CYRUS Soccer Simulation 2D Team Description Paper 2021

Nader Zare[1], Aref Sayareh[3], Mahtab Sarvmaili[1], Omid Amini[4], Amílcar Soares[1], and Stan Matwin[1 2]

[1] Institute for Big Data Analytics, Dalhousie University, Halifax
[2] Institute for Computer Science, Polish Academy of Sciences, Warsaw
[3] Shiraz University, Iran
[4] Qom University of Technology, Iran
{nader.zare, mahtab.sarvmaili, amilcar.soares}@dal.com
stan@cs.dal
{arefsayareh, omidamini}@gmail.com

Abstract. In this report, we briefly present the technical procedure and simulation steps for the 2D soccer simulation of team Cyrus. We emphasize on this document on how the prediction of teammates' behavior is performed. In our proposed method, the agent receives the noisy inputs from the server, and predicts the ball holder full state behavior. Taking advantage of this approach for choosing the optimal view angle shows 11.30% improvement on the expected win rate.

Keywords: RoboCup · Soccer Simulation 2D · Behavior Predictor.

1 Introduction

The idea of robotic soccer games was proposed as a novel research topic in 1992, and since then the RoboCup has been considered as an annual international competition for developing new ideas in A.I. and robotics. This competition is formed of various leagues such as Rescue[24,25,26,27,28], Soccer Simulation[29] and Standard Platform[30] leagues.

Cyrus Team is one of the soccer simulation team in the 2D Soccer Simulation league. This team was established in 2013, and it has engaged in RoboCup and IranOpen competitions since then. It is worth mentioning that this team has gained the second, third, fourth, and fifth places in RoboCup 2018, 2019, 2017, 2014 years respectively. Also, Cyrus won first place in IranOpen 2018 and 2014, RoboCup Asia-Pacific 2018, and second place in JapanOpen 2020 competitions. The Cyrus's team base is agent2d[1].

1.1 Previous Work

In the recent years we have concentrated on exploiting artificial intelligence and machine learning techniques to improve the functionality of Cyrus team [16,17,18,19]. Among these works, we can mention the improvement of agents' defense decision-making process using Reinforcement Learning (RL)[23], prediction of an opponent's behavior, and optimization of the shoot skill. Helios has developed an algorithm for the analysis of the agents' offensive behavior [3,4,2]. Fractals, 2019 which is partially based on Gliders2d used elements of evolutionary computation, within the framework of Guided Self-Organisation [5]. FRA-United has researched on the commutation of agents in games [6,7,8]. FCP_GPR teams has developed a framework for the free-kick [9], while the Namira has implemented a python-based application for the analysis of soccer simulation games[10,11,12]. Razi has worked on scoring the offensive behavior in the 2D soccer simulation[13,14].

1.2 Release

Cyrus 2014 Source As a part of our contribution to the development of the 2D Soccer Simulation league, we have released the Cyrus 2014 [16] source code to encourage new teams to participate in the competitions. Cyrus 2014 won the 1st and 5th places in the Iran-Open RoboCup Competition and International RoboCup Competition, respectively. The source code can be found in our github[1].

Starter Agent and Starter Librcsc Cyrus team members - in cooperation with IranOpen technical committee of 2D soccer simulation league - have designed a simplified version of the agent base [1] and the librcsc library for the 2D soccer simulation starter league. High-level behaviors like passing, dribbling, and shooting have been omitted from this base. This version of 2D soccer simulation base and librcsc - specifically are designed for junior students - have been exploited in 2D soccer simulation starter league during both IranOpen RoboCup 2018, IranOpen RoboCup 2020 and RoboCup Asia-Pacific 2018. More than ten teams participated in each of the competitions, with more than fifty participants in total. All of the participants have used the this base developed by Cyrus and IranOpen committee of 2D soccer simulation league. The base can be found in our github[2] [3].

CppDNN The C++ Deep Neural Network (CppDNN) library has been developed by Cyrus team members to facilitate the implementation of Deep Neural Network in the 2D Soccer Simulation environment. This library stores the trained weights of a neural network which has been trained by Keras library. The developed script within this library transforms the trained weights of a deep neural network into a text file. Subsequently, it loads the trained weights to recreate the original deep neural network in C++. This library employs Eigen Library for its calculation. The library can be found in our github[4].

