Training Multi-Agent Teams from Zero Knowledge with the Competitive Coevolutionary team-based Particle Swarm Optimiser

Christiaan Scheepers · **Andries P. Engelbrecht**

Abstract A new competitive coevolutionary team-based particle swarm optimiser (CCPSO(t)) algorithm is developed to train multi-agent teams from zero knowledge. The CCPSO(t) algorithm is applied to train a team of agents to play simple soccer. The algorithm uses the charged particle swarm optimiser in a competitive and cooperative coevolutionary training environment to train neural network controllers for the players. The CCPSO(t) algorithm makes use of the FIFA league ranking relative fitness function to gather detailed performance metrics from each game played. The training performance and convergence behaviour of the particle swarm is analysed. A hypothesis is presented that explains the lack of convergence in the particle swarms. After applying a clustering algorithm on the particle positions, a detailed visual and quantitative analysis of the player strategies is presented. The final results show that the CCPSO(t) algorithm is capable of evolving complex gameplay strategies for a complex non-deterministic game.

Keywords Cooperative coevolution · Competitive coevolution · Neural networks · Charged particle swarm optimiser · Zero knowledge · Multi agent system · Simple soccer

1 Introduction

The Robocup (Kitano 1993) initiative was created to promote research in the areas of robotics and artificial intelligence by offering a publicly appealing but formidable challenge. The techniques applied in training a team to win the game of soccer can be mapped to the techniques capable of solving real-world problems, such as further automating space exploration robots (Laubach et al 1998). This paper presents the competitive coevolutionary team-based particle swarm optimiser (CCPSO(t)) algorithm capable of training players for a simplified soccer game from zero knowledge about playing strategies.

The particle swarm optimiser (PSO) algorithm (Kennedy and Eberhart 1995) is a recently developed population-based optimisation method, with its roots in the simulation of the social behaviour of birds within a flock. First developed by Kennedy and Eberhart (Kennedy and Eberhart 1995) in 1995, the PSO algorithm has been more successful in solving complex problems than traditional evolutionary computation (EC) algorithms (Kennedy and Eberhart 2001). Particle swarm optimisers have proved successful in training board state evaluators for games such as Tic-Tac-Toe, Checkers and Bao (Messerschmidt and Engelbrecht 2002; Franken and Engelbrecht 2003a,b, 2004; Conradie and Engelbrecht 2006). The aforementioned training techniques require the construction of traditional game trees, and using a competitive coevolutionary PSO to train a neural network game state evaluator. Constructing traditional game trees can become impractical for games that are more complex, such as Go (Davis et al 2000). The same applies to games such as simulated soccer where it may not even be possible to always construct a game tree. In contrast to the above mentioned techniques, the technique presented in this paper does not make use of a game tree. Instead, the actions taken by each player is directly controlled by a neural network.

The CCPSO(t) algorithm presented in this paper is applied to train soccer-playing robot teams in a simplified soccer game. In addition to training a team of players, the training is performed from zero knowledge, that is, no domain information is provided to the training algorithm; only the game outcome is known during training. As no domain information is required the algorithm can easily be adapted for games other than simple soccer.

C. Scheepers · A.P. Engelbrecht
Department of Computer Science, School of Information Technology, University of Pretoria,
Pretoria 0002, South Africa

Initial analysis of the training performance showed that the particles of the CCPSO(t) algorithm were not converging on a single solution, or playing strategy (Scheepers 2013). A hypothesis is presented that explains the lack of convergence. After applying the X-means clustering algorithm (Pelleg and Moore 2000) on the particles, the produced centroids were used as the playing strategies. These playing strategies were then evaluated. Visual analysis confirmed that each centroid represents a different playing strategy. Results given in this paper show that the CCPSO(t) algorithm is capable of successfully training players to play a complex non-deterministic game.

The main contributions of this paper are: the CCPSO(t) algorithm for training multi-agent teams from zero knowledge, confirmation of a hypothesis that explains why the algorithm does not converge on a single solution, and a quantitative analysis of the learned strategies to show that it is possible to strategies from zero knowledge for a competitive team-based game.

The remainder of this paper is organised as follows: Section 2 presents the CCPSO(t) algorithm. Simple soccer and the neural network architecture is described in section 3. A detailed performance analysis is presented in section 4. The performance analysis includes an analysis of the convergence behaviour and verification of a hypothesis to explain the lack of convergence to a single playing strategy. Findings and conclusions are presented in section 5.

2 The CCPSO(t) Algorithm

Scheepers introduced the first CCPSO model for training of multi-agent teams from zero knowledge (Scheepers 2013; Scheepers and Engelbrecht 2014). The CCPSO(t) proposed in this paper augments the CCPSO algorithm by introducing a bound restriction on the personal best particle positions. The bounding operator was implemented as a guard on the personal best position update. The personal best position was only updated if all the components of the new personal best position fell within the $(-5.0, 5.0)$ predefined boundary. By bounding the personal best particle position, particles will always be pulled back into the bounded region by the cognitive component of the velocity update equation.

Bounding the region of the personal best position in turn helps to prevent values from moving too far away from the active region of the neural network activation functions as used in the problem being optimised. Specifically, the bounding operator was shown to avoid saturation of the neural network weights during the search process (van Wyk and Engelbrecht 2010; Scheepers 2013).

Each game agent, or neuro-controlled player, is represented in the CCPSO(t) algorithm as a separate swarm of particles. Each particle position represents a weight vector for the neural network of the corresponding player. The training objective for the CCPSO(t) algorithm is to find the best performing particle positions for each of the swarms. These best performing particle positions represent the best performing players for each of the corresponding player positions.

Previous work showed that a competitive coevolutionary training environment can be seen as a dynamical environment (Scheepers 2013). To improve performance in the presence of a constantly changing search space, the CCPSO(t) algorithm makes use of the charged PSO as introduced by Blackwell and Bentley (2002a,b); Blackwell (2003).

The CCPSO(t) uses the *FIFA relative fitness function* to measure relative fitness for each particle. The FIFA relative fitness function calculates the relative fitness for a player as follows:

$$\mathscr{F}(t) = \sum_{n=1}^{N} (\mathscr{M}_n \times \frac{200 - \mathscr{T}_n}{100}) \tag{1}$$

where N is the number of games played in the competitive coevolution tournament, $\mathscr{M}_n$ is the n'th game outcome, defined as:

$$\mathscr{M}_n = \begin{cases} 3 & \text{for a victory} \\ 1 & \text{for a draw} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

and $\mathscr{T}_n$ is the n'th opposition team's rank. Each tournament team consists of randomly chosen team members. The opposition team rank, $\mathscr{T}_n$, is calculated by averaging the rank of the members of the opposition team as follows:

$$\mathscr{T}_n = \frac{\sum_{p=1}^{P} r(p)}{P} \tag{3}$$

where function $r(p)$ represents the rank of player p in a team consisting of P players. A player's rank is determined based on the relative fitness of the particle in the previous iteration. The player with the highest relative fitness in the previous iteration is ranked first, followed by the player with the second highest relative fitness, and so forth.

Algorithm listing 1 presents the pseudocode implementation for the CCPSO(t) algorithm as used in this work. To avoid early stagnation and aid exploration, a hall of fame (HOF) was also used (Rosin and Belew 1995, 1997). The HOF maintains a collection of the best performing individuals from each generation since the evolutionary process has started. This collection of previous best performing individuals helps to combat the loss of exploration by preserving past good characteristics.

3 Simple Soccer Reference Problem

This section provides an overview of the simple soccer training model. The simple soccer training model was specifi-

```
Create and initialise a swarm of neural networks for each game position.
repeat:
    for each swarm $O(t)$ do
        for all swarms $O_s(t) \neq O(t)$ do
            Add each personal best position to the competition pool for swarm $O_s(t)$.
            Add each particle to the competition pool for swarm $O_s(t)$.
        end for
        for each particles $P_i$ (or NN) in the swarm $O(t)$ do
            repeat:
                Select team members and opponents from the competition pools.
                Play a game using the selected players.
                Record if game was won, lost or drawn.
            until predefined number of games has been played.
            Determine a score for each particle.
            if the particle position is within the search space boundary
                Compute new personal best position based on score.
            end if
        end for
        Compute new neighbourhood best position.
        Update particle velocities.
        Update particle positions.
    end for
until all swarms converge or iteration limit is reached.
The Global best particle in each swarm is the trained neural network game agent for the corresponding team position.
```

Algorithm 1: Pseudocode for a basic CCPSO(t) algorithm implementation.

cally designed for this study. The neural network architecture, as used by the neuro-controlled players, is also discussed.

