a column was more than one, the corresponding node would have more than one parent and its parent could be a candidate for this move. Afterwards, one candidate is selected randomly from the candidate list and another child is selected as its child (Figure 10). The following properties are also considered for the candidates of changing an edge:

a) A player with less than two children is removed from the candidate list.

b) The maximum velocity of the candidate's row is assigned to its current child.

5.2.5. Fitness

In each time step, p_{id} and p_{gd} are updated. In this manner, the fitness value of each particle (x_{id}) is estimated and in cases where its value is greater than the fitness value of p_{id}, p_{id} is replaced with x_{id}. Similarly, p_{gd} is replaced with x_{id} when the fitness value

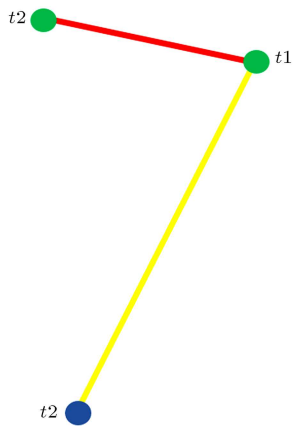

Figure 10. Old solution (yellow edges) and updated solution by changing an edge (red edges).

of x_{id} is greater than the fitness value of p_{gd}. The fitness function is also formulated as follows:

$$\text{Fitness} = \sum_{i=1}^{N} \frac{1}{\text{Dist}_i} \times \text{Label}_i,$$

$$\text{Label}_i = \begin{cases} 1 & \text{for the edge between teammates} \\ \frac{1}{\gamma} & \text{for the edge between players with dissimilar labels} \end{cases} \tag{9}$$

where Dist_i and N represent the distance between two nodes of an edge and the number of edges, respectively. This parameter weights the distance and labeling terms in the fitness definition. If the value is too large, dissimilar labels will be dramatically penalized, which is more appropriate for a precise labeling step; if the value is too small, more edges will be defined between competitors, which is more appropriate for an imprecise labeling step where players might be labeled incorrectly. Since the main goal of this parameter is lessening the fitness value of edges among competitors with dissimilar labels, it must be more than one. In our system, this parameter has been set to 5 empirically and kept the same across all the datasets.

The algorithm in pseudo code is as follows:

```
Initialize the particle swarm
In each time step {
  for each particle (i)
    Calculate fitness value
    if fitness (i) > fitness (p_id) then p_id = i
    if fitness (i) > fitness (p_gd) then p_gd = i
  end
  for each particle
    Update velocity and position
  end
}
```

The process iterates until reaching a stopping condition.

6. Experimental results

The algorithm was tested on 491 frames from 6 different video sequences recorded from regular television broadcasts under different video, weather, ground and lighting conditions involving sunlight, shadows, cloudy weather and low quality videos. The first video, including the Beira-Mar versus Benfica game, was at 25 frames per second, and the image size of this sequence was 704×576. Although the tracking was almost straightforward in the middle field arena, more challenging situations occurred near the goal posts, due to more occluding players. The second sequence was a short clip at 25 frames per second with an image size of 720×576 representing player trajectories during a goal scene from the middle of the playfield to near the goalpost. The image size of this sequence was also 720×576. Besides common occlusions, players were blurred during the sequence. The third video, including the Yemen versus Kuwait game, was at 25 frames per second and the image size of this sequence was 720×576. This data set was particularly challenging due to low quality shadows from a light source at the base of the players, significant image blur and doubled players. The fourth video was Ayaks versus Real at 30 frames per second with an image size of 544×576, and the fifth sequence was selected from the La Gantoise versus FC Malines game at 25 frames per second. The image size was 544×576 and players were blurred in the sequence. The last sequence, including Lyon versus Bayern, was also a low quality blur sequence at 25 frames per second. The image size of this sequence was also 704×576, and occlusions along with blurred players were the main challenges. Moreover, multiple shadows from different light sources in the stadium were visible at the base of the players. The proposed algorithm was also implemented in MATLAB software.