[1] Cyrus 2014 Source `https://github.com/naderzare/cyrus2014`.
[2] Starter Agent 2D `https://github.com/naderzare/StarterAgent2D`
[3] Starter LibRCSC `https://github.com/naderzare/StarterLibRCSC`
[4] CppDNN Source Code `https://github.com/Cyrus2D/CppDNN`

Pyrus - Python Soccer Simulation 2D Base Most of 2D soccer simulation teams exploit the Helios [1], Gliders2d [20], WrightEagle [21] or Oxsy [22] base. All of these bases have been developed in C++. Although those have shown fast processing and execution time, developing machine learning algorithms will be a challenging and time-consuming process. Due to the fast growth of Python language popularity among students and scientist, and its strength for implementing machine learning algorithms, Cyrus team members have started developing an open source python base for 2D soccer simulation league. This base is currently available in Cyrus github[5] and it will support all features of current 2D soccer simulation server in the Full-State mode in the near future.

2 Kick Behavior Predictor

One of the main goals of 2D Soccer team is increasing the winning chance, and it can be achieved by enhancing the general performance of the team. This objective can be interpreted as increasing the team's number of goals and reducing the number of goals against the team. Enhancing the functionality of the team's results in a better performance in the field. However, random noises on the observation of the agents from the environment are the major challenge the agents face while they want to choose their actions. The 2D soccer environments exert the random noises on the observation of agents from the environment to simulate the real-world soccer match; however, these noises complicate the agents' decision-making process. The soccer simulation server provides an option known as ***"full-state mode"*** to eliminate the random noises from the the agents' observation. If the server runs with the ***"full-state mode"***, it distributes the pure state of the game to the teams. In order to understand the impact of noise on the functionality of teams, we tested the Cyrus against Helios 2019[3] with two different settings for the simulation server. In the first, server was run with its default settings. In the second, the server was run with the full-state mode. This phase was divided into two sub-experiments. Cyrus receives the full-state of the game from the server and uses it in two different fashions: 1 - full-state observation: the agents exert the pure observation of the system for their decision-making; 2 - full-state chain action: the pure observations are only exploited for the chain action of agents, and the noisy world model was used for the rest of processing. These three operation modes have been tested 500 times, and the experimental results are reported in the following section. The distribution of goal for and goal against for these experiments are shown in Fig. 1. Also, the win rate, expected win rate, and average score are denoted in Table1. The results of these experiments prove the extreme effect of noisy data on the functionality of the teams. In order to tackle this problem, many team are exploiting opponent behavior prediction or noise reduction algorithms. In this TDP, we aim to address this challenge by enabling agents to predicts their full-state behavior using the noisy observations and exploiting this prediction for the optimization of their behavior. Correspondingly, the server ran in the fullstate mode, and it passed the word

[5] Pyrus Base Source Code `https://github.com/Cyrus2D/Pyrus`

Model (WM) and the Fullstate World Model (FWM) to the agents. At this point the WM and FWM will be received by the agent for the further processing. The agent passes the FWM and WM to the Kick Decision-Making module, and it only passes the WM to Move Decision-Making module. If the ball is not within the kickable area of the agent, move-decision module chooses behavior of agent and it sends the action to the server, otherwise kick-decision making module sends the WM and FWM to Data Extractor module and Chain action module respectively. Rhe Chain Action module employs the FWM to choose optimal action, then afterward, it sends the action and to Data Extractor module and server.

The Data Extractor Module receives the WM and action and it attempts to generate the Data Set using its submodules (Feature Extractor and Label Generator). Feature extractor is a part of the data extractor module to select the important features (will be discussed in the next subsection) from the received data. Label Generator takes the action of the agent from Chain Action module and generates the fullstate action label for this data. The structure of agent and its processing modules are presented in the Fig. 2.

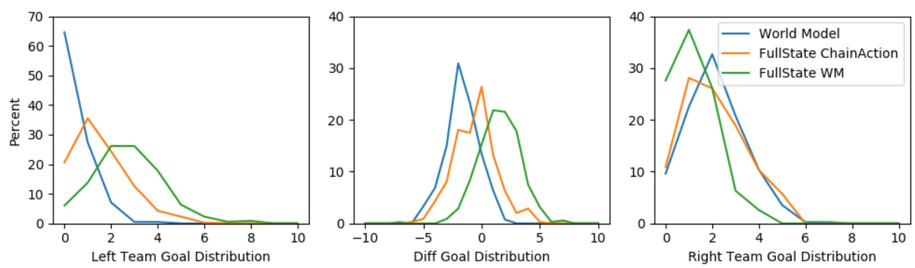

Fig. 1. The distribution of goal for and goal against between Cyrus and Helios in three different modes: a. WM b. FWM and c. fullstate chain action .