Simple soccer is played on a 5×6 grid based field as depicted in figure 1. The game is played by two teams, A and B, each consisting of two players A_1, A_2 and B_1, B_2 respectively.

Each player is controlled by a feed forward neural network (Haykin 1998). Information is fed to the neural network using four neurons per class of object on the field. Classes of objects are the ball, nearest team-mate, nearest opponent, and wall. The four neurons represent the distance to the object in a specific direction, i.e. north, south, east or west. The input functions for the four neurons are defined as follows:

$$f(north) = \begin{cases} \frac{\Delta y}{d^2} & \text{if } y > 0 \\ 0 & \text{if } y \leq 0 \end{cases} \quad (4)$$

$$f(south) = \begin{cases} \frac{\Delta y}{d^2} & \text{if } y < 0 \\ 0 & \text{if } y \geq 0 \end{cases} \quad (5)$$

$$f(west) = \begin{cases} \frac{\Delta x}{d^2} & \text{if } x < 0 \\ 0 & \text{if } x \geq 0 \end{cases} \quad (6)$$

$$f(east) = \begin{cases} \frac{\Delta x}{d^2} & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases} \quad (7)$$

where Δx is the distance along the east-west direction and Δy is the distance along the north-south direction; d is defined as $d = \sqrt{\Delta x^2 + \Delta y^2}$.

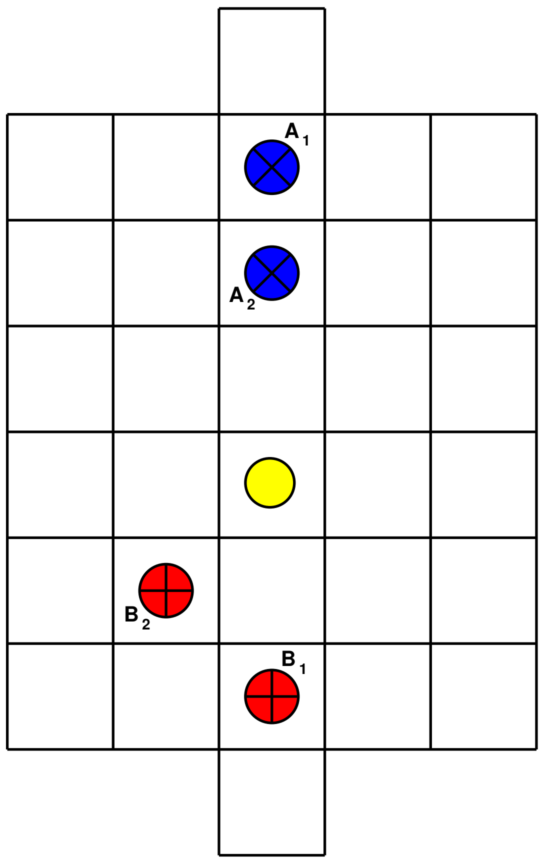

Fig. 1 Simple soccer player positions

The neural network output consists of an eight dimensional floating-point vector with values scaled between 0 and 1. The vector is constructed by combining the movement and kick vectors. The movement vector indicates the movement direction, and the kick vector indicates the direction to kick the ball. The four values per vector are desirability factors to move or kick in each of the primary directions: forward, backward, left, and right. A desirability factor larger than 0.5 indicates a desire to move or kick in the corresponding direction.

The work presented in this paper used a feed forward neural network with five hidden neurons (Scheepers 2013). The hyperbolic tangent activation function was used for both the hidden and output layer neurons.

For a detailed discussion on simple soccer and the neural network architecture used in this work refer to Scheepers (2013) and Scheepers and Engelbrecht (2014).

4 Performance Analysis

This section discusses the performance of the CCPSO(t) algorithm. The experimental procedure is described in section 4.1, followed by an analyses of the convergence behaviour in section 4.2. The convergence behaviour analysis showed that the swarms did not converge. Section 4.3 presents and investigates the multiple strategies hypothesis as a reason for the lack of convergence. Finally, sections 4.4, 4.5, and 4.6 present a gameplay analyses of the various trained players.

4.1 Experimental Procedure

The parameters of the CCPSO(t) algorithm were optimised for training using the simple soccer training model discussed in the previous section. Parameter optimisation was done using a parallel coordinate visualisation technique (Franken 2009). The parallel coordinate visualisation technique works by plotting the performance of different parameter combinations along with the measured performance of each parameter combination. The well performing values of the various parameters can be seen by highlighting the area formed by the top performing parameter combinations. For a full parameter sensitivity analysis, the interested reader is referred to Scheepers (2013) and Scheepers and Engelbrecht (2014). Table 1 provides a summary of the optimised parameter values used in this work.

The optimised parameters were used to run 30 independent simulations. Each simulation was executed with an iteration limit of 500 iterations. The measurements reported below were taken over the 30 independent simulations.

The following five subsections analyse and discuss in detail the training performance of the CCPSO(t) algorithm. The analysis starts by investigating the convergence behaviour of the CCPSO(t) algorithm. The convergence behaviour showed that the swarms did not converge during training. The multiple strategies hypothesis is then presented to explain why the swarms did not converge. X-means clustering was applied to verify the aforementioned hypothesis. The reader should note that any other clustering algorithm could have been used. X-means clustering was chosen for its simplicity and the fact that it dynamically determines the optimal number of clusters. Finally, the found cluster centroids were evaluated visually and quantitatively as trained candidate players.

Due to space concerns, the visual analysis presented below was performed on the cluster centroids of a single simulation. Visual analysis of the cluster centroids for the other simulations yielded similar results.

4.2 Convergence Behaviour

In order to analyse the convergence behaviour of the particles, the swarm diversity was measured for each of the swarms during the search process. Swarm diversity is calculated as the degree of dispersion of particles and is formally defined in Krink et al (2002). A high swarm diversity is desired early on in the search process. A high swarm diversity promotes better exploration of the search space. As the search progresses, a lower swarm diversity becomes desirable. A low swarm diversity indicates more particles are converging on the already found solution.

Figure 2 depicts the swarm diversity obtained by measuring the swarm diversity for one of the swarms when using the CCPSO(t) algorithm to train simple soccer players. The average swarm diversity as well as standard deviation over 30 simulations is shown. The swarm diversity of the

Table 1 CCPSO(t) parameters

Parameter	Value
Swarm type	Atomic (50%)
Neighbourhood structure	Von Neumann
Inertia weight	0.729844
Social constricting coefficient	1.49618
Cognitive constricting coefficient	1.49618
Initial particle velocity	0
Initial particle positions	$R(-1,1)$
Swarm size	20 particles
Perception limit	500.0
Perception core	2.0
Charge magnitude	15.0
Maximum particle velocity	15.0
Hall of fame size	4
Hall of fame update iterations	30
Competition pool size	15
Personal best re-evaluation iterations	3
Personal best bounding restriction	$(-5.0, 5.0)$

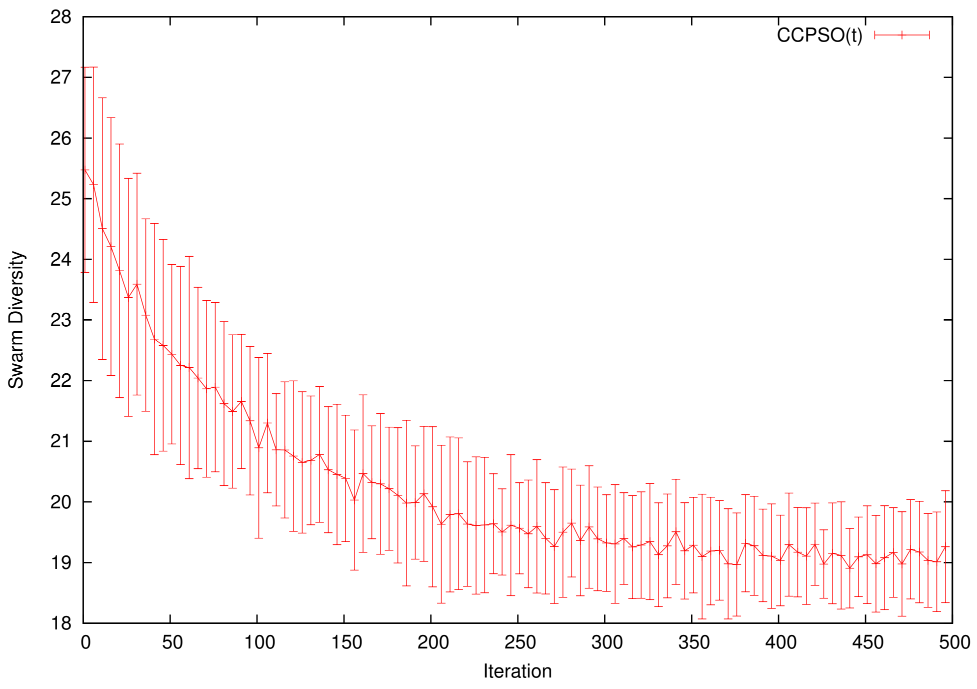

Fig. 2 Swarm diversity using the CCPSO(t) algorithm over 30 simulations

other swarms reflect a similar pattern as that depicted in figure 2. The initial drop in swarm diversity is clearly visible up to iteration 250. However, after iteration 250 the diversity no longer decreases a notable amount, indicating that the swarm is not converging on a single solution. Scheepers showed that the high swarm diversity was not due to the use of the charged PSO (Scheepers 2013).