Since player trajectories are extracted using segmentation results, player tracking is affected by results of a player segmentation method. In order to assess the player detector, the player position (the center bottom pixel of the bounding boxes) is manually labeled as the ground truth. Then, the precision is evaluated using frames from 6 different videos by three criteria. In order to evaluate the applied detection method, the player detection is assessed by recall and precision. The precision of player localization is also determined by RMSE. Since players are often motion-blurred due to the fast camera motion, player feet are missed in many cases, which decreases the precision of localization as much as several meters in a real coordinate system. So, RMSE is used to consider the capability of the method to keep a player's feet stuck to his body by calculating precision in a real coordinate system.

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\Delta X_i^2 + \Delta Y_i^2)},$$

$$\mathrm{RMSEX} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\Delta X_i^2},$$

$$\mathrm{RMSEY} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\Delta Y_i^2},$$

$$\mathrm{Recall} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}, \tag{10}$$

$$\mathrm{Precision} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}},$$

where N is the number of players in all frames, ΔX and ΔY are player position deviations from the ground truth in X and Y directions (X and Y are adapted to the length and width of the playfield) and TP, FN, FP denote true positive, false negative and false positive, respectively. During an attack, a player's step could be considered longer than one meter, and the whole space between the feet is assumed as the player position. However, this space is supposed to be less than half a meter for a stationary player, and, therefore, a large deviation from the ground truth would be tolerable. According to this point, the segmentation results are good enough to initialize the tracker, as illustrated in Table 1. As far as the player is further displaced in the y direction by missed feet, more errors appear in the y direction than in the x direction. In some cases, big displacements are caused by missed feet, which usually happens for distant or blur players, and, thus, overall accuracy is decreased. The final results could be also affected by the occlusion

Table 1. Player detection performance and player position error.

Method	Recall (%)	Precision (%)	RMSEX (m)	RMSEY (m)	RMSE (m)
Adaboost	95	88	0.4	0.7	0.8
Khatoonabadi et al.	86.97	89.79	-	-	-
Utsumi et al.	64.9	66.9	-	-	-

handling step. Adaboost has been shown to be a fast detection technique with the drawback of time complex training, where a large training set is required to train the classifier. The more time spent for training, the more precise positions are retrieved. The results of Adaboost are compared to a region based algorithm implemented by Khatoonabadi et al. [13] and detection using color rarity and a local edge property applied by Utsumi et al. [7], as shown in Table 1. However, player localization in a real coordinate system is not assessed in these papers.

The proposed tracking algorithm based on discrete particle swarm optimization was also tested on 7 shots from six different videos. As shown in Figure 11,

