State Machines Synchronization for Collaborative Behaviors Applied to Centralized Robot Soccer Teams

Jose Guillermo Guarnizo[1](✉) and Martin Mellado[2]

[1] Grupo de Estudio y Desarrollo en robótica, Universidad Santo Tomás,
Bogotá, Colombia
jose.guarnizo@usantotomas.edu.co

[2] Instituto AI2, Universitat Politècnica de València, Valencia, Spain
martin@ai2.upv.es

Abstract. In robot soccer, collaborative behaviors are necessary to establish team coordination. In centralized architectures with global perception, the team coordination is carried out by a making decision system, where the team strategy is programmed out. Finite state machines are an alternative for the making decision systems design in order to assign players roles and behaviors, depending on the game conditions. In this paper a team strategy for robot soccer architectures with global perception and centralized control is proposed, through the use of synchronized state machines for collaborative behaviors among the players by using a synchronization function in some determinate states. This function is used to synchronize one machine state which selects the behavior of one player, with other state which selects the behavior of another player. The synchronization is used, for instance, to coordinate a pass between two players looking for a goal, or blocking an opposite goal by an opposite defender player. Synchronized state machines presented better results than strategies with state machines non-synchronized on different matches played.

Keywords: Multi-agent systems · Robot soccer · Architecture
Finite state machine · Synchronization

1 Introduction

Robot soccer is considered a multi-robot system that includes uncertainties and hostile environment, where robots are working in a coordinated manner on a real challenging problem. Coordination is an important issue for robot soccer, since undesirable behaviors can arise in uncoordinated teams, for example, teammates blocking the ball, obstructing each other or failing to block opposing players [1]. In robot soccer, strategy is defined as the plan of the robot soccer team expecting to win a match, tactic is defined as the organization of the team for the game. Roles are defined as a list of behaviors to perform by the robot. Behaviors are the basic sensorimotor skills of the robot, such as shoot the ball, to block an opponent or to intercept the ball [2].

Robot soccer leagues are divided into two main categories. One is the centralized leagues, using one decision-making body, with global perception and global motion

G. R. Simari et al. (Eds.): IBERAMIA 2018, LNAI 11238, pp. 132–144, 2018.
https://doi.org/10.1007/978-3-030-03928-8_11

control [3]. The other corresponds to the distributed leagues, which use fully autonomous robots, with local perception and autonomous control [4]. Many studies have been published on robot soccer strategies, looking for team coordination. Some other are focused on machine learning, for example using case-based approach for coordinated action selection in distributed robot soccer teams [5]. Or presenting a novel model of reinforcement learning algorithm applied to learning behavior [6]. In [7] is used fuzzy neural networks to task planning and action selection. In [8] is training the decision-making system by Bayesian SOM. Other papers proposed learning machine algorithms using robot soccer strategies for validation purpose [9].

One way to design robot soccer strategies is by using different computational models, such as coordination graphs for role assignment [10]. Finite state machines to switch tactics combined with Petri Net Plans for tasking execution are presented in [11]. Other example of using hierarchical state machines for robot soccer architectures is presented in [12]. Other research uses finite state machines for tactic selection and behavior selection [13]. Collaborative Filtering techniques combined with set-plays are also presented [14]. The use of Hierarchical Finite State Machines is presented in [15]. Synchronization has been used in different robot soccer distributed architectures in order to coordinate autonomous robots [16], focused on communication protocols. Other example of synchronization in computational models for robot soccer architectures is presented in [17], introducing the concept of Multi Robot Plans. In [18] a team strategy is modeled based on the discrete events system theory. In [19] it is proposed dynamic role assignment based on action utility prediction.

In [20] a robot soccer strategy based on tactics, roles and behaviors is presented, which uses a hierarchical state machines. Nevertheless the state machines are based on the game conditions, without a feedback of the behaviors selected by their teammates. This strategy was used to control the operation mode in an islanded ac microgrid [21]. Subsequently, in [22] a robot soccer strategy based on roles was proposed, in this work one role is assigned to each player and a behavior is assigned depending on game conditions, however a behavior feedback selected by their teammates is neither presented. This situation is similar to other strategies related above where synchronization is not presented. In this paper, is proposed a strategy for centralized robot soccer architectures based on a hierarchical finite state machine for tactic selection, dynamic roles assignment and behaviors selection. The behavior selection layer is performed by synchronized state machines, for the purpose of coordinate collaborative behaviors between players. In this case, the behavior selected by one player depends on a function activated by a state of its teammate. This model is designed for a robot soccer team with two-wheeled robots, with a central computer used for computational processes. This strategy is tested using the SimuroSot 5 vs. 5 league simulation platform, and programmed using C++. The proposed strategy has different advantages such as easy programming and low computational costs, and is scalable and easily adaptable to changes in the decision-making system.

2 Hierarchical Team Architecture Using Synchronization Functions

In centralized architecture there is a top-view vision system, which provides full global knowledge of the game field, showing the position of the players and the ball. The image processing is performed by the host computer. Coordinates of each robot and the ball are supplied to the decision-making body which selects the tactic, roles and behaviors. Once behaviors are selected, the decision-making body calculates and transmits the signals control for each robot. State machines have often been used in the robot soccer domain, as an alternative for team architectures. In this paper is proposed a decision-making system based on hierarchical state machine, which is divided into levels. The highest level is presented in Fig. 1(a), where a hierarchical finite state machine (HFSM) selects a tactic from a set of n predefined tactics $T = \{t1,t2,...,tn\}$, using environmental conditions *Ev* as transitions.

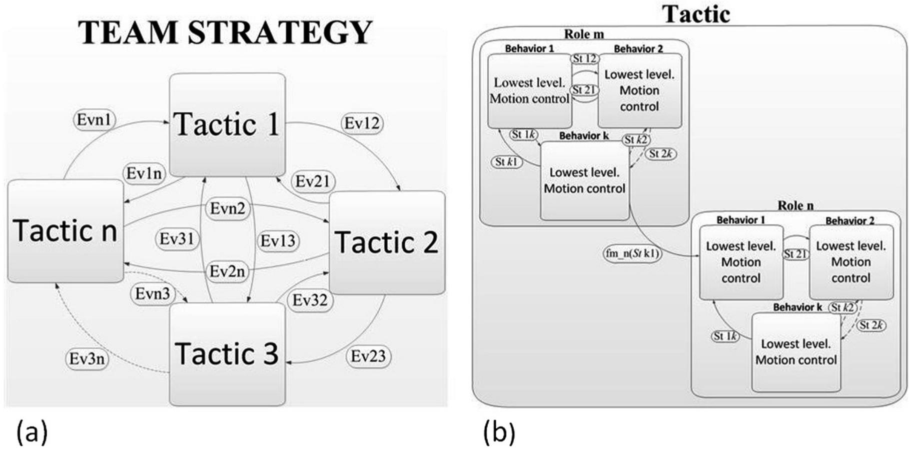

Fig. 1. (a). Highest level of HFSM. (b). Behavior selection using synchronization function.

Once the tactic has been selected, the next layers are activated. At this level one role is assigned to each player. Players with roles perform specific behaviors controlled by a lower layer of the state machine, using environmental conditions *St* as transitions. To carry out the behaviors, the lowest layer, namely the motion control layer is activated. Once the tactic has been selected, it is assigned a set of roles $R = \{r1,r2,\ldots,rm\}$ to a finite number of agents $A = \{a1,a2,\ldots,am\}$, such that $i \neq j \rightarrow Ri \neq Rj$. Note that there is the same number of agents and roles. One role has been assigned to each agent, without repeating roles. There is a set of behaviors $B = \{b1,b2,\ldots,bk\}$, which will be performed by the agents. Each role *r* has been associated to one or more behaviors that will be executed by agent *a* in the game, according to game conditions. There is a set of characteristics $G = \{g1,g2,\ldots,gp\}$ that describes the changes in the game field. The highest level of the HFSM uses set *G* to select *T*, and then is used set *G* to assign *R* to

A. Subsequently, the second layer of the HFSM selects one behavior bk to be executed by each a. Control of the robot motion is implemented in the lowest layer of the HFSM. The procedure is performed as follows:

ALGORITHM 1 DECISION SYSTEM OF THE STRATEGY

1. *Hierarchical state machine* **reads** set G and **assigns** tactic t.
2. Given t selected in step 1, C **assigns** one role r to each agent a, ensuring that complete set R is assigned to complete set A.
3. *Hierarchical state machine* **reads** set G and assigns behaviors b to each agent a, depending on the role r assigned in step 2, and set G.
4. The set of behaviors B are controlled by robot motion control, and this is carried out by the lowest level of the hierarchical state machine.
5. On the soccer field, set A **executes** behaviors associated with set R, which was assigned in step 2, modifying set G.
6. When conditions are triggered, **repeat** step 1.

It is proposed synchronization functions to feedback the state of the other player to synchronize collaborative behaviors between players. As is shown in Fig. 1(b), the player with the role m triggers the behavior k when the conditions $St2k$ or $ST1k$ are triggered. The player with the role n may trigger the behavior 1 when the condition $St1\ k$ is activated and the agent with the role m plays its respective behavior k and the condition of the role m triggers the function $fm\ n(Stk1)$. This happens when the player with the role m players its behavior k. This strategy is implemented in a robot soccer team with five wheeled robots. One is the goalkeeper, which is the only constant role.