Convergence in the CCPSO(t) algorithm is driven by the shared $\mathbf{x}_{gbest}$ component of the velocity update equation. All the particles in the swarm is pulled towards the global best position. The search process leads to a behaviour where the global best position jumps around in large steps right after the swarm is initialised. As the search process progresses and a smaller area of the search space is exploited, the jumps in the global best position becomes smaller. Obviously, should a faraway solution be found that presents a better solution, the global best position moves in a single large jump to that new location.

As previously noted, the problem investigated by this work is not a static, predictable one, but is rather classified as a dynamic environment problem in which the search space constantly changes. For a dynamic environment, it is expected that the global best position changes more often than for a static problem. However, in the case of the simple soccer problem investigated in this work, convergence on specific gameplay strategies is hoped for. It is expected that changes in the global best position decrease to extremely small jumps as a player/swarm exploits a gameplay strategy.

In order to investigate the behaviour of the global best position, a new measurement was implemented to track the magnitude of the jumps in the position of the global best position between consecutive iterations. The global best movement measure, $\Phi(t)$, is defined as

$$\Phi(t) = \sum_{j=1}^{J} |\mathbf{x}_{gbest,j}(t) - \mathbf{x}_{gbest,j}(t-1)| \tag{8}$$

where J is the number of dimensions of $\mathbf{x}_{gbest}$. This new measure simply adds the magnitude of the change in each dimension together. Small values indicate a small change in position, whereas large values indicate a larger change in position. The measurement gives an indication of the magnitude of a change in position.

Figure 3 depicts the results obtained when measuring $\Phi(t)$ for the CCPSO(t) algorithm. Both the average and standard deviation is shown. The graph indicates fairly chaotic behaviour in the swarm, with the global best position changing continuously by making fairly large jumps. A high standard deviation among the 30 evaluated simulations is also noted.

The CCPSO(t) algorithm exhibited a slight decrease in the size of the jumps noted up to around iteration 200, after which the jump size stabilised. The average jump size still exceeded 200 after 500 iterations, with a noticeably high standard deviation among the 30 simulations evaluated.

The observed behaviour contradicts the expected behaviour that the global best position jumps would become smaller

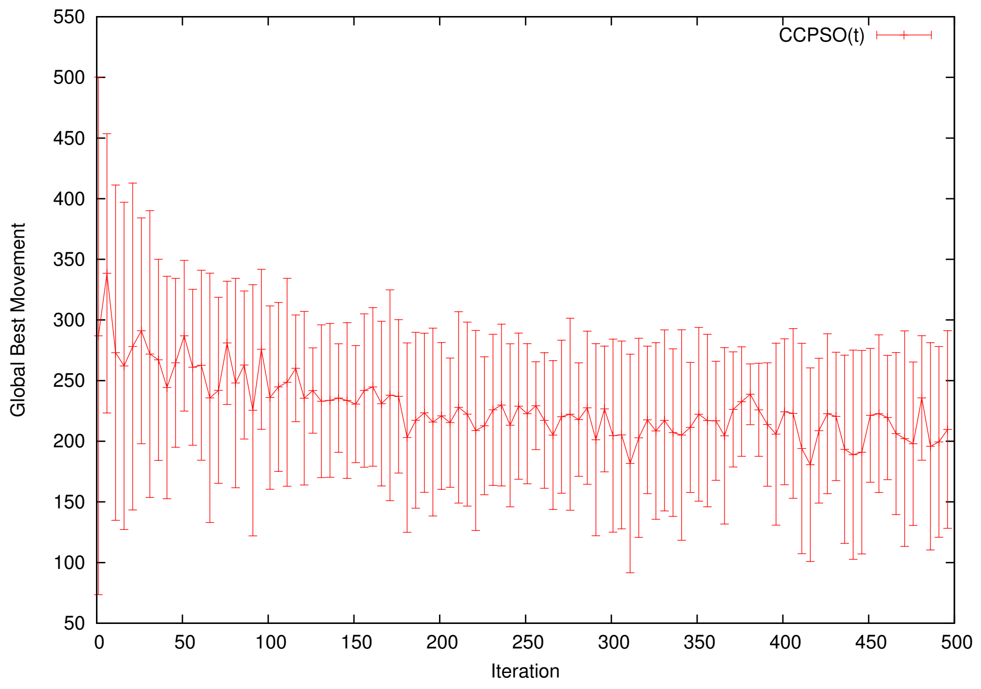

Fig. 3 Measured Φ using the CCPSO(t) algorithm over 30 simulations

as the search progresses. The observed behaviour clearly indicates that the CCPSO(t) algorithm is not leading to any form of convergence.

From the $\mathbf{x}_{gbest}$ behaviour analysis it is clear that the global best position still changes position quite dramatically, and no single solution is converged onto.

4.3 Multiple Strategies Hypothesis

The observed jumping behaviour might be explained by evaluating the following hypothesis: If multiple gameplay strategies are found by each swarm, the optimal strategy, represented by the global best position, will move between these strategies as no one strategy prove superior to other gameplay strategies. An offensive strategy might prove superior to a defensive strategy against an extremely aggressive opposition team. In other cases, however, the defensive strategy might prove superior or an alternative attack strategy might exploit a weakness against an opposing team leading to a win. The optimal and best performing strategy will thus vary between iterations with no single optimal strategy being superior.

Multiple gameplay strategies are present in the search space as different optimal areas (or peaks in the hyper dimensional search space). If multiple strategies (solutions) exist, particles will “group” around these optimal areas forming clusters of particles, with each cluster effectively converging on its own optimal position. The behaviour observed in the previous subsection can then be explained as the global best position changing position between different clusters. Because each cluster is spaced noticeably far away from the others, the jumps between the clusters are clearly visible in the figures.

In order to evaluate the clustering hypothesis, X-means clustering was applied to cluster the global best positions found for each iteration that the algorithm ran. The X-means clustering algorithm extends the K-means clustering algorithm with efficient estimation of the optimal number of clusters (Pelleg and Moore 2000). X-means allow for data to be clustered without knowing beforehand how many clusters exist. X-means also attempts to address the computational scalability shortcoming of the K-means algorithm when dealing with larger datasets. The X-means clustering algorithm was applied to the global best positions for each of the swarms separately. This resulted in four sets of clusters (one set for each player position).

The X-means algorithm revealed that clusters were indeed formed in the swarms. The number of clusters found per swarm varied between two and five. In effect, this indicate that each swarm, or game agent learned between two to five different gameplay strategies. The next subsection explores the different strategies that were learned. It should be noted that two clusters may represent exactly the same behaviour, because equivalent neural networks can be obtained by simply swapping hidden layer neurons. The resulting neural networks, after swapping hidden neurons, have exactly the same behaviour.

To confirm the proposed hypothesis, each cluster is analysed and the gameplay strategy represented by each cluster is identified. To achieve this goal, agents were trained with the CCPSO(t) training algorithm and clusters were computed for each of the four swarms that represented the trained players. Each cluster centroid was taken as the playing strategy of the player that corresponds to that swarm. Each of these candidate players was then visually analysed when playing against every other candidate player.

4.4 Player Strategy Analysis

Figure 1 labels the simple soccer player positions. For each position, each centroid was analysed independently against every other centroid. The observed behaviour of each centroid is discussed in this subsection. Analysis of the combined effect of the observed behaviours is provided in the next subsection. It should be noted that when the player behaviours are viewed in isolation, the behaviours described below can be seen as undesirable or even counter-productive, though the combined effect and interaction of these behaviours sheds light on how these behaviours evolved. The notation used, $X_y(z)$, indicates team X, position y, and centroid number z.

4.4.1 Player A_1

Four centroids were identified for player A_1. Each centroid's behaviour is described below.

Centroid 1*:* The first complex gameplay behaviour observed was ball chasing behaviour. When the ball is kicked over the block player A_1 occupies, player A_1 moves in the direction of the ball to gain ball ownership. Once player A_1 has the ball, the ball is kicked in the direction of the opposition team's goal. This chasing behaviour is repeated until a goal is scored. Figure 4 depicts the ball chasing behaviour. For illustrative purposes, only the applicable players are shown in the figure. Chasing behaviour was limited to forward and backward movement.

Centroid 2*:* Basic forward and backward ball chasing behaviour was observed in some games. However, it was not as successful as that of centroid 1. Sideways kicking behaviour was also observed, where player A_1 kicked the ball to the left and continued to move forward.

Centroids 3 *to* 4*:* Basic forward goal-running behaviour was observed. In cases where an opponent player obstructed the forward path a kick was used to pass the ball over the opponent player. No forward, backward, or sideways ball-chasing behaviour was observed. This was the most simplistic playing behaviour observed.

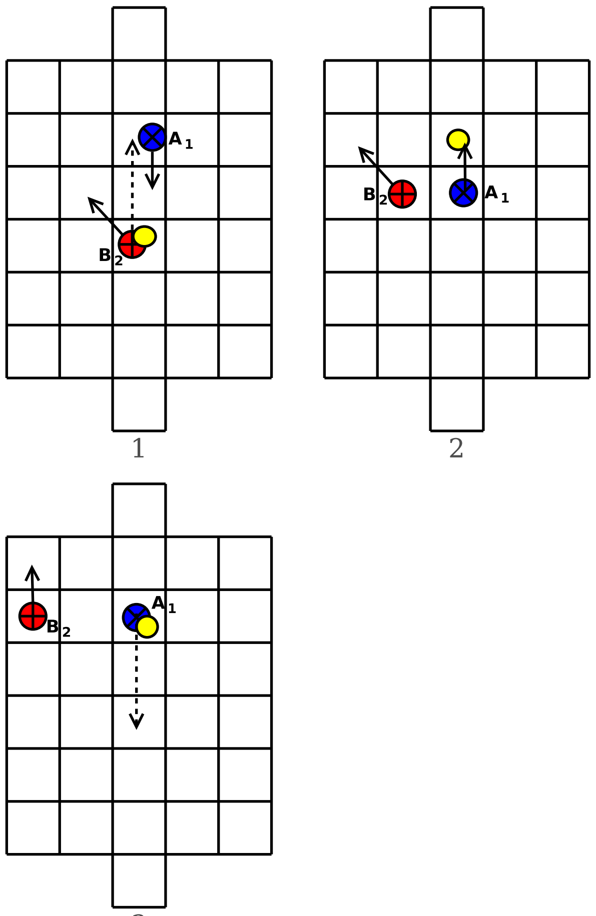

Fig. 4 Player $A_1(1)$ demonstrating ball fetching behaviour

4.4.2 Player A_2

Two centroids were identified for player A_2, each centroid's behaviour is described below.

Centroid 1*:* The player moves, without the ball, to the centre line of the field and then proceeded in the direction of the opposition goal. In cases where the ball ownership was gained a constant rule applied: If the opposition goal is three blocks away, kick to the left, or if the opposition goal is four blocks away, kick the the right. When viewed in isolation, this kicking behaviour does not look sensible. The next subsection shows that this behaviour forms part of a more complex strategy. Figure 5 depicts this behaviour.

Centroid 2*:* Player A_2 simply moved to the right of the field, irrespective of the ball position.

4.4.3 Player B_1

Four centroids were identified for player B_1, and all the centroids for player B_1 exhibited similar behaviour.

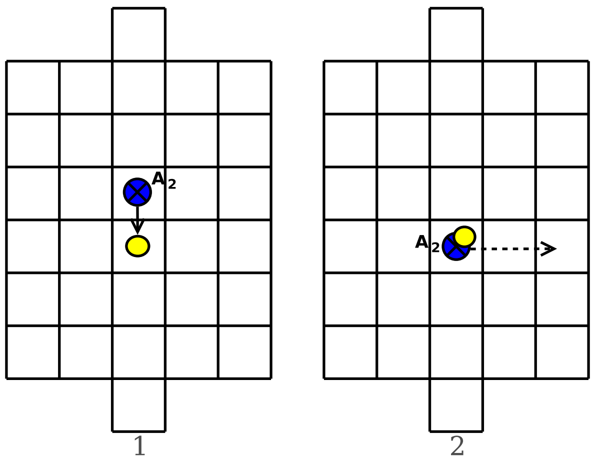

Fig. 5 Sideways kick

Centroids 1 *to* 4*:* Player B_1 exhibited a forward-moving ball-kicking behaviour. In cases where opposition players blocked the way forward, player B_1 kicked the ball over them. Once player B_1 recovered the ball, a kick into the goal is performed. Figure 6 depicts a typical game. Visual analysis showed this strategy was quite successful where the outcome was often determined by the ball's random starting position.

4.4.4 Player B_2

Four centroids were identified for player B_2, each centroid's behaviour is described below.

Centroid 1*:* Player B_2 simply moved to the left of the field, irrespective of the ball position.

Centroid 2*:* An interesting looping behaviour was observed for player B_2, where the player moves one block forward to the left, and then backwards one block to the right. This looping behaviour led to an interesting catch of the ball, as depicted in figure 7. Although the looping behaviour overshadowed most of this players' other movements, the actual position of where on the field the player looped is at least partially linked to the position of player B_1. The final sideways movement depicted in figure 7 is due to player B_1 (not shown in the figure) moving closer to the opposition goal.

Centroid 3*:* The player only moves in a forward and left direction. Once the edge of the board is reached, the player continues to move forward. Ball-kicking behaviour is not observed, except in a very special case: if the player has ball ownership in the top left corner of the board, a right forward kick is performed that results in a goal being scored. This behaviour is shown in figure 8.

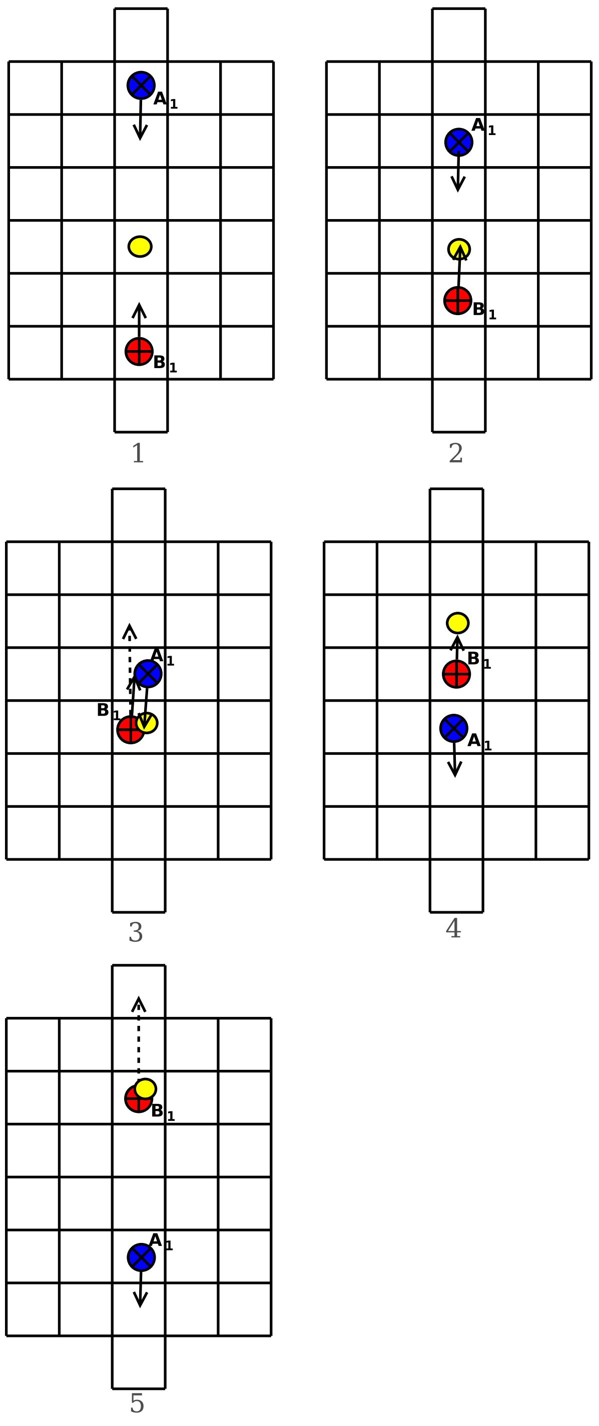

Fig. 6 Player $B_1(1)$ scores a goal

Centroid 4*:* This player exhibited the most complex behaviour of all the behaviours analysed: sideway movement, looped movement, chasing behaviour, and opponent avoidance behaviour were observed. Figure 9 depicts a simulation where player B_2 moved away from the goal only to return just in time to prevent the opposition team from scoring a goal. A number of games showed ball-chasing and recovery behaviour. Looping behaviour where the player moved one