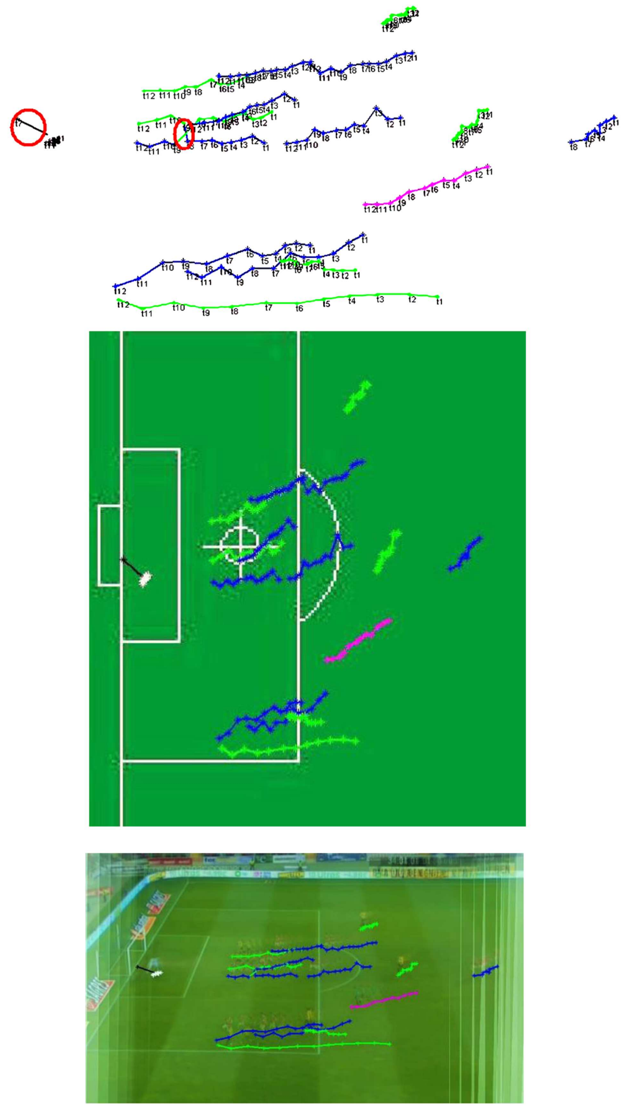

Figure 11. Estimated trajectories: (a) Trajectories in image coordinate system. The red ellipses indicate errors; (b) trajectories in model coordinate system; and (c) trajectories illustrated on constructed mosaic. The dots indicate estimated positions of the players and their colors indicate team A (blue), team B (green), goal keeper (white), referee (pink) and outliers (black).

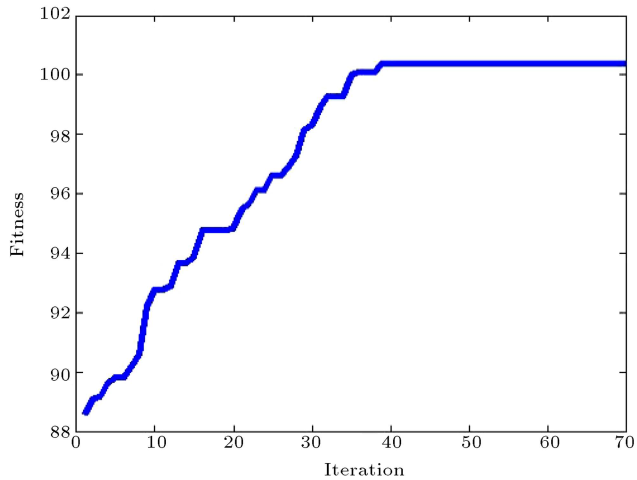

Figure 12. Convergence plot of PSO.

the results could be used by experts and coaches to observe player trajectories and judge the weaknesses and strengths of the players and the team. The false edges could be caused by segmentation error or the trouble of jumping out of good local optima, as shown in Figure 11(a). The player trajectory was also extracted in both the model (Figure 11(b)) and the image (Figure 11(c)) spaces to show the results in more realistic space. The results are evaluated for each shot by running the program 10 times and reporting the average of results. The convergence curve of the following sequence is also presented in Figure 12. As this figure shows, the fitness value is increased during iterations. Besides, the more the number of edges, the more iterations are required to converge.

Tracking is solved as an optimization problem in our proposed scheme. Moreover, a precision of 94.2% and a recall of 96.4% were reached by our tracking, based on PSO, on the ground truth of six clips. As far as different algorithms are implemented on different videos, using different methods (segmentation, occlusion handling etc.) for initialization of the tracker, they could not be compared using only their reported results. Furthermore, occlusion was handled during tracking in some papers [16] or occluded players were not isolated in some others, such as the research carried out by Khatoonabadi et al. [13]. Tracking algorithms were also not evaluated by Beetzet al. [18], or evaluated with different criteria by some authors [16-20], such as the capability of handling occlusions. Moreover, the different steps of each paper could affect the next steps, and the variability in the applied steps makes it hard to perform a comparison of the final step for tracking, influenced by all the previous steps. For instance, player segmentation using the Adaboost classifier requires collecting a large number of training data, and most errors for player tracking were caused by the player detection step. However, the more time spent for training, the more precise positions could be retrieved. To sum up, errors in player tracking were mainly caused by player detection, and tracker performance will be certainly improved by more efficient detection. The proposed scheme is compared to the minimal search approach in the graph, while the previous steps to tracking (playfield detection, player detection and occlusion resolution) are implemented in the same manner to make this comparison meaningful. In a minimal path search, the graph is traversed by considering a minimal path using the distance information between the blobs. However, false trajectories are detected in situations where multi players are moving close to each other, since some nodes in the next frame might be assigned to more than one player in the current frame, or some nodes might be assigned to no player due to non-simultaneous tracking of all players in each frame. Moreover, incorrect trajectories were formed due to false alarms. As shown in Table 2, the proposed PSO scheme outperforms the minimal search approach.