Table 1. The win rate, expected win rate, and average score

Run Type	Win Rate	Expected Win Rate	Cyrus Average Goal	Helios2019 Average Goal
Normal	7.09	8.19	0.45	2.13
Chain Action Full State	24.64	33.46	1.59	2.09
Full State	72.7	85.76	2.72	1.19

2.1 Feature Extractor

As we mentioned in Section 2, the feature extractor module receives the WM, and it extracts the most significant attributes of the input data. The related features of the ball, players, and others are denoted in Tables 2 3 respectively.

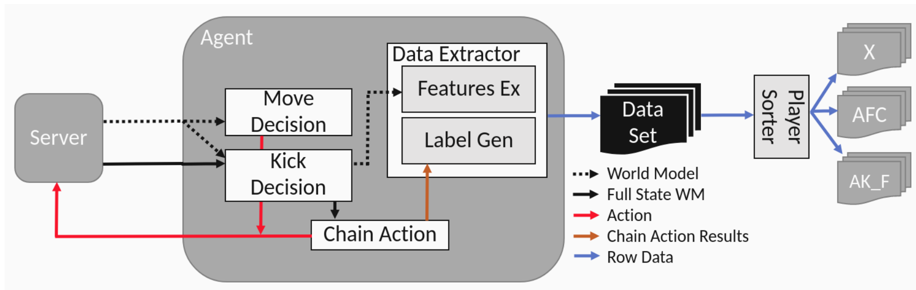

Fig. 2. The internal structure of agent and its processing modules

Table 2. List of Ball Features and Other

Feature Class	Feature Name	Description
Ball_Position	Ball_X	Ball Position - X
Ball_Position	Ball_Y	Ball Position - Y
Ball_Position	Ball_RX	Distance to Holder Player - X
Ball_Position	Ball_RY	Distance to Holder Player - Y
Ball_Position	Ball_R	Euclidean Distance from Ball to Holder Player
Ball_Position	Ball_Teta	Angle From Holder Player to Ball
Ball_Velocity	Ball_VX	Ball Velocity - X
Ball_Velocity	Ball_VY	Ball Velocity - Y
Ball_Velocity	Ball_VR	Ball Velocity - Length
Ball_Velocity	Ball_VTeta	Ball Velocity - Angle
Dribble	Dribble_Free_Distance	Distance of ball to the nearest opponent in 12 sector
Other	Cycle	Cycles of the game
Other	Offside count	The accuracy count for the offside line

2.2 Features Sorting Methods

In our proposed method, we take advantage of a deep neural network for the prediction of the agents' behavior using the noisy observations. We've generated 10 different datasets from the world model to examine the effect of the input setting on the prediction of the network. To create each one of this dataset, we used one of the sorting method that is explained in Table 4. Each one of this sorting method changes the order of players' features. To make the process of these sorting methods more clear, the results of them for the players in Fig. 3 are demonstrated in Table 4.

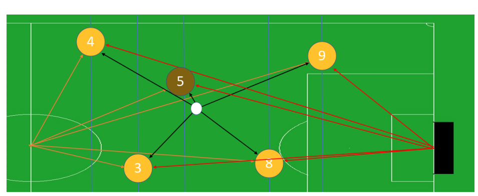

Fig. 3. Sample positions of agents in the field

Table 3. List of player's features

Feature Class	Feature Name	Tm or Opp	Description
Other	Player_Side	Both	Side of player 1 or -1
Other	Player_Unum	Both	Uniform number of player
Other	Player_Body	Both	Body angle
Other	Player_Face	Both	Face angle
Other	Player_Tackling	Both	Is player tackling
Other	Player_Kicking	Both	Is player kicking
Other	Player_Card	Both	Has player yellow card or no
Type	Player_Type_DashRate	Both	Dash Rate of player
Type	Player_Type_EffortMax	Both	Maximum Effort of player
Type	Player_Type_EffortMin	Both	Minimum Effort of player
Type	Player_Type_KickableDist	Both	Kickable Distance of player
Type	Player_Type_MarginDist	Both	Margin Distance of player
Type	Player_Type_KickPowerRate	Both	Kich Power rate of player
Type	Player_Type_Decay	Both	Decay of player
Type	Player_Type_Size	Both	Size of player
Type	Player_Type_SpeedMax	Both	Maximum speed of player
Position	Player_X	Both	Position of player - X
Position	Player_Y	Both	Position of player - Y
Position	Player_RX	Both	Distance to holder player - X
Position	Player_RY	Both	Distance to holder player - Y
Position	Player_R	Both	Distance of player to holder player
Position	Player_Teta	Both	Angle from holder player to player
Position	Player_Offside	Teammate	Player is in offside
Velocity	Player_VX	Both	Velocity of player - X
Velocity	Player_VY	Both	Velocity of player - Y
Velocity	Player_VR	Both	Velocity of player - Length
Velocity	Player_VTeta	Both	Velocity of player - angle
Position	Player_PosCount	Both	Count since last position observation
Velocity	Player_VelCount	Both	count since last velocity observation
Other	Player_IsKicker	Teammate	Is this player kicker
Pass	Player_FreePassAngle	Teammate	Maximum free angle for direct pass
Pass	Player_DirectPassDist	Teammate	Distance from ball to player
Opponent	Player_NearestOpponentDist	Teammate	Minimum distance from opponent to player
Position	Player_GCA	Both	Angle from player to opponent goal center
Position	Player_GCD	Both	Distance from player to opponent goal center
Shoot	Player_FreeShootAngle	Teammate	Maximum free angle for shoot
Stamina	Player_Stamina	Both	Stamina of player
Stamina	Player_StaminaCount	Both	Count since last stamina observation

Table 4. Sorting Algorithm

Sorting Method	Description
X	Sorting players of each team by their X of position Sorting Results: 9 8 5 3 4
X_FK	Similar to X approach, but the Kicker player has the first place in sorting Sorting Results: 5 9 8 3 4
Unum	Sorting players of each team by their Uniform Number Sorting Results: 3 4 5 8 9
Unum_FK	Like X, But Kicker Player be first Sorting Results: 5 3 4 8 9
AFC	Sorting player of each team by their angle from their current position to center of field Sorting Results: 4 5 9 8 3
AFC_FK	Similar to AFC, but the Kicker player has the first place in sorting Sorting Results: 5 4 9 8 3
AK	Sorting player of each team by their angle from their current position to the kicker player Sorting Results: 4 5 9 8 3
AK_FK	Similar to AK, but the Kicker Player has the first place in sorting Sorting Results: 5 4 9 8 3
AKG	Sorting Player of each Team by angle from position to Goal Center Sorting Results: 9 4 5 3 8
AKG_FK	Similar to AK, but the Kicker player has the first place in sorting Sorting Results: 5 9 4 3 8

2.3 Label Generator

This module takes action from the chain action module to generate the labels for each the data raw. The labels of data are noted in Table 5.

Table 5. List of Labels

Label	Description
Category	Hold || Pass || Dribble
TargetUnum	Uniform number of target player
TargetIndex	Index of target player after sorting
Description	Dribble || Direct Pass || Cross Pass || Through Pass || Lead Pass
TargetPosition	Target position
FirstKickAngle	Angle of selected action from ball
FirstKickSpeed	Ball kick speed

2.4 Results

To examine the impact of different input features on the prediction of the neural network, we chose 1 million of the Cyrus and Helios2019 raw data for training the deep neural network. We have created 10 diverse dataset using different sorting methods(see Table 4). The whole process of behavior prediction is demonstrated in Fig. 5. We reported the accuracy and error rate of model for those 10 dataset in Table 6. According to Table 6, Unum_FK has better accuracy in comparison to the approaches. In this section we try to evaluate the value of features

Table 6. Accuracy and error rate of the model for 10 datasets

Prediction	Type	*X*	*X_FK*	*Unum*	*Unum_FX*	*AKG*	*AKG_FK*	*AK*	*AK_FK*	*AFC*	*AFC_FK*
Category	Classification	76.55	77.23	76.69	77.37	76.09	76.61	76.08	76.63	76.03	76.41
Unum	Classification	57.22	57.71	60.57	60.39	56.17	56.48	56.22	56.93	56.7	57.31
Unum in Passes	Classification	57.87	58.04	61.80	62.51	57.27	57.71	57.07	57.49	57.70	57.20
Index	Classification	58.89	60.20	60.22	60.84	57.79	58.45	57.08	58.66	56.51	58.57
Index in Passes	Classification	58.49	59.73	61.79	62.01	58.58	58.13	58.72	58.42	57.39	57.06
Description	Classification	71.62	72.12	71.31	71.53	71.36	71.34	71.32	71.58	71.70	71.49
TargetPosition	Regresion	2.44	2.43	2.59	2.38	2.50	2.42	2.79	2.41	2.29	2.59
FirstKickAngle	Regresion	5.14	6.51	5.22	6.58	6.65	7.30	5.11	5.36	7.34	5.14
FirstKickSpeed	Regresion	0.041	0.054	0.043	0.054	0.055	0.06	0.042	0.044	0.061	0.042

for the prediction of agents' behavior. To accomplish this task, we exerted the Random Forest algorithm implemented in Sckiti-Learn and Permutation Feature Importance implemented by ELI5 library. The Permutation Feature Importance algorithm attempts the most effective features of input data for a trained neural network. We evaluated the value of sorted data features (sorted by UNUM) using Random Forest algorithm . Also, we selected two of our predictor deep neural networks that were trained by the UNUM sorted dataset to predict the Category or UNUM of target teammate. Using these two neural networks and the Permutation Feature Importance algorithm we chose the significant features of data. see Fig.4.

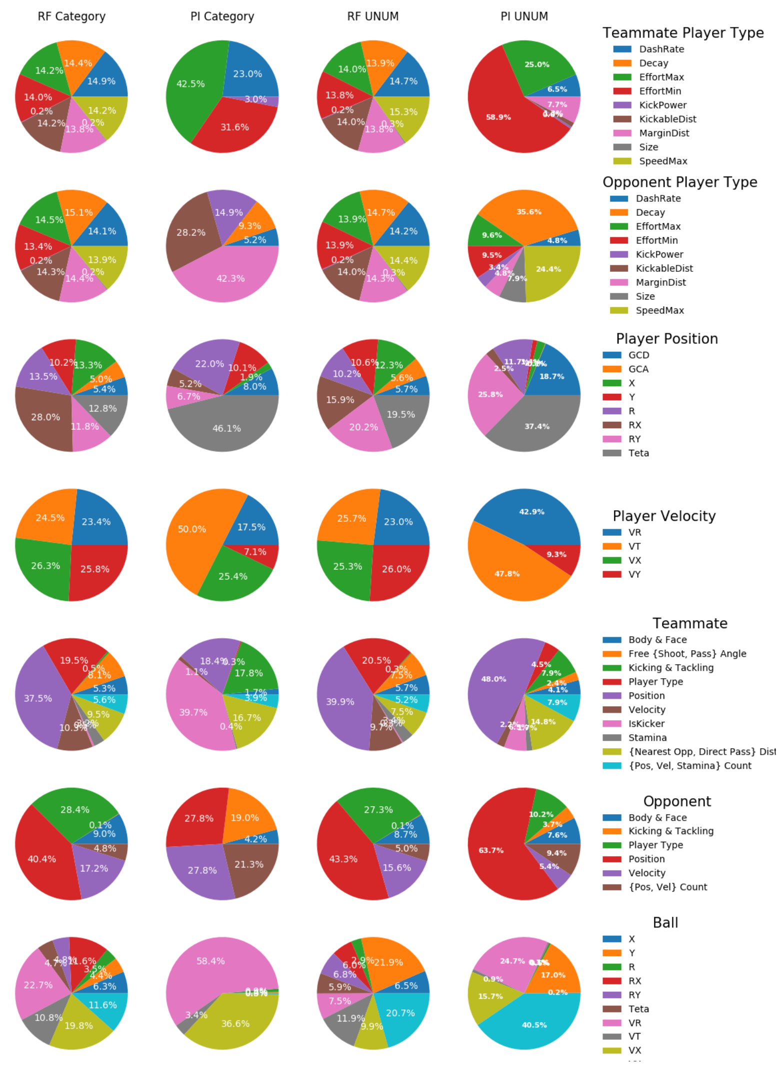

Fig. 4. The important features of sorted data extracted by Random Forest and Permutation Feature Importance

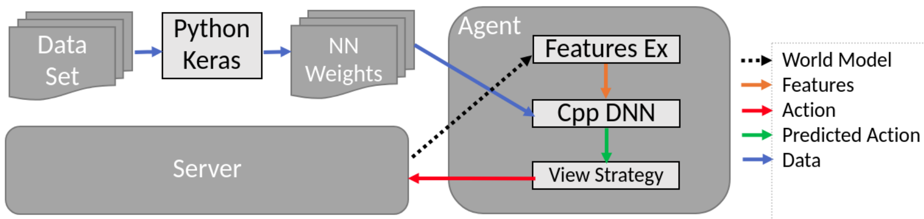

Fig. 5. The process of behavior prediction

2.5 How To Use Predictor

To assess the effect of the fullstate predictor network, we have examined it on Cyrus 2020. The trained neural network - that has been trained by sorted data (UNUM-FK sorted) - predicts the UNUM of fullstate target player. If the ball holder wouldn't have accurate information about the ball receiver agent, it prioritizes observing that agent. The experimental results of this approach on Cyrus team suggest win rate improvement from 8.19% to 17.49% and goal rate from 0.45% to 0.89%.

3 Conclusion

This paper describes all of the previous efforts and current research of Cyrus2020 on the exploitation of AI algorithms in 2D soccer simulation. Using the ***"full-state mode"*** of the server, we created a dataset from agents' perceived observations and their FWM behavior. Then we sorted this dataset, and we fed them to the disparate deep neural network for the behavior prediction. Subsequently, we exerted the best trained neural network to optimize the viewpoint of players. The experimental results demonstrate the improvement of the win rate and goal rate.

4 Future Work

In the near future we plan to improve the proposed approach in this TDP. We are planning to exert the Convolution Neural Network (CNN) as our predictor network. For the next step, we intend to process our data using the recurrent neural network which can process temporal data.

References

1. Akiyama H, Nakashima T. Helios base: An open source package for the robocup soccer 2d simulation. InRobot Soccer World Cup 2013 Jun 24 (pp. 528-535). Springer, Berlin, Heidelberg.

2. Fukushima T, Nakashima T, Akiyama H. Similarity Analysis of Action Trajectories Based on Kick Distributions. InRobot World Cup 2019 Jul 2 (pp. 58-70). Springer, Cham.
3. Akiyama, H., Nakashima, T., Fukushima, T., Suzuki, Y., Ohori, A.: Helios2019: Team description paper. In: RoboCup 2019 Symposium and Competitions, Sydney, Australia (2019)
4. Nakashima T, Akiyama H, Suzuki Y, Ohori A, Fukushima T. Helios2018: Team description paper. InRoboCup 2018 Symposium and Competitions: Team Description Papers, Montreal, Canada 2018 Jul.
5. Prokopenko, M., Wang, P.: Fractals2019: Guiding self-organisation of intelligent agents. In: RoboCup 2019 Symposium and Competitions, Sydney, Australia (2019)
6. Gabel T, Roser C. FRA-UNIted–team description 2016. In RoboCup 2016 Symposium and Competitions: Team Description Papers. Leipzig, Germany 2016 (pp. 10-18).
7. Gabel, T., Kloppner, P., Godehardt, E., Tharwat, A.: Communication in soccer simulation: On the use of wiretapping opponent teams. In: RoboCup 2018: Robot Soccer World Cup XXII. Springer (2018)
8. Gabel T, Sommer F, Breuer S, Godehardt E. FRA-UNIted–team description 2019. In RoboCup 2019 Symposium and Competitions: Team Description Papers. Sydney, Australia 2019.
9. Fabro JA, Dias ÁF, Júnior ET, Buiar GR, da Costa MR, Reis LP, Lau N. FCP_GPR_2019 Soccer Simulation 2D Team Description Paper.
10. Asali E, Negahbani F, Tafazzol S, Maghareh MS, Bahmeie S, Barazandeh S, Mirian S, Moshkelgosha M. Namira Soccer 2D Simulation Team Description Paper 2018. RoboCup 2018. 2018.
11. Asali E, Moravej A, Akbarpoor S, Asali O, Katebzadeh M, Tafazol S, Negahbani F, Valipour M, Mirian S, Mehran H, Haghighi AB. Persian Gulf Soccer 2D Simulation Team Description Paper 2017. InThe 21th annual RoboCup International Symposium, Japan, Nagoya 2017.
12. Asali E, Valipour M, Zare N, Afshar A, Katebzadeh M, Dastghaibyfard GH. Using Machine Learning approaches to detect opponent formation. In2016 Artificial Intelligence and Robotics (IRANOPEN) 2016 Apr 9 (pp. 140-144). IEEE.
13. Noohpisheh M, Shekarriz M, Bordbar A, Liaghat M, Salimi A, Borzoo D, Zarei A. Razi Soccer 2D Simulation Team Description Paper 2018.
14. Noohpisheh M, Shekarriz M, Bordbar A, Liaghat M, Salimi A, Borzoo D, Zarei A. Razi Soccer 2D Simulation Team Description Paper 2019.
15. Prokopenko, M., Wang, P.: Gliders2d: Source Code Base for RoboCup 2D Soccer Simulation League. In: RoboCup 2019: Robot World Cup XXIII, Springer (2020, to appear), also: arxiv.org/abs/1812.10202
16. Khayami R, Zare N, Karimi M, Mahor P, Afshar A, Najafi MS, Asadi M, Tekrar F, Asali E, Keshavarzi A. CYRUS 2D simulation team description paper 2014. InRoboCup 2014 Symposium and Competitions: Team description papers 2014.
17. Zare N, Karimi M, Keshavarzi A, Asali E, Alipour H, Aminian A, Beheshtian E, Mowla H, Jafari H, Khademian MJ. CYRUS 2D Simulation Team Description Paper 2015. InThe 19th annual RoboCup International Symposium, China, Hefei 2015.
18. Zare N, Sadeghipour M, Keshavarzi A, Sarvmaili M, Nikanjam A, Aghayari R, Firouzkoohi A, Abolnejad M, Elahimanesh S, Akhgari A,Cyrus 2D Simulation Team Description Paper 2018. RoboCup 2018. 2018.
19. Zare N, Sarvmaili M, Mehrabian O, Nikanjam A, Hossein S, Khasteh AS, Amini O, Barahimi B, Majidi A, Mostajeran A. Cyrus 2D Simulation 2019.