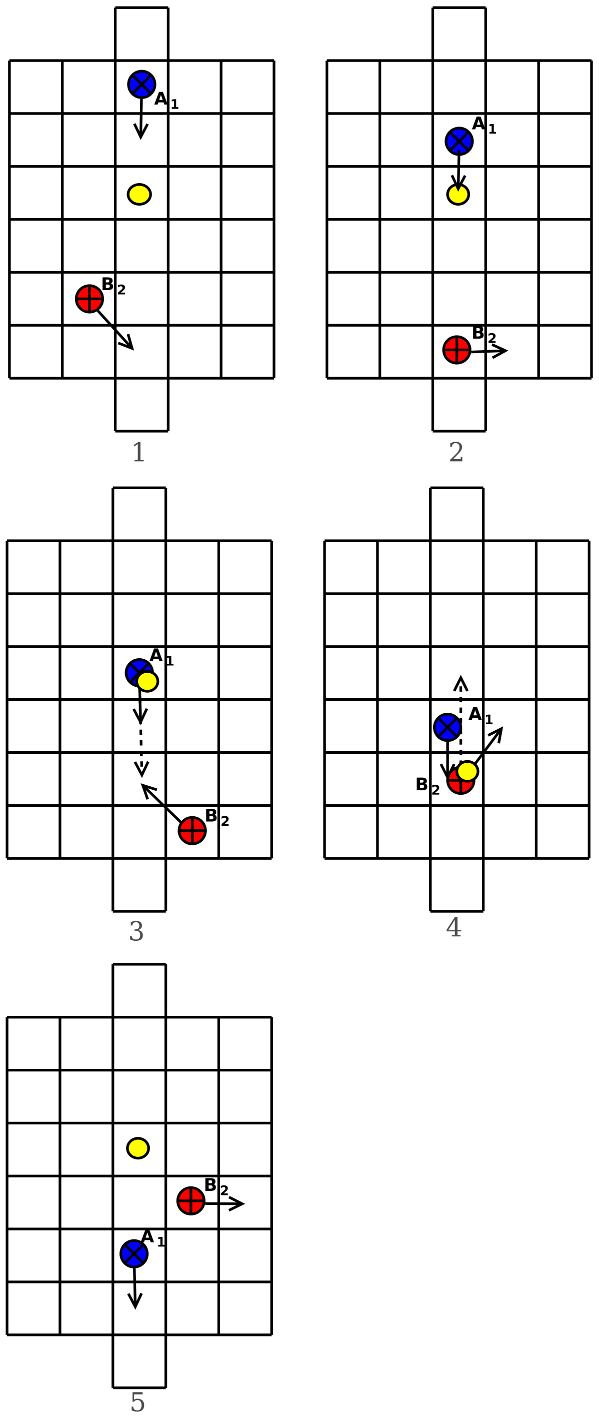

Fig. 7 Player $B_2(2)$ catches the ball

block left, followed by one block right, and so forth, was noted in other games.

Analysis of the playing strategies represented by the centroids in isolation demonstrated that each centroid does indeed represent different behaviours.

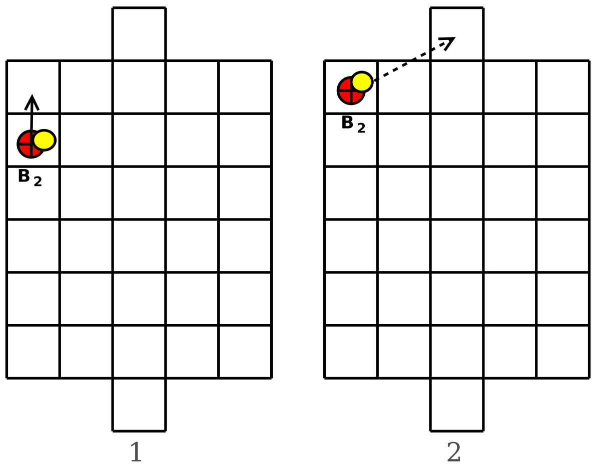

Fig. 8 Player $B_2(3)$ kick sideways

4.5 Game Strategy Analysis

Analysis of each behaviour showed that the centroids do indeed represent different behaviours, and the proposed hypothesis that each centroid represents a different optimum is true. This subsection explores the more complex behaviours that emerged from different permutations of these behaviours in a team context. Interaction between players clearly demonstrates scenarios where some of the previously shown to be *bad* behaviours can result in good gameplay strategies, scoring more goals and preventing the opponents from scoring goals more often. For each of the games discussed below it is shown which player (centroid) behaviour permutations form part the game. The centroid numbers correspond to the behaviours analysed in the previous subsection.

The four identified gameplay strategies are discussed below in detail.

4.5.1 Ball Ownership Exchange

The first complex gameplay strategy that was observed is illustrated in figure 10. The players that made up two teams were $A_1(1)$, $A_2(1)$, $B_1(3)$ and $B_2(2)$. The gameplay features various kicks matching individual behaviours previously analysed. Player B_2 also moved in to intercept the ball successfully. As previously noted, the position of player B_2 is loosely linked to the position of B_1 and this is also the reason why player B_2 moved in to intercept the ball. The final kick performed by player A_2 effectively takes the ball out of play leading to a draw.

Even though the game ended in a draw due to none of the players being able to find the ball in the end, the gameplay strategy visible here is fairly complex.

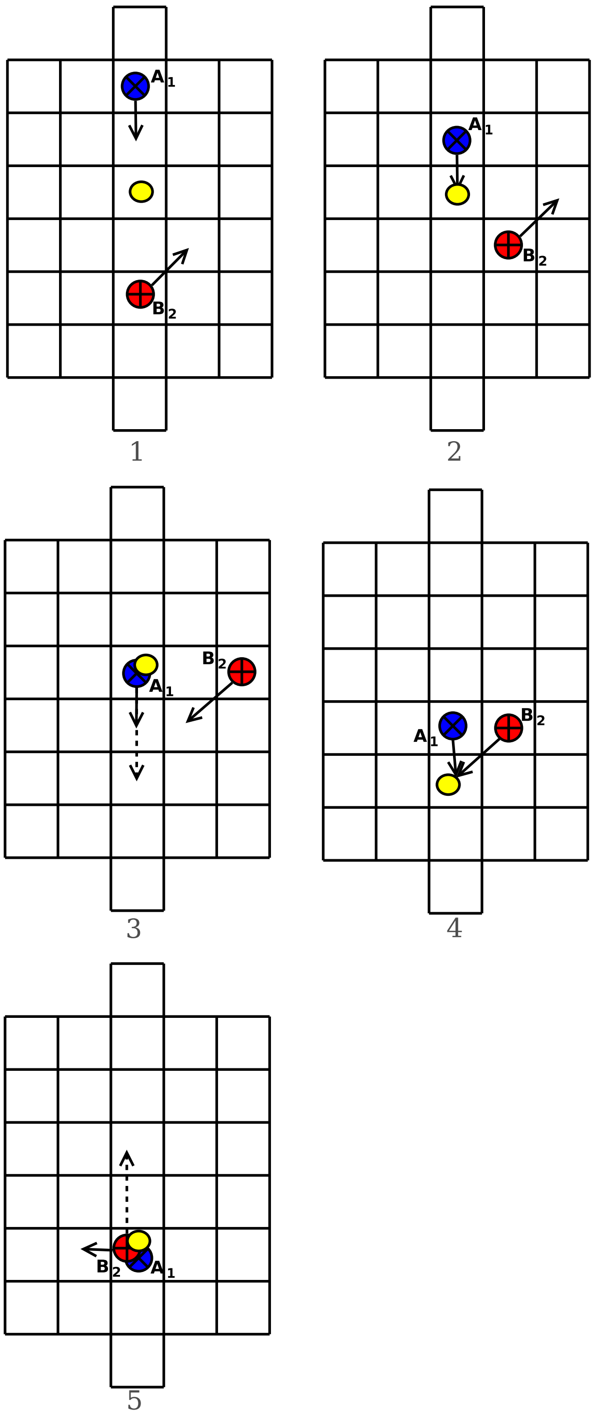

Fig. 9 Player $B_2(4)$ moves over the field and returns to protect the goal

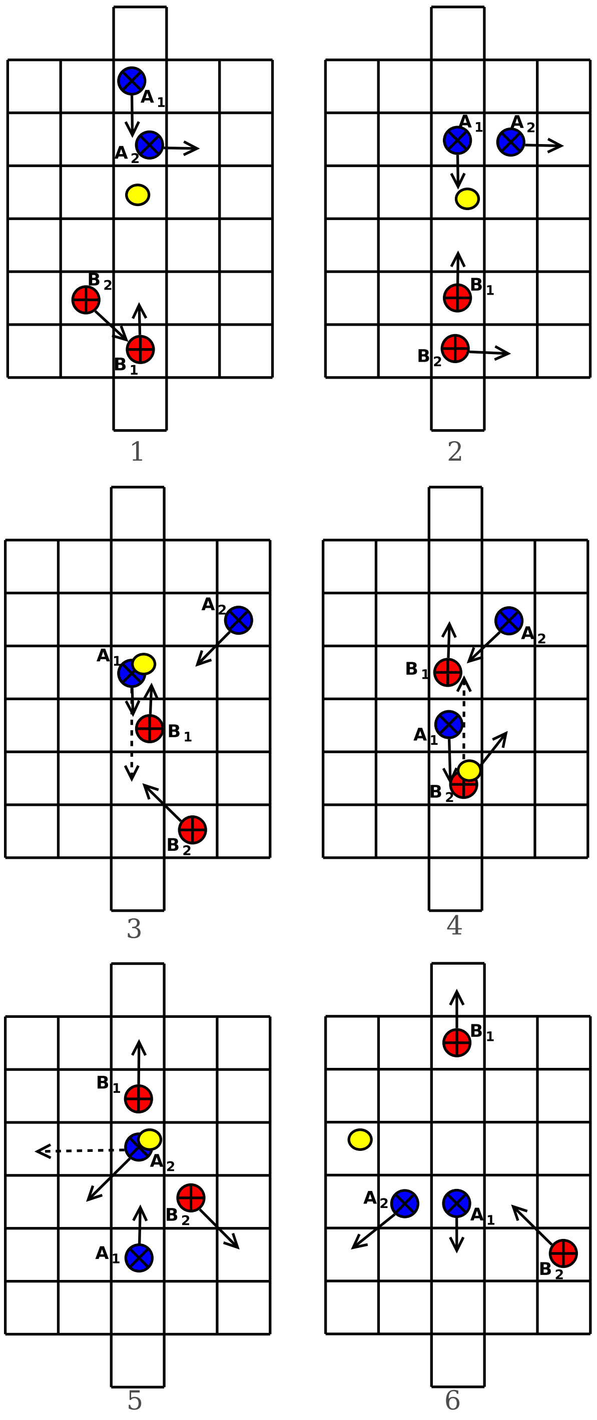

Fig. 10 Multiple ball ownership exchanges

4.5.2 Anticipatory Counter-move

As with the previous game analysed, the second game also revealed how the previously individually analysed behaviours led to more complex gameplay strategies if combined and not studied in isolation. Figure 11 depicts the second game to be analysed with players $A_1(4)$, $A_2(1)$, $B_1(4)$ and $B_2(3)$. Subsection 4.5.1 showed how player B_2 had the potential to score a goal by moving on the left edge of the playing field with the ball, all the way to the team A side. It was also shown that player A_2 kicked the ball to the left once it had the ball and was still four blocks away from the team B goal.

The second game demonstrates how player B_2 learned to exploit the predictable sideways kick behaviour of player A_2.

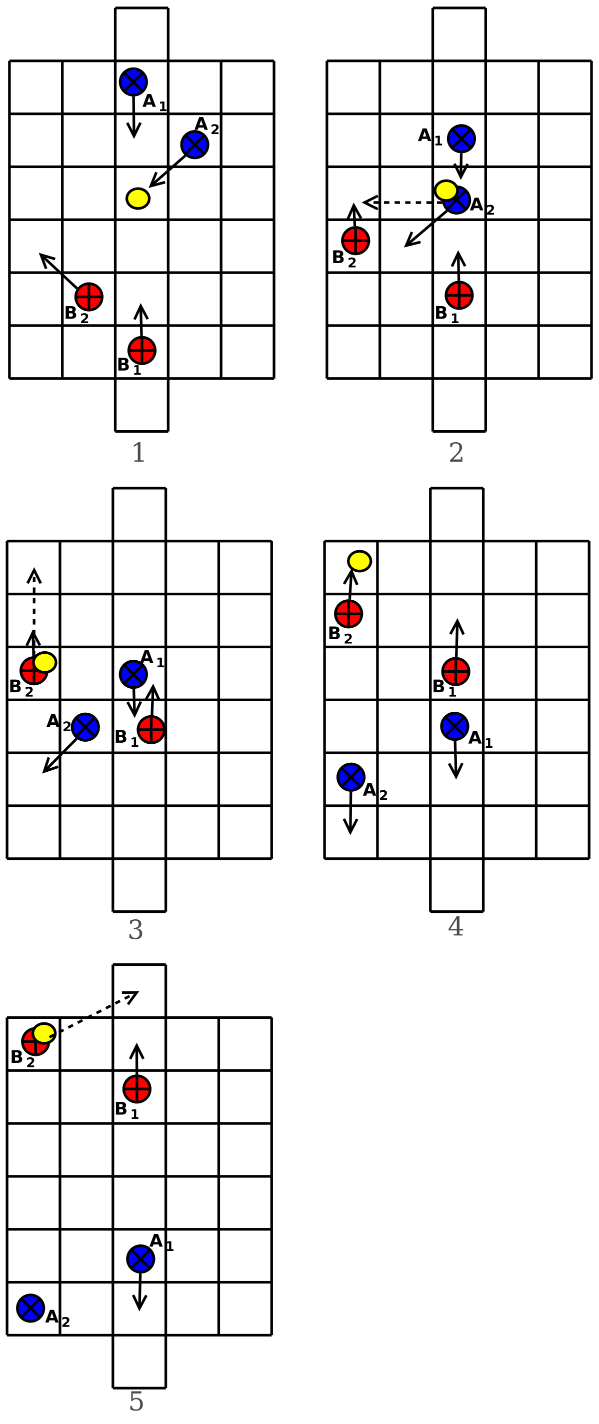

Fig. 11 Example of the anticipatory counter-move gameplay strategy

While the sideways kick in other games forces a draw result, in this game the sideways kick leads to a loss for team A.

4.5.3 Runaround Movement

The third game, as depicted in figure 12, demonstrates dribbling behaviour. Players $A_1(4)$, $A_2(2)$, $B_1(1)$ and $B_2(4)$ participated in the game.

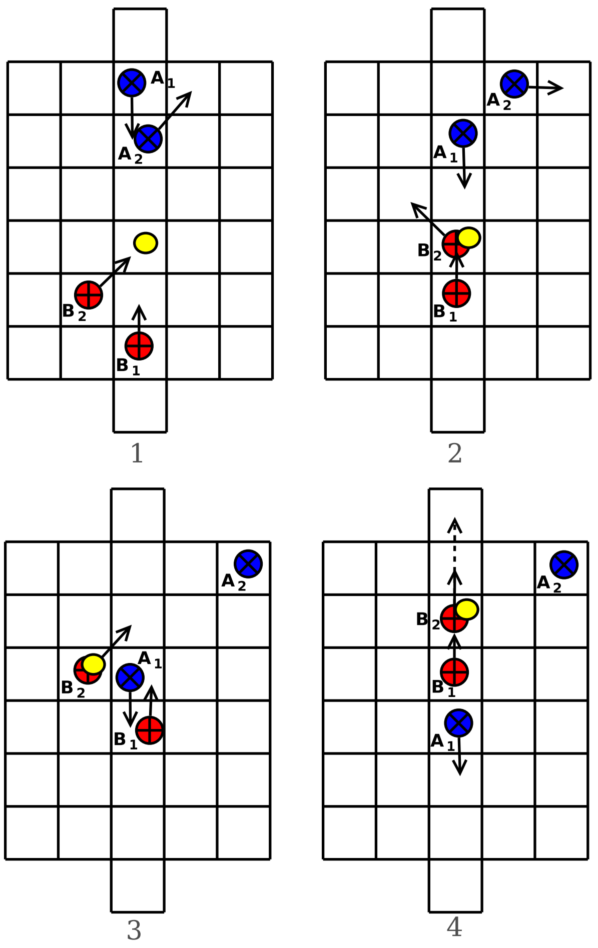

Fig. 12 Example of the runaround movement gameplay strategy

Dribbling was used by player B_2 to effectively bypass player A_1 without risking loss of ball ownership. The move is finished off by moving back onto the middle line in a sideways forward movement, allowing for a direct goal kick.

The effectiveness and success of this strategy is clearly visible.

4.5.4 Complex Comeback

The final game that demonstrated a unique complex playing strategy demonstrates how an offensive strategy is turned around to defend against a goal being scored. Players $A_1(4)$, $A_2(2)$, $B_1(2)$ and $B_2(4)$ played in the game depicted in figure 13, showing player B_2 moving aggressively sideways and forward until team A assumes ownership of the ball.

Player A_1 follows a straightforward attack strategy where the ball is kicked towards the team B goal, while the player also moves towards the goal. The ball is kicked over player B_1 to prevent losing the ball to team B. Player A_1 keeps on moving towards the block with the ball for the final goal kick. However, player B_2 at this point is the same distance away from the ball, also moving towards the ball. The two

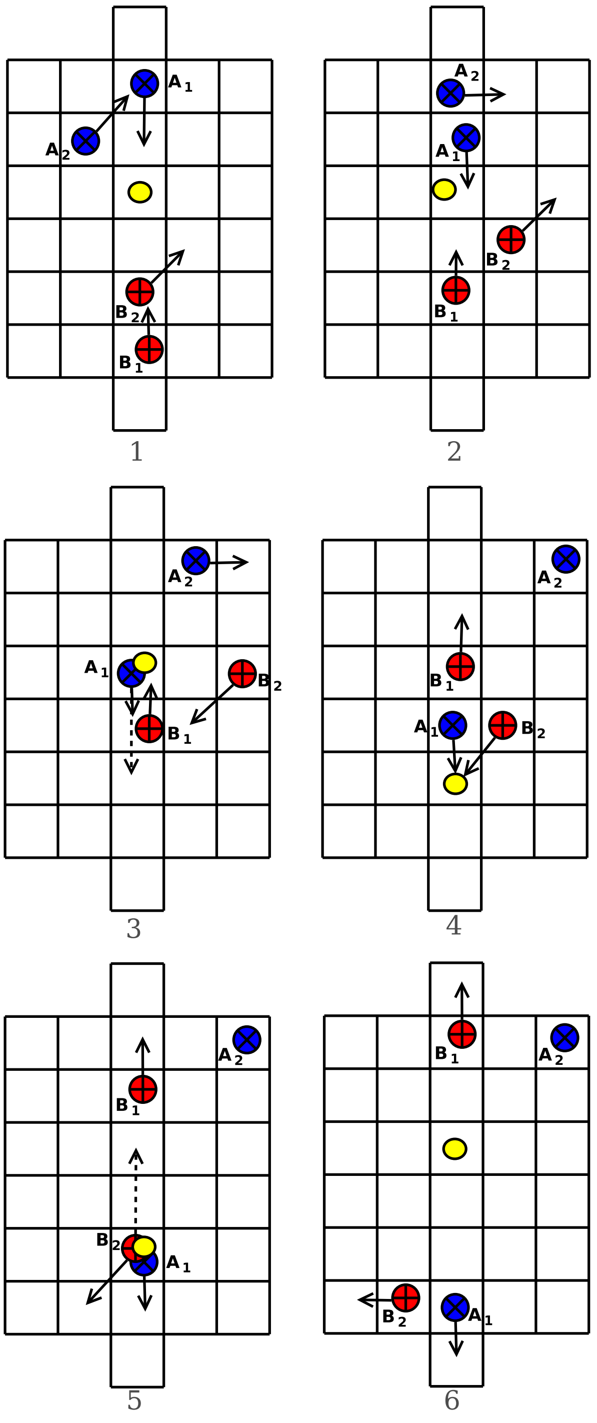

Fig. 13 Example of the complex comeback gameplay strategy

players arrive on the block with the ball at the same time, in which case ball ownership is determined randomly. In this case, B_2 managed to attain ownership. The ball is subsequently kicked towards the middle of the board, where no other players manage to retrieve it. The game ended in a draw.

4.6 Quantitative analysis

The visual analysis presented a limited-scope review of the player behaviours, although it did show behavioural traits that evolved which would otherwise not be visible in a quantitative analysis. This section provides the reader with a quantitative analysis of the trained players.

In order to compensate for the non-deterministic nature of simple soccer, each player, representing a cluster centroid in the particle swarm, was evaluated playing 15000 games in each team combination. The results are presented in tables 2 and 3. Table 2 presents the results for the different team permutations averaged over all the opposition team permutations. Four measurements are reported in the table: The average number of goals scored in each game, the average iterations it took for a goal to be scored, the percentage of games that ended in a draw, and the percentage of games the team won. It should be noted that the percentage of games ending in a draw are reported in a separate column in the tables and not calculated as part of the win/loss percentage. For example, the team $\{A_1(1),A_2(2)\}$ won 75% of the rounds not ending in a draw. Table 3 presents the same results at player level.

A number of observations can be made by reviewing the team performance results:

- Team B teams never lost more than 50% of the games.
- Three team B teams won 100% of the games that did not end in a draw. A minimum of 30% of the games played by these teams ended in draws.
- Two team A teams lost 100% of the games. A minimum of 37% of the games played by these teams ended in draws.
- Team $\{A_1(2),A_2(2)\}$ did not score a single goal in all the games.
- 20.41% of the total number of games played resulted in draws.
- With the exception of three team A teams, $\{A_1(1),A_2(1)\}$, $\{A_1(1),A_2(2)\}$ and $\{A_1(2),A_2(1)\}$, it took an average of 5 iterations per goal. For the three team A teams it took an average of 6 iterations per goal.

The results for the two team A teams, $\{A_1(2), A_2(1)\}$ and $\{A_1(2), A_2(2)\}$, that lost all of their games can be traced back to the player performance figures in table 3 for player $A_1(2)$, revealing that this player did not win a single game in any team combination. In addition, this player also had the highest game draw percentage of 38.58%.

The average number of iterations per goal can also be related to the player performance figures. Player $A_1(1)$ and $A_1(2)$ are the only players that had an average of 6 iterations per goal, corresponding to the three team A teams, $\{A_1(1), A_2(1)\}$, $\{A_1(1), A_2(2)\}$ and $\{A_1(2), A_2(1)\}$, in table 3. The fourth team A combination, $\{A_1(2), A_2(2)\}$, did not score

Table 2 Team performance

Team		Average goals scored	Iterations per goal	% draws	% wins
A_1	A_2				
1	1	2.482	5.952	22.10	37.50
1	2	3.286	6.479	16.23	75.00
2	1	0.533	6.000	39.86	0.00
2	2	0.000		37.30	0.00
3	1	2.268	5.000	16.25	12.50
3	2	3.403	5.000	7.72	37.50
4	1	2.272	5.000	16.17	12.50
4	2	3.403	5.000	7.67	25.00
B_1	B_2				
1	1	3.540	5.023	12.66	75.00
1	2	1.844	5.032	36.95	50.00
1	3	4.037	5.022	7.53	62.50
1	4	1.761	4.968	39.35	62.50
2	1	3.538	5.023	12.60	75.00
2	2	2.439	5.031	25.46	87.50
2	3	4.056	5.022	7.65	62.50
2	4	2.198	4.967	31.10	100.00
3	1	3.536	5.023	12.55	75.00
3	2	2.434	5.031	25.63	87.50
3	3	4.048	5.023	7.50	50.00
3	4	2.196	4.967	31.35	100.00
4	1	3.539	5.023	12.35	75.00
4	2	2.434	5.032	25.52	87.50
4	3	4.036	5.023	7.64	50.00
4	4	2.219	4.994	30.57	100.00

Table 3 Player performance

Player	Average goals scored	Iterations per goal	% draws	% wins
A_1				
1	2.884	6.215	19.17	56.25
2	0.267	6.000	38.58	0.00
3	2.836	5.000	11.98	25.00
4	2.837	5.000	11.92	18.75
A_2				
1	1.889	5.454	23.59	15.63
2	2.553	5.493	17.23	34.38
B_1				
1	2.796	5.011	24.12	62.50
2	3.057	5.011	19.20	81.25
3	3.053	5.011	19.26	78.13
4	3.057	5.018	19.07	78.13
B_2				
1	3.538	5.023	12.59	75.00
2	2.288	5.031	28.39	78.13
3	4.044	5.023	7.58	56.25
4	2.093	4.974	33.09	90.63

any goals so no average was recorded. From the Simple Soccer rules it can be deduced that the minimum number of iterations required to score a goal is 4 in the case where player A_1 or B_1 scores the goal and 5 in the case where player A_2 or B_2 scores the goal. Only player $B_2(4)$ had an average number of iterations per goal less than 5. Based on the minimum number of iterations needed to score a goal, it can be deduced that player $B_2(4)$ had to have scored the goals in those cases. Player $B_2(4)$ was also previously shown, in subsection 4.4.4, to have exhibited the most complex behaviour in the visual analysis, a trait that is also clearly visible in the discrete measurements. The team performance results shown indicate that $B_2(4)$ won all the games in all its team combinations, except when paired with player $B_1(1)$ - leading to an average of 90.63% games won. Player $B_1(1)$ is shown to be the weakest of the team B players, winning only 62.50% of the games.