As mentioned earlier, the main goal of the proposed approach is deciding about all trajectories by postponing to the later time step. This simple way of simultaniously estimating all players' trajectories according to multiple frames does increase the accuracy of the estimated tracks, but at the cost of increasing computational time. In an exhaustive brute search, all edges should be explored for the players. Therefore, the search space required to adequately track all players might be huge, according to the different combination of edges, which also increases as the number of potential edges grows. A possible solution to this computationally expensive problem would be performing a partial exploration of the search space by the proposed PSO and yet achieving a satisfactory tracking result. Therefore, PSO is an appropriate solution to the simultaneous tracking of all players in a big solution space due to the large number of combinations for edges. The time taken by an algorithm depends mostly on the number of fitness evaluations or scanned trajectories. Since possible edges for each player are denoted by the number of its children, the total number of scanned trajectories by a brute search procedure is computed by $\prod_{i=1}^{n}\sum_{j=1}^{n}P(i,j)$, where P is the position matrix defined in Section 5.2.1. On the other hand, multiplying the number of particles and iteration steps gives the number of hits where the search space is scanned in the proposed PSO scheme. As shown in Table 3, computational complexity is remarkably increased by increasing the number of look ahead frames, and PSO can significantly reduce scan numbers compared to an exhaustive search, which is a remarkable achievement in computational reduction. This kind of configuration, by increasing the number of look ahead frames, could be particularly beneficial in cases of adding smoothness or other constraints to the fitness function, which is out of scope of this paper.

Table 2. Results obtained from the tracking algorithms.

Method	Recall (%)	Precision (%)
Minimal path search	90.2	88
Proposed approach	96.4	94.2

7. Conclusion

PSO was originally proposed for continuous function value optimization and was developed to solve discrete problems afterwards. However, the need for modifying the discrete version in different applications is inevitable. A modified PSO, which deals with tracking problem, is introduced in this paper. In order to initialize the tracker, player segmentation, occlusion handling and player labeling are applied and estimated player positions are evaluated in a real coordinate system. The player trajectories are also extracted in the image and the model coordinate system. Experimental results show that the PSO algorithm has competitive potential for solving discrete optimization problems. Moreover, the modified particle swarm for tracking applications performs quite well, but had trouble jumping out of good local optima. In future, other optimization techniques, such as simulated annealing, will be applied, which has the capability of escaping from the local minimum trap, and automatic image registration will be studied to accelerate tracking initialization and to evaluate the proposed method on a large dataset.

Table 3. Number of scans for PSO versus brute search.

Method	# Frames	# Nodes	# Iterations	# Particles	# Scan
Exhaustive search	2	33	-	-	2160
PSO	2	33	5	10	50
Exhaustive search	12	202	-	-	9×10^{43}
PSO	12	202	50	200	10000

References

1. Assfalg, J., Bertini, M., Colombo, C., Bimbo, A.D. and Nunziati, W. "Semantic annotation of soccer videos: automatic highlights identification", *Int. J. of Computer Vision and Image Understanding*, **92**(2-3), pp. 285-305 (2003).

2. D'Orazio, T. and Leo, M. "A review of vision-based systems for soccer video analysis", *Int. J. of Pattern Recognition*, **43**(8), pp. 2911-2926 (2010).

3. Zhu, G., Xu, C., Huang, Q., Rui, Y., Jiang, S., Gao, W. and Yao, H. "Event tactic analysis based on broadcast sports video", *Int. J. of IEEE Transactions on Multimedia*, **11**(1), pp. 49-67 (2009).

4. D'Orazio, T., Leo, M., Spagnolo, P., Mazzeo, P.L., Mosca, N., Nitti, M. and Distante, A. "An investigation into the feasibility of real-time soccer offside detection from a multiple camera system", *Int. J. of IEEE Transactions on Circuit and System for Video Technology*, **19**(12), pp. 1804-1818 (2009).

5. Liu, J., Tong, X., Li, W., Wang, T., Zhang, Y. and Wang, H. "Automatic player detection, labeling and tracking in broadcast soccer video", *Int. J. of Pattern Recognition*, **30**(2), pp. 103-113 (2009).