Two parameters have been chosen for tactic selection. The first one refers to ball localization on the game field. For this purpose the field is divided into three different zones defined as follows: The defensive zone is the zone where the goalkeeper of the own team is located. The middle zone is in the half of the playing field, and the offensive zone is where the opponent's goal post is located. Ball zone corresponds to the zone where the ball is located. It is defined the ball possession as the minimal distance between a robot and the ball, where the robot can shoot the ball. This distance is obtained experimentally as an appropriate distance at which a robot can shoot the ball.

Four tactics are defined as follows:

- Defensive defense tactic: This is selected when the ball is located in the defensive zone or the middle zone, and one's own-team does not keep ball possession.
- Defensive attack tactic: This is selected when the ball is located in the defensive zone or the middle zone, and own-team keeps ball possession.
- Offensive defense tactic: This is selected when the ball is located in the offensive zone, and own-team does not keep ball possession.
- Offensive attack tactic: This is selected when the ball is presented in the offensive zone, and own-team keeps ball possession.

Four tactics correspond to states, whereas the environmental conditions (ball zone, and ball possession), correspond to the transitions. For role assignment, there are five players per team. One of them is the goal-keeper ap, which keeps a constant role. Thus, there is a finite number of agents $A = a1;\ a2;\ a3;\ a4$ that perform a set of roles

R = *rad; rpd; rs; ra*, where *rad* corresponds to active defender, *rpd* corresponds to passive defender, *rs* is supporter, and *ra* is attacker. The decision-making body executes the algorithm to assign roles, depending on the tactic previously selected. It is important to highlight that the role assignments is similar between defensive tactics (defensive defense and defensive attack tactics), and offensive tactics (offensive defense and offensive attack tactics).

For defensive tactics, the role assignment is reaches by both defensive defense and defensive attack tactics. Once each tactic is activated, the role selection algorithm is carries out. In these tactics, the player closest to the ball becomes the active defender *rad*, and this player attempts to take the ball. The second nearest player to the ball becomes the passive defender *rpd*, which blocks an opponent player. The third nearest player to the ball is the attacker *ra*, which goes to the offensive zone, waiting for the ball. The last player becomes the supporter *rs*, which goes to offensive zone.

For the offensive tactics, the role assignment is reaches by both offensive defense and offensive attack tactics. Similarly, once each tactic is activated, the role selection algorithm is carries out again. The player nearest to the ball becomes the attacker *ra*, and this player attempts to take possession of the ball, in order to score a goal. The second nearest player to the ball becomes the supporter *rs*, which supports the attacker. The third nearest player to the ball is the active defender *rad*, which is in the middle zone in case of counter attack. The last one is the passive defender *rpd*, which supports the active defense.

2.1 Behavior Selection

Many roles were implemented to execute different behaviors, depending on the tactic chosen. Goalkeeper is the only static role in this strategy. The behaviors of the goalkeeper are presented in Fig. 2. There is a goalkeeper zone, which is an area in front of the own-team's goal post. When goalkeeper behavior is triggered, the player checks its location. If it is within the goalkeeper zone, the robot localizes and follows the ball in a line running parallel to the goal line. If the player is not located in the goalkeeper zone, the robot must become properly located and subsequently checks its location and continues playing.

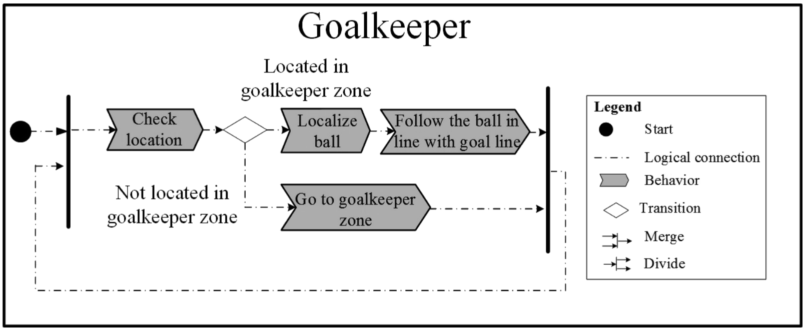

Fig. 2. Goalkeeper behaviors.

For the tactic defensive defense, the active defender role goes by the ball as is shown in the Fig. 3(a). Firstly, the ball is located. If the ball is in the goalkeeper zone, the robot follows the ball outside, to avoid own-goals or block the goalkeeper. If the ball is outside of the goalkeeper zone, the active defender goes to catch the ball. The Fig. 3(b) shows the behaviors in the tactic defensive attack. Once the tactic is activated the ball is located. If the ball is inside the goalkeeper zone, then the active defender follows the ball outside. If the ball is not located in the goalkeeper zone, then the active defender checks the ball possession, if it has ball possession, the robot shoots the ball to offensive zone. If it has not the ball possession, the active defender tries to catch the ball. In the Fig. 3(c) is presented the behaviors of the active defender for both offensive tactics. There is an active defender zone, which corresponds to a zone behind the half-way line. When the tactics are activated, the active defender checks its location, if it is not inside the active defender zone, the robot goes to its zone. If the robot is located in its zone, the player follows the ball in line with the half-way line.

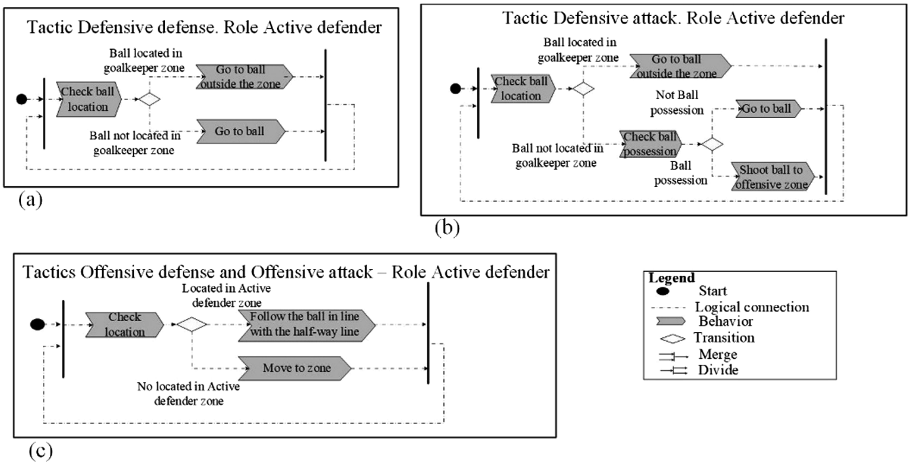
Tactic Defensive defense. Role Active defender
Ball located in goalkeeper zone
Go to ball outside the zone
Check ball location
Ball not located in goalkeeper zone
Go to ball
(a)
Tactic Defensive attack. Role Active defender
Ball located in goalkeeper zone
Go to ball outside the zone
Check ball location
Not Ball possession
Go to ball
Ball not located in goalkeeper zone
Check ball possession
Ball possession
Shoot ball to offensive zone
(b)
Tactics Offensive defense and Offensive attack – Role Active defender
Located in Active defender zone
Check location
Follow the ball in line with the half-way line
Move to zone
No located in Active defender zone
(c)
Legend
Start
Logical connection
Behavior
Transition
Merge
Divide

Fig. 3. Active defender behaviors

The behaviors of the role passive defender for both defensive tactics are shown in the Fig. 4(a). In these tactics, the passive defender checks its nearest opponent and goes to block without touch the player, in order to prevent possible opponent passes avoiding collisions with the player. In the Fig. 4(b) are shown the behaviors of the passive defender for both offensive tactics. For this purpose there is a passive defender zone, behind the active defender zone, where the passive defender covers the other side that the active defender blocks. Thus, when the offensive tactics are activated, the passive defender check its location, if the robot is not located in its respective zone, it goes to its zone. If the robot is located in its respective zone, the player covers in line with the half-way line the other side where the ball is located.

For the defensive tactics, the behaviors of the supporter and attacker are presented in the Fig. 5. In the case of the attacker (Fig. 5(a)), there is an attacker zone in the

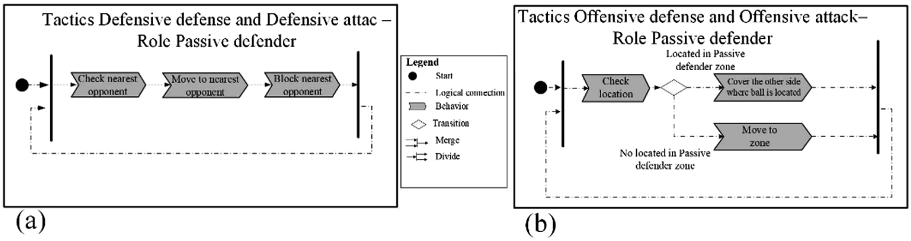
Tactics Defensive defense and Defensive attac – Role Passive defender
Check nearest opponent
Move to nearest opponent
Block nearest opponent
Legend
Start
Logical connection
Behavior
Transition
Merge
Divide
Tactics Offensive defense and Offensive attack– Role Passive defender
Located in Passive defender zone
Check location
Cover the other side where ball is located
Move to zone
No located in Passive defender zone
(a)
(b)

Fig. 4. Behaviors of passive defender.

offensive zone, where the robot follows the ball in line with the half-way line, waiting for the ball. Thus, when the defensive tactics are activated, coordinates of attacker are checked, if the robot is in the attacker zone, the robot follows the ball in line. If the attacker is not properly located, the robot goes to its zone. In the Fig. 5(b), it is presented the behaviors for the supporter. It is defined the supporter zone, located in front of the attacker zone, where supporter covers the other side where attacker blocks. If the supporter is inside its zone, the player covers the zone in line with the half-way line. If not, the supporter goes to its zone.

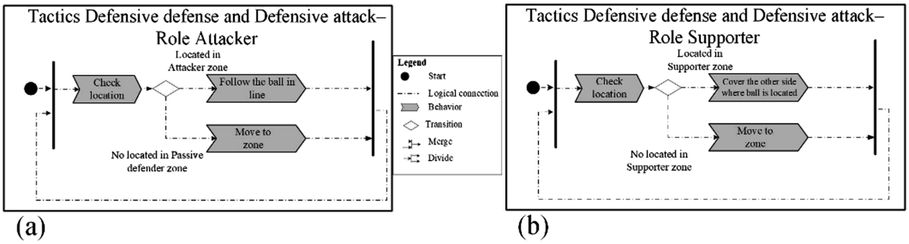
Tactics Defensive defense and Defensive attack– Role Attacker
Located in Attacker zone
Check location
Follow the ball in line
Move to zone
No located in Passive defender zone
Legend
Start
Logical connection
Behavior
Transition
Merge
Divide
Tactics Defensive defense and Defensive attack– Role Supporter
Located in Supporter zone
Check location
Cover the other side where ball is located
Move to zone
No located in Supporter zone
(a)
(b)

Fig. 5. Behaviors of the supporter and attacker in defensive tactics.

2.2 Synchronization Functions in Coordinate Attack Behaviors

For the tactic offensive attack, the main purpose of the attacker is to shoot the ball to opponent's goal post, seeking score a goal. As is shown in the Fig. 6(a), ball possession is checked in the attacker, if the attacker not has the ball possession, player goes to catch the ball. If the robot has the ball possession, it is checked if there is an opponent player different from the goalkeeper, which blocks the trajectory to opponent's goal post. If the trajectory is blocked, attacker activates a synchronization function to supporter, in order to coordinate a ball pass. In the Fig. 6(b), is presented the behaviors of the supporter. Initially, the supporter checks if the attacker activates the synchronization function. If there is not activated, supporter works as is described in the Fig. 5 (b). If the synchronization function is activated, the supporter receives a ball pass of the attacker, when the tactic changes and new role assignment is reached, supporter becomes the new attacker.

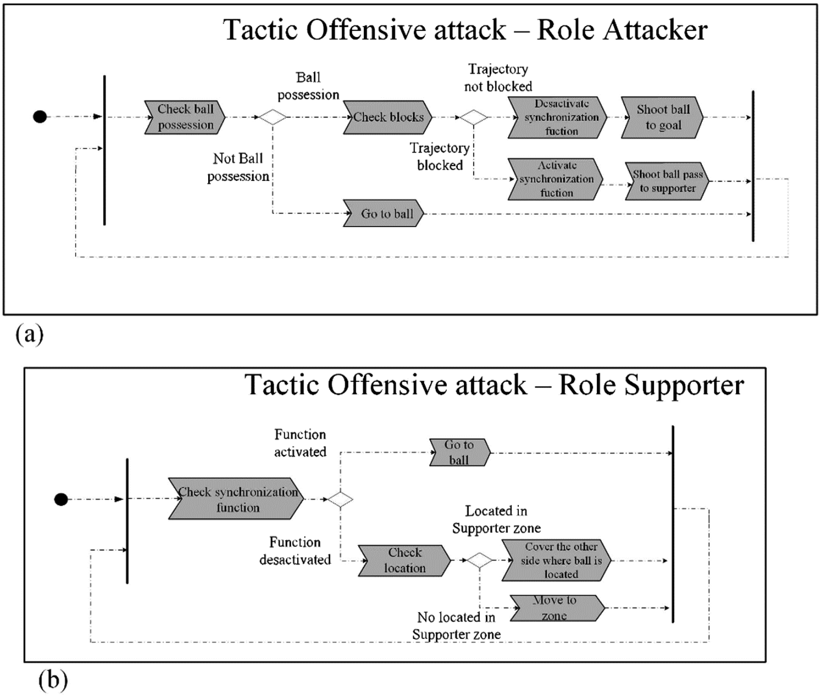
Tactic Offensive attack – Role Attacker
Check ball possession
Ball possession
Check blocks
Trajectory not blocked
Desactivate synchronization fuction
Shoot ball to goal
Not Ball possession
Trajectory blocked
Activate synchronization fuction
Shoot ball pass to supporter
Go to ball
(a)
Tactic Offensive attack – Role Supporter
Function activated
Go to ball
Check synchronization function
Located in Supporter zone
Function desactivated
Check location
Cover the other side where ball is located
Move to zone
No located in Supporter zone
(b)

Fig. 6. Behaviors of the supporter and attacker in offensive attack tactic

In the Fig. 7, are shown the different zones presented to perform the different roles (goalkeeper zone, active defender zone, passive defender zone, attacker zone, supporter zone and lateral zones).

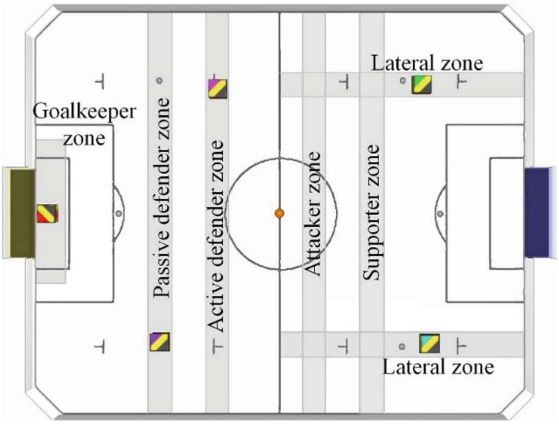

Fig. 7. Zones in game field.

3 Simulation and Results

In order to test and validate the proposed model, the strategy is implemented in FIRA's SimuroSot and programing in C++. In order to validate the performance of the proposed strategy, there were carried out 100 games against a team strategy provided by default in the simulator and programming in LINGO (hereinafter called LINGO strategy). LINGO strategy consists of five constant roles which are the goalkeeper, two defenders that blocking the ball in parallel to the goal line, and two attackers which go to the ball and shoot to their opponent's goal. All games were conducted applying the rules of FIRA SimuroSot 5 vs. 5 League.

All games were won by the strategy with synchronization, in many cases with ample goal difference. In Table 1 is presented the comparison of the different results obtained in the 100 games by each team. The average of goals scored, the average of ball possession and the average of successful passes is significantly higher in the team with the proposed strategy. This is because a greater number of collaborative behaviors, due to the synchronization between attacker and supporter. It should also be noted that in all games the ball remained in the offensive zone in the many of cases. The variations of goals scored, successful passes or ball possession observed in different games are due to an incremental odometry error induced by the simulator. However, it shown that team with the strategy proposed presented a better adaptation to game conditions than LINGO strategy. In all games was observed that team with LINGO strategy shown different behaviors that produce faults or own-goals.

Table 1. Comparison of the parameters of 100 games against team with LINGO strategy

Parameters		Local	Opponent
Goals average		5.66	1.8
Standard deviation		2.49	1.53
Ball possession average		64.87%	35.13%
Standard deviation		5.98%	5.98%
Successful passes average		6.56	1.12
Standard deviation		2.14	0.88
	Defensive zone	Middle zone	Offensive zone
Ball location	27.53%	12.30%	60.17%
Standard deviation	7.09%	3.06%	6.89%

In Table 2 is presented a comparison among different results obtained in games between different teams against LINGO strategy. The comparison includes the results previously reported of the team with the proposed strategy, and the results obtained by other strategies against LINGO strategy reported in [9, 15, 18–20, 22]. As can be seen the team with the proposed strategy presented a higher average of goals by game than the strategies previously published in [18–20, 22], the goal difference is higher than reported in [15], the only strategy which presented considerably higher average of goals is the strategy presented in [9] however, this work is focused on machine learning. Strategies which reported ball possession showed similar percentages. N.R. corresponds to data not reported. Goals av., refers to goals average. Ball poss., refers to ball possession. Standard deviation is similar in the cases when it was reported is similar (excepting [20], where were only performed 10 games), this is because an incremental error of odometry induced by the simulator, in order to perform more realistic simulations.

Table 2. Comparison of results reported of other strategies against team with LINGO strategy

Teams	Games played	Games			Goals average		Ball possession	
		Won	Tied	Lost	Local	Opp.	Local	Opp.
Sync.	100	100%	0	0	5.88	1.8	64.87%	35.13%
					S.D.	**S.D.**	**S.D.**	**S.D.**
					2.49	1.53	5.98%	5.98%
[20]	10	100%	0	0	3	1	72.3%	27.7%
					S.D.	**S.D.**	**S.D.**	**S.D.**
					0.82	0.82	7.09%	7.09%
[18]	10	70%	10%	20%	5.5	4.2	N.R	N.R
					S.D.	**S.D.**	**S.D.**	**S.D.**
					2.01	2.53	N.R	N.R
[19]	10	100%	0	0	5.1	3.7	71.7%	28.3%
					S.D.	**S.D.**	**S.D.**	**S.D.**
					N.R	N.R	N.R	N.R
[15]	50	100%	0	0	6.1	2.8	67%	37%
					S.D.	**S.D.**	**S.D.**	**S.D.**
					N.R	N.R	N.R	N.R
[9]	50	100%	0	0	9.74	1.78	N.R	N.R
					S.D.	**S.D.**	**S.D.**	**S.D.**
					2.21	1.22	N.R	N.R
[22]	100	100%	0	0	5.29	2.17	68.3%	31.7%
					S.D.	**S.D.**	**S.D.**	**S.D.**
					1.93	1.33	8.3%	8.3%