Note that there is not a very strong correlation between the average number of goals scored and the outcome of a game. A high average number of goals does not automatically lead to a win. Player $B_2(4)$, the best performing player, maintained an average of only 2.093 goals scored per game while winning 90.63% of the games. The only player with a lower number of iterations per goal was $A_2(2)$, which also lost all its games.

Also note the influence that player $B_2(3)$ had on the teams in which it played. All these games had the lowest draw percentages, but not the best performance. These team compositions resulted in quite poor performance when evaluating the percentage of games won. As previously noted in the visual analysis, this player had learned a very specific move - to intercept a sideways kick and perform an angled goal kick. In cases where it did not have the ball, it simply moved towards the opposition goal on the side of the field. The measured bad performance of this player emphasises the importance of both players in a team staying in the game for the best possible result - once one player is taken out of the game, the combined performance decreases visibly. This also demonstrates the importance of defensive play as the low draw percentage is simply due to the opposition team's scoring goals more easily. Remember that a draw is more desirable than a loss result. It is also illustrated that the trained players are able to exploit even a small weakness in their opposition. In this case, instead of drawing more games, they succeed in scoring more goals against their weak opposition.

5 Findings and Conclusions

This paper proposes a new competitive coevolutionary team-based particle swarm optimisation (CCPSO(t)) algorithm to train multi-agent teams from zero knowledge. Initial performance analysis of the CCPSO(t) algorithm revealed that the particles in the various swarms did not converge onto single solutions. Instead, the swarms maintained a notably high swarm diversity. A behavioural analysis of the global best position revealed that the global best position jumped considerable distances between iterations. This was seen as a sign that the particle swarm might be converging on multiple optimal areas in the search space. In order to verify this hypothesis, the X-means clustering algorithm was applied to the particles. It was revealed that the particles were indeed clustering around certain areas in the search space. The cluster centroids were calculated as candidate players for each position for further investigation.

Each centroid was evaluated as it played in each team permutation against each opposition team permutation. The resulting games were visually inspected to identify the behavioural traits of each player and team. The resulting strategies were discussed in detail. To counter the possible subjectivity of the visual analysis, a quantitative analysis of the performance was also conducted and the results of that analysis were discussed in detail. The results revealed that strong gameplay strategies, based on at least a manner of teamwork, emerged. In addition, it was shown that each centroid represents a different player behaviour that should be treated, and thus scored, individually as a candidate player for the team. The results also revealed that weaknesses in the players are easily exploited by the opposition team, indicating that a good team required not only a good offensive behaviour, but also a good defensive behaviour.

Overall it was shown that the CCPSO(t) training algorithm, along with the *FIFA league ranking* relative fitness function, was capable of training players from zero knowledge to successfully play a complex, non-deterministic game. A weakness in the algorithm was also identified; it was shown that the training algorithm does not converge on a specific playing strategy. Instead, it was shown that the particles in each swarm converged on different playing strategies. It is left as future work to further improve the training algorithm to deal with the multiple playing strategies found by each swarm. Also, to use a niching PSO to promote finding of multiple strategies.

While the goal of this article was mainly to analyse the learned playing strategies and to show that sensible strategies for team-based games can be learned from zero knowledge, future work will also focus on benchmarking against other team-based approaches to simple soccer.

References

Blackwell T (2003) Swarms in Dynamic Environments. In: Proceedings of the 2003 Genetic and Evolutionary Computation Conference, Springer-Verlag, pp 1–12

Blackwell T, Bentley P (2002a) Dont push me! Collision-avoiding swarms. In: Proceedings of the 2002 Congress on Evolutionary Computation, vol 2, pp 1691–1696

Blackwell T, Bentley P (2002b) Dynamic Search with Charged Swarms. In: Proceedings of the 2002 Genetic and Evolutionary Computation Conference, Morgan Kaufmann Publishers, pp 19–26

Conradie J, Engelbrecht A (2006) Training Bao Game-Playing Agents using Coevolutionary Particle Swarm Optimization. In: Proceedings of the 2006 IEEE Symposium on Neural Networks, pp 67–74, DOI 10.1109/CIG.2006.311683

Davis D, Chalabi T, Berbank-Green B (2000) Artificial-life, agents and GO. In: Mohammadian M (ed) New Frontiers in Computational Intelligence and Its Applications, IOS Press, Amsterdam, The Netherlands, pp 125–139

Franken C (2009) Visual Exploration of Algorithm Parameter Space. Proceedings of IEEE Congress on Evolutionary Computation pp 389–398, DOI 10.1109/CEC.2009.4982973

Franken C, Engelbrecht A (2003a) Comparing PSO Structures to Learn the Game of Checkers from Zero Knowledge. In: Proceedings of the 2003 Congress on Evolutionary Computation, pp 234–241

Franken C, Engelbrecht A (2003b) Evolving intelligent game-playing agents. In: Proceedings of the 2003 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists on Enablement through Technology, South

African Institute for Computer Scientists and Information Technologists, pp 102–110

Franken C, Engelbrecht A (2004) PSO Approaches to Coevolve IPD Strategies. In: Proceedings of the 2004 Congress on Evolutionary Computation, vol 1, pp 356–363, DOI 10.1109/CEC.2004.1330879

Haykin S (1998) Neural Networks: A Comprehensive Foundation. Prentice-Hall

Kennedy J, Eberhart R (1995) Particle Swarm Optimization. In: Proceedings of the 1995 IEEE International Conference on Neural Networks, vol 4, pp 1942–1948, DOI 10.1109/TSMCB.2010.2043527

Kennedy J, Eberhart R (2001) Swarm Intelligence. Morgan Kaufmann

Kitano H (1993) Massively Parallel Artificial Intelligence and Grand Challenge AI Applications. Tech. rep., SS-93-04, Association for the Advancement of Artificial Intelligence

Krink T, Vesterstrøm J, Riget J (2002) Particle swarm optimisation with spatial particle extension. Proceedings of the Fourth Congress on Evolutionary Computation 2:1474–1479

Laubach S, Burdick J, Matthies L (1998) An Autonomous Path Planner Implemented on the Rocky 7 Prototype Microrover. In: Proceedings of the 1998 IEEE International Conference on Robotics and Automation, vol 1, pp 292–297

Messerschmidt L, Engelbrecht A (2002) Learning to Play Games using a PSO-based Competitive Learning Approach. Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution and Learning pp 444–448

Pelleg D, Moore A (2000) X-means: Extending K-means with Efficient Estimation of the Number of Clusters. In: Proceedings of the Seventeenth International Conference on Machine Learning, pp 727–734

Rosin C, Belew R (1995) Methods for Competitive Co-evolution: Finding Opponents Worth Beating. In: Proceedings of the Sixth International Conference on Genetic Algorithms, Morgan Kaufmann Publishers, Inc., pp 373–380

Rosin C, Belew R (1997) New Methods for Competitive Coevolution. Evolutionary Computation 5(1):1–29

Scheepers C (2013) Coevolution of Neuro-controllers to Train Multi-Agent Teams from Zero Knowledge. Msc thesis, University of Pretoria, URL http://hdl.handle.net/2263/31625

Scheepers C, Engelbrecht A (2014) Competitive Coevolutionary Training of Simple Soccer Agents from Zero Knowledge. In: Proceedings of the 2014 IEEE Congress on Evolutionary Computation, pp 1–8

van Wyk A, Engelbrecht A (2010) Overfitting by PSO Trained Feedforward Neural Networks. In: Proceedings of the 2010 IEEE Congress on Evolutionary Computation, pp 1–8, DOI 10.1109/CEC.2010.5586333