6. Jiang, S., Ye, Q., Gao, W. and Huang, T. "A new method to segment playfield and its applications in match analysis in sports video", *12th Int. Conf. on Multimedia*, New York, USA, pp. 292-295 (2004).

7. Utsumi, O., Miura, K., Ide, I., Sakai, S. and Tanaka, H. "An object detection method for describing soccer games from video", *Int. Conf. on Multimedia and Expo (ICME)*, **1**, pp. 26-29 (2002).

8. Huang, Y., Llach, J. and Bhagavathy, S. "Players and ball detection in soccer videos based on color segmentation and shape analysis", *Int. Conf. on Multimedia Content Analysis and Mining*, **4577**, Weihai, China, pp. 416-425 (2007).

9. Gong, Y., Sin, L.T., Chuan, C.H., Zhang, H. and Sakauchi, M. "Automatic parsing of TV soccer programs", *Int. Conf. on Multimedia Computing and Systems*, Washington, DC, USA, pp. 167-174 (1995).

10. Yamada, A., Shirai, Y. and Miura, J. "Tracking players and a ball in video image sequence and estimating camera parameters for 3d interpretation of soccer games", *Int. Conf. on Pattern Recognition*, **1**, Washington, DC, USA, pp. 303-306 (2002).

11. Figueroa, P.J., Leite, N.J. and Barros, R.M.L. "Tracking soccer players aiming their kinematical motion analysis", *Int. J. of Computer Vision and Image Understanding*, **101**(2), pp. 122-135 (2006).

12. Abbott, R.G. and Williams, L.R. "Multiple target tracking with lazy background subtraction and connected components analysis", *Int. J. of Machine Vision and Applications*, **20**(2), pp. 93-101 (2009).

13. Khatoonabadi, S.H. and Rahmati, M. "Automatic soccer players tracking in goal scenes by camera motion elimination", *Int. J. of Image and Vision Computing*, **27**(4), pp. 469-479 (2009).

14. Ren, J., Xu, M., Orwell, J. and Jones, G.A. "Multi-camera video surveillance for real-time analysis and reconstruction of soccer games", *Int. J. of Machine Vision and Applications*, **21**(6), pp. 855-863 (2010).

15. Lefèvre, S., Fluck, C., Maillard, B. and Vincent, N. "A fast snake-based method to track football players", *Int. Conf. on Machine Vision Applications*, Tokyo, Japan, pp. 501-504 (2000).

16. Figueroa, P., Leite, N., Barros, R.M.L., Cohen, I. and Medioni, G. "Tracking soccer players using the graph representation", *17th Int. Conf. on Pattern Recognition (ICPR)*, **4**, Washington, DC, USA, pp. 787-790 (2004).

17. Davis, M. "Investigation into tracking football players from single viewpoint video sequences", *Bachelor of Science in Computer Science*, The University of Bath, p. 147 (2008).

18. Beetz, M., Gedikli, S., Bandouch, J., Kirchlechner, B., Hoyningen-Huene, N.V. and Perzylo, A. "Visually tracking football games based on TV broadcasts", *20th Int. Conf. on Artificial Intelligence*, Hyderabad, India, pp. 1805-1819 (2007).

19. Chiang, T.K., Leou, J.J. and Lin, C.S. "An improved mean shift algorithm based tracking system for soccer game analysis", *Int. Conf. on Asia-Pacific Signal and Information Processing Association (APSIPA)*, Sapporo, Japan, pp. 4-7 (2009).

20. Sato, K. and Aggarwal, J.K. "Tracking soccer players using broadcast TV images", *Int. Conf. on Advanced Video and Signal Based Surveillance (AVSS)*, pp. 546-551 (2005).

21. Xing, J., Ai, H., Liu, L. and Lao, S. "Multiple player tracking in sports video: A dual-mode two-way Bayesian inference approach with progressive observation modeling", *Int. J. of IEEE Transactions on Image Processing*, **20**(6), pp. 1652-1667 (2011).

22. Liu, G., Tang, X., Cheng, H.D., Huang, J. and Liu, J. "A novel approach for tracking high speed skaters in sports using a panning camera", *Int. J. of Pattern Recognition*, **42**(11), pp. 2922-2935 (2009).

23. Berclaz, J., Fleuret, F., Turetken, E. and Fua, P. "Multiple object tracking using k-shortest paths optimization", *Int. J. of IEEE Transactions on Pattern Analysis and Machine Intelligence*, **33**(9), pp. 1806-1819 (2011).

24. Ben Shitrit, H., Berclaz, J., Fleuret, F. and Fua, P. "Tracking multiple people under global appearance constraints", *Int. Conf. on Computer Vision*, Barcelona, Spain, pp. 137-144 (2011).

25. Kennedy, J. and Eberhart, R. "Particle swarm optimization", *Int. Conf. on Neural Networks*, **4**, Perth, Australia, pp. 1942-1948 (1995).

26. Kennedy, J. and Eberhart, R.C. "Discrete binary version of the particle swarm algorithm", *Int. Conf. on*

Systems, Man and Cybernetics, **5**, Orlando, Florida, USA, pp. 4104-4108 (1997).

27. Wang, K.P., Huang, L., Zhou, C.G. and Pang, W. "Particle swarm optimization for traveling salesman problem", *Int. Conf. on Machine Learning and Cybernetics*, **3**, pp. 1583-1585 (2003).

28. Fan, H. "Discrete particle swarm optimization for TSP based on neighborhood", *Int. J. of Computational Information Systems*, **6**(10), pp. 3407-3414 (2010).

29. Cheng, W., Maimai, Z. and Jian, L. "Solving traveling salesman problems with time windows by genetic particle swarm optimization", *Int. Conf. on Evolutionary Computation*, Hong Kong, China, pp. 1752-1755 (2008).

30. Fang, L., Chen, P. and Liu, S. "Particle swarm optimization with simulated annealing for TSP", *6th Int. Conf. on Artificial Intelligence, Knowledge Engineering and Data Bases*, **6**, Corfu Island, Greece, pp. 206-210 (2007).

31. Yin, P.Y. "Genetic particle swarm optimization for polygonal approximation of digital curves", *Int. J. of Pattern Recognition and Image Analysis*, **16**(2), pp. 223-233 (2006).

32. Viola, P. and Jones, M.J. "Rapid object detection using a boosted cascade of simple features", *Int. Conf. on Computer Vision and Pattern Recognition*, **1**, pp. 511-518 (2001).

Biographies

Mehrtash Manafifard was born in Tehran, Iran, in 1985. She received BS (Geomatics Engineering) and MS (Photogrammetry) degrees in 2007 and 2011, respectively from K.N. Toosi University of Technology, Tehran, Iran, where she is currently working towards her PhD degree. Her main research interests are close range photogrammetry, computer vision, pattern recognition, image processing and object tracking.

Hamid Ebadi obtained his PhD degree from the University of Calgary, Canada, and is currently Associate Professor and Head of the Faculty of Geomatics Engineering at K.N. Toosi University of Technology, Iran, where he has been a faculty member since 1998. His research interests include integration of photogrammetry and GIS, information extraction from satellite imagery, digital photogrammetry and videogrammetry. He has over 100 published conference papers and 40 published journal papers.

Hamid Abrishami Moghaddam was born in Iran, in 1964. He received a BS degree in Electrical Engineering from Amirkabir University of Technology, Tehran, Iran, in 1988, an MS degree in Biomedical Engineering from Sharif University of Technology, Tehran, in 1991, and a PhD degree in Biomedical Engineering from the Université de Technologie de Compiègne, Compiègne, France, in 1998.

He is currently Professor of Biomedical Engineering with K.N. Toosi University of Technology, Tehran, Iran, and since 2004 has been collaborating with the Biophysics Laboratory at the Université de Picardie Jules Verne, Amiens, France, as an Invited Researcher on Medical Image Processing,. His research interests include pattern recognition, image processing, and machine vision in areas of which he has published more than 120 articles in scientific journals and conferences. Dr. Abrishami-Moghaddam chaired the Machine Vision and Image Processing (MVIP2003) conference in Iran, in 2003. He is Vice President of the Iranian Society of Machine Vision and Image Processing and member of the Editorial Board of the Iranian Journal of Biomedical Engineering.