20. Prokopenko M, Wang P. Gliders2d: source code base for RoboCup 2D soccer simulation league. InRobot World Cup 2019 Jul 2 (pp. 418-428). Springer, Cham.
21. Bai A, Lu G, Zhang H, Chen X. WrightEagle 2D soccer simulation team description 2011. InRoboCup 2011 Symposium and Competitions: Team Description Papers, Istanbul, Turkey 2011 Jul.
22. Marian S, Luca D, Sarac B, Cotarlea O. OXSY 2014 Team Description. InRoboCup 2016 Symposium and Competitions: Team Description Papers, Leipzig, Germany 2016 Jul.
23. Sutton RS, Barto AG. Reinforcement learning: An introduction. MIT press; 2018 Oct 19.
24. Najafi, F., Dadvar, M., Habibian, S., Hosseini, A., Haeri, H., Arvan, M., Peykari, B. and Bagheri, H., 2016. RoboCup Rescue 2016 Team Description Paper MRL. Robocup Rescue 2016 TDP Collection.
25. Habibian, S., Dadvar, M., Peykari, B., Hosseini, A., Salehzadeh, M.H., Hosseini, A.H. and Najafi, F., 2021. Design and implementation of a maxi-sized mobile robot (Karo) for rescue missions. ROBOMECH Journal, 8(1), pp.1-33.
26. Buffinton, K.W., Wheatley, B.B., Habibian, S., Shin, J., Cenci, B.H. and Christy, A.E., 2020, May. Investigating the mechanics of human-centered soft robotic actuators with finite element analysis. In 2020 3rd IEEE International Conference on Soft Robotics (RoboSoft) (pp. 489-496). IEEE.
27. Dadvar, M., Moazami, S., Myler, H.R. and Zargarzadeh, H., 2020. Multiagent task allocation in complementary teams: a hunter-and-gatherer approach. Complexity, 2020.
28. Habibian, Soheil, "Analysis and Control of Fiber-Reinforced Elastomeric Enclosures (FREEs)" (2019). Master's Theses. 229. https://digitalcommons.bucknell.edu/masters_theses/229
29. MacAlpine, P., Urieli, D., Barrett, S., Kalyanakrishnan, S., Barrera, F., Lopez-Mobilia, A., Stiurca, N., Vu, V. and Stone, P., 2012, June. UT Austin Villa 2011: a champion agent in the RoboCup 3D soccer simulation competition. In AAMAS (pp. 129-136).
30. Barrett, S., Genter, K., He, Y., Hester, T., Khandelwal, P., Menashe, J. and Stone, P., 2012, June. Ut austin villa 2012: Standard platform league world champions. In Robot Soccer World Cup (pp. 36-47). Springer, Berlin, Heidelberg.