In the Fig. 8 is presented an example of synchronization of behaviors between attacker and supporter, including a change in the roles. In the Fig. 8(a) the player 2 (attacker) detects that an opponent player blocks the trajectory of the ball to the goal, player 1 (supporter) accompanies. In the Fig. 8(b) player 2 activates the synchronization function and sends the ball to player 1, who, in turn goes to catch the ball. When player 2 do not have ball possession a new tactic is selected and new roles are assigned, player 1 becomes attacker and player 2 becomes supporter. In the Fig. 8(c) the new attacker catches the ball and try to score a goal, meanwhile player 2 supports.

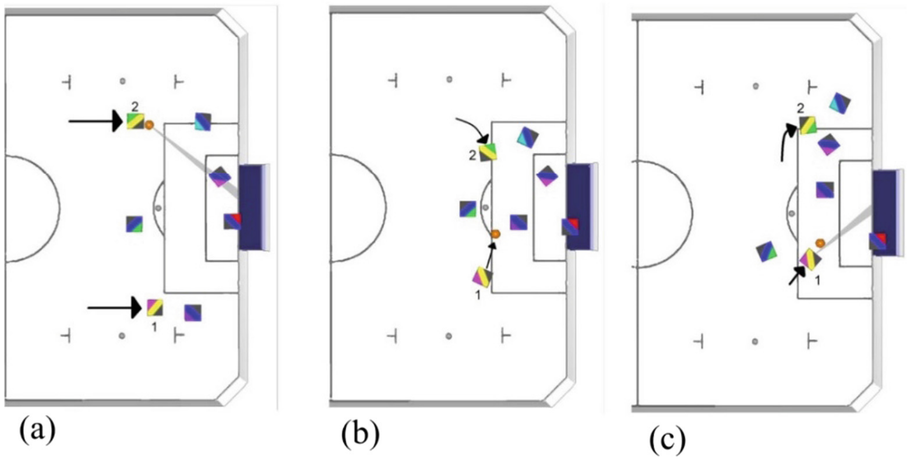

Fig. 8. Example when an opponent player blocks the goal, with synchronization function.

4 Conclusions and Future Works

Finite state machines for multi-agent coordination in centralized robot soccer architectures allow an intuitive design of the strategy keeping in mind the league rules. Given that robot soccer is a multi-agent system, coordination among player is necessary and given that, synchronization function merge as alternative to improve the performance of collaborative behaviors such as successful passes. The proposed strategy was compared with other reported strategies that played games against a basic strategy provided by the simulator. The strategy proposed showed better results that those strategies, excepting one case where machine learning where implemented. Even though the use of computational models as finite state machines have displayed satisfactory results in coordination in robot soccer, the use of synchronization functions among players allow to obtain better results in cooperative behaviors that other similar strategies that not use synchronization functions obtaining, for example, more goals scored, more number of successful passes or more percentage of ball possession. The Synchronization functions usage do not represent any increasing in strategy complexity, neither a relevant increasing in source code or computation cost. Likewise these synchronization functions can be also applicable to other architectures, such as distributed architectures or using other computational models.

In future works will be proposed synchronization function implemented in distributed strategies seeking better results in cooperative behaviors. As well nondeterministic finite state machines and probabilistic finite state machines to design coordination strategies and the making decision systems in robot soccer leagues.

Acknowledgements. This work has been funded by “Decimosegunda convocatoria interna para el fomento de la investigación - FODEIN 2018” at Universidad Santo Tomás, Bogotá Colombia, entlited “Localización y mapeo por medio de redes inalámbricas de datos aplicado a roots móviles colaborativos”, project code: 1836001.

References

1. Stulp, F., Utz, H., Isik, M.: Implicit coordination with shared belief: a heterogeneous robot soccer team case study. Adv. Robotics **24**(7), 1017–1036 (2010)
2. Drogoul, A., Collinot, A.: Applying an agent-oriented methodology to the design of artificial organizations: a case study in robotic soccer. Auton Agent Multi-Ag **1**(1), 113–129 (1998). https://doi.org/10.1023/A:101009862392
3. Stone, P., Veloso, M.: A layered approach to learning client behaviors in the robocup soccer server. Appl. Artif. Intell. **12**, 165–188 (1998)
4. Marsella, S., Tambe, M., Adibi, J., Al-onaizan, Y., Kaminka, G., Muslea, I.: Experiences acquired in the design of RoboCup teams: a comparison of two fielded teams. Auton. Agent Multi-Ag. **4**(1), 115–129 (2001). https://doi.org/10.1023/A:1010027016147
5. Ros, R., Arcos, J.L., Lopez de Mantaras, R., Veloso, M.: A case-based approach for coordinated action selection in robot soccer. Artif. Intell. **137**(9–10), 1014–1039 (2009)
6. Cunha, J., Serra, R., Lau, N., Lopes, L.S., Neves, A.: Batch reinforcement learning for robotic soccer using the Q-batch update-rule. J. Intell. Robot. Syst. **80**(3), 385–399 (2015). https://doi.org/10.1007/s10846-014-0171-1
7. Jolly, K.G., Sreerama, R., Vijayakumar, R.: Intelligent task planning and action selection of a mobile robot in a multi-agent system through a fuzzy neural network approach. Eng. Appl. Artif. Intel. **23**(6), 923–933 (2010). https://doi.org/10.1016/j.engappai.2010.04.001
8. Chen, B., Zhang, A., Cao, L.: Autonomous intelligent decision-making system based on Bayesian SOM neural network for robot soccer. Neurocomputing **128**, 447–458 (2014)
9. Hwang, K.S., Chen, Y.J., Lee, C.H.: Reinforcement learning in strategy selection for a coordinated multirobot system. IEEE Trans. Syst. Man. Cybern. A Syst. Hum. **37**(6), 1151–1157 (2007)
10. Wang, J., Wang, T., Wang, X., Meng, X.: Multi-robot decision making based on coordination graphs. In: Mechatronics and Automation International Conference on, pp. 2393–2398 (2009)
11. Kontes, G., Lagoudakis, M.G.: Coordinated team play in the four-legged robocup league. In: 19th IEEE International Conference on Tools with Artificial Intelligence, pp. 109–116 (2007)
12. Guarnizo, J.G., Mellado, M.: Robot soccer strategy based on hierarchical finite state machine to centralized architectures. IEEE Lat. Am. Trans. **14**(8), 3586–3596 (2016)
13. Lou, Y., Chen, B., Shi, H.: Decision making model based on state assessment and hierarchical FSM in robot soccer. In: Automatic Control and Artificial Intelligence (ACAI 2012)
14. Abreu, P.H., Castro, D., Almeida, F., Mendes-Moreira, J.: Improving a simulated soccer team’s performance through a memory-based collaborative filtering approach. Appl. Soft Comput. **23**, 180–193 (2014). https://doi.org/10.5772/56189
15. Lou, Y., Chen, B., Shi, H. Decision making model based on state assessment and hierarchical FSM in robot soccer. In: International Conference on Automatic Control and Artificial Intelligence (ACAI 2012), Xiamen, pp. 756–759 (2012)
16. Stone, P., Veloso, M.: Task decomposition, dynamic role assignment, and low bandwidth communication for real-time strategic teamwork. Artif. Intell. **110**(2), 241–273 (1999)
17. Palamar, P.F., Ziparo, V.A., Iocchi, L., Nardi, D., Lima, P.: Teamwork design based on petri net plans. In: Robot Soccer World Cup XII, RoboCup 2008, pp. 200–211 (2008)
18. Cardoso, P., Molina, L., Freire E.O., Carvalho, E.A.N.: A methodology to designing strategies for robot soccer based on discrete event systems formalism. In: Brazilian Robotics Symposium and Latin American Robotics Symposium, Fortaleza, pp. 143–149 (2012)

19. Yang, M., Jia, Y.: Action utility prediction and role task allocation in robot soccer system. In: 12th International Conference on IEEE Control Automation Robotics and Vision (ICARCV), Guangzhou, pp. 112–117 (2012)
20. Guarnizo, J.G., Mellado, M., Low, C.-L., Blanes, F.: Architecting centralized coordination of soccer robots based on principle solution. Adv. Robot. **29**(15), 989–1004 (2015)
21. Diaz, N.L., Guarnizo, J.G., Mellado, M., Vasquez, J.C., Guerrero, J.M.: A robot-soccer-coordination inspired control architecture applied to islanded microgrids. IEEE Trans. Power Electron. **32**(4), 2728–2742 (2017)
22. Guarnizo, J.G., Mellado, M.: Centralized robot soccer architecture based on roles. RIAI **13** (3), 370–380 (2016). https://doi.org/10.1016/j.riai.2016.05.005