

A thesis submitted in partial satisfaction of the requirements
for the degree of Master of Computer Science and Engineering
in the Graduate School of the University of Aizu

Towards Data-Driven AI for the Game of Soccer



By

Mola Bogdan Georgii

August 2019

© Copyright by Mola Bogdan Georgii, August 2019
All Rights Reserved.

The thesis titled
Towards Data-Driven AI for the Game of Soccer

by

Georgii Mola Bogdan

is reviewed and approved by:

Main referee

Associate Professor

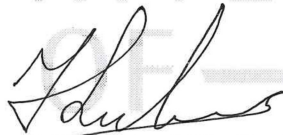
Maxim Mozgovoy



Date 22/07/2019

Professor

Ihor Lubashevsky



Date 22/07/2019

Senior Associate Professor

Evgeny Pyshkin



Date 22/07/2019

THE UNIVERSITY OF AIZU

August 2019

Contents

Chapter 1 Introduction	1
Chapter 2 Related works	2
2.1 Multiagent approaches	2
Chapter 3 Problem definition	3
3.1 Goals	4
Chapter 4 Dataset	6
4.1 TRACAB	6
4.2 STATS	8
4.3 Dataset details	8
Chapter 5 Method	10
5.1 Challenges	10
5.2 Agent design.....	11
5.2 Case extraction	11
5.3 Global/local context.....	12
Chapter 6 Results	13
6.1 Experiments with dataset from TRACAB.....	13
6.1 Experiments with dataset from STATS.....	15
Chapter 7 Discussion	17
Chapter 8 Conclusions	18
References	19

List of Figures

Figure 4.1. Real-world data visualization.....	9
Figure 6.1 Situations matched as a result of a global query (range = 8m)	14
Figure 6.2 Situations matched as a result of a local query (range = 1m, radius = 9m)	14
Figure 6.3 Situations matched as a result of a global query (range = 4m)	15
Figure 6.4 Situations matched as a result of a local query (range = 2m, radius = 9m)	16

List of tables

Table 4.1 TRACAB dataset description (chunk 2)	6
Table 4.2 TRACAB dataset description (chunk 3)	7
Table 6.1 Successful local context queries (%) TRACAB	13
Table 6.2 Successful local context queries (%) STATS	15

Abstract

Real soccer tracking data can be potentially used for building a learning by observation-based game AI system, possessing human-like decision making traits. Such system should be able to find matching example cases for the current game situation and act accordingly. This can be a challenging issue due to high dimensionality of soccer data and the requirements to detect approximate matches. We show how a simple k-d tree-based search can accomplish this task with modest space and time requirements, making it a feasible approach for a practical game system.

Chapter 1. Introduction

Creating a reasonable AI system for the game of soccer is a challenging task, addressed in numerous research works. [1-3] Designing high-performance soccer teams consisting of physical robots or virtual agents is the goal of annual RoboCup competition [4], attracting numerous participants and observers. Currently, most successful 2D Simulation League teams employ diverse techniques to obtain efficient, winning behavior [5].

Advancements in machine learning methods motivate some teams to experiments with approaches based, e.g., on neural networks. One such example is Brainstormers team [6] that won several major RoboCup events in 2000s. Machine learning can be used within the context of learning by observation methods, aiming to acquire patterns of successful behavior from the top teams. This idea is expressed, in particular, in the work of Michael and Obst [7], who propose learning from the best participants of the RoboCup 2D simulation league.

The growing availability of real sports tracking data (such as provided by STATS.com) begs the question whether it is possible to learn *virtual* team behavior from *real* soccer teams. Michael and Obst [7] note that teams of humans “*were easily won by computer programs*” in Robocup 2D environment, adding though that “*the soccer simulation was not designed to be played by humans*”. Thus, it is possible (but not certain) that learning from human data would not help creating strong virtual teams.

However, being strong is not the only requirement for game AI systems. In soccer, it is important to provide a realistic and enjoyable environment for human players, facilitating immersion and suspension of disbelief. As Sicart [8] observes, the aim of mainstream soccer game projects is “*to make the games even closer to the actual game, that is, to make the computer game converge with the sport*”. In particular, it requires the AI systems to behave in “human-like” way or even mimic behavior styles of particular real-life soccer teams, which can be a strong motivation for learning from player tracking data.

Learning soccer strategies from observation is a challenging task, since agent coordination is implicit, and dimensionality is very high [9]. However, there are recent promising results, obtained, for instance, with deep learning technologies [9, 1].

Traditionally, machine learning techniques rely on huge amount of data. In soccer domain human behavior datasets are available nowadays, but their size is not large, and it is probable that this situation will not change in near future. This factor is important to note, because it has influence on current and further research efforts.

Since computer soccer is essentially a game of spatial tactics, existing learning by observation approaches are primarily based on spatial features, such as player and ball coordinates. This observation motivated us to experiment with a straightforward k-d tree-based approach, able to find close points in a multidimensional space, as suggested in [10]. This method possesses a number of attractive features: it gives us an explicit criterion of similarity between the current onscreen game situation and game situations in the training dataset; it is computationally inexpensive; it allows us to see specific base cases for each decision, thus helping to fine tune and improve the system.

Chapter 2. Related works

2.1 Centralized and Concurrent Learning

Various techniques and design patterns are considered in related research works that study multiagent environments. In particular, Weber *et al.* [11] apply special tools like A Behavior Language (ABL) for controlling massive groups of agents in real time strategy games (RTS). In such environments, game AI is supposed to perform both short-term and long-term decision making, necessary to achieve tactical and strategical goals respectively.

Several deep learning techniques are applied in [12]. First, the authors discuss a *centralized* approach, where a controller reflects all states in a composed action (a set of actions for each agent). This way, the whole virtual environment can be considered a multi-agent partially observable Markov decision process. Collaboration is easier to model within this approach, but it significantly increases the state space. Next, the authors describe a so-called *concurrent learning*, where every agent is associated with a separate AI system, able to learn distinct roles, but lacking the knowledge spreading between the agents. Soccer is also a subject of such works, but so far smaller-team scenarios such as 3 vs 3 [13] and 2 vs 2 [14] are usually considered.

2.2 Believable agents

In the task of building an AI agent for a soccer video game, *believability* (or human-likeness) arises as an important requirement. Our current work relies on the behavior-capture system based on case-based reasoning and case-based planning [35]. The main idea of system is “*believable behavior is a key feature of game AI*”. Development of agent is based on transferring knowledge from skilled soccer players.

Learning by observation is an efficient form of knowledge acquisition, which requires the expert demonstrating the task being learned. An AI agent "observes" the human game play and derives a knowledgebase based on the player actions in each game situation. By analyzing a soccer match frame by frame, the agent processes raw data based on the decision logic/factors that are provided to it, and develops action graphs and situation generalization structures forming the basis of its behavior. Obtained behavior can be evaluated by having the agent to play the game instead of a human.

Chapter 3. Problem definition

Nowadays AI-related endeavors are typically aimed to solve sets of problems related to specific well-defined areas, while building general artificial intelligence is considered as a goal of theoretical cutting edge of computer science. That is why one of factors that contribute to an appropriate agent design is mostly defined by environment and a field of application.

According to Russel and Norvig [15], soccer game is a *task environment* that can be classified as *stochastic, dynamic, multiagent, fully observable, sequential* and *continuous* in context of AI. Another notable properties of task environment are *performance, actuators* and *sensors*.

Soccer game is *stochastic*: every next state of environment is not defined. It is still possible to predict states of a controlled team (within some limits), but there are no possibilities to obtain knowledge about states of an opponent team in next moment of time.

Dynamic characteristics of soccer follows from the previous statement. The environment (represented by large with an opponent team) can change its state regardless of AI-controlled team's actions. The opposite example is chess (without clock), where the environment will never change until the next AI move is done.

Since soccer is a team game, it can easily be identified as a *multiagent* environment. However, the notion of an individual agent is not defined in soccer. Soccer team members have to *collaborate* and *compete* against the opposing team, and the approaches to achieve collaboration vary. For example, RoboCup competitions presume that every player is equipped with a separate AI system, having its own unique set of parameters, and the means of communication between team members are deliberately limited. Thus, a single agent in RoboCup corresponds to an individual soccer team member. In contrast, one could also organize a team as a swarm-like entity, where all members share one "mind", implementing a "team as a single agent" model. Sometimes agents coworking principles influenced by application field. For example, "single agent per player" approach originates from limits of environment. For example, agents have to act independently in area where communication is limited or impossible due to obstacles.

Depending on environmental specifics, game of soccer is *partially observable* or *full observable task environment*. In RoboCup, each agent has its own limited visibility zone, so at any moment of time only a *part* of environment is observed by each agent. Here the ability of the agents to communicate is vital. In other task environments (for example, commercial soccer video games) it might be advisable to let the agents observe the positions of all the players and the ball on the field to simplify the process of AI development. This issue is directly related to *sensors* that define how information from the outer world is delivered to agents.

Task environment can be *sequential* or *episodic*. In *episodic* environments every generated decision does not depend on previous decisions. For example, classification tasks belong to such environments. In particular, in image recognition process a decision made for the current image will not affect subsequent decisions. In a sequential task environment short-term actions can have long-term consequences. In a soccer match every decision will cause opponent team to react, but at the same time there is no long-term planning like in real-time

strategy games, only some general tactical considerations and quick reactions to rapidly changing conditions. I tend to consider such environment as *sequential* because game situations can be represented by chains of actions. Moreover, in the general vision of this thesis it is presumed that AI can and should accumulate previously obtained experience (i.e., learn).

All states in the game of soccer are smoothly changing within a range of continuous values. Thus, soccer game agent must act in a *continuous game* environment (modeled within a discrete world of a digital computer). In practice it means that an AI system handles constantly changing spatial geometrical data and must react to a game situation with appropriate responsiveness. In other words, it is a real-time system. In its turn, this consideration makes fast reaction time a major factor in AI design.

3.1 Goals

While games can be used and are actually used to benchmark general AI technologies, there are important features, characterizing specifically *game* AI systems. It is generally presumed that the main goal of a player in a game is to win, and thus the best AI system in a typical AI competition is the system that wins.

However, the purpose of game AI is not necessarily to be strong. According to Dill [16], “*The one thing that is universally true is that games are about creating a particular experience for the player—whatever that experience may be. The purpose of Game AI (...) is to support that experience.*” Thus, depending on a particular game, the goal of a good AI system might well involve being strong, predictable, friendly, hostile, and so on. In other words, a good game AI testbed should support, at least, theoretically, various possible goals for AI-controlled characters. The goal of strong AI development for games like chess or Go can be considered achieved, since computers are able to defeat even the best human players. We can expect that more game genres will be added to this list in the nearest future. For example, a recent work by Oh *et al.* [17] discusses the development of an AI system, able to defeat professional human players in a modern fighting game. It is difficult to say how good modern AI methods are in playing sport games. Michael and Obst [16] observe that AI teams of RoboCup 2D Simulation League play better than human teams; they remark, however, that RoboCup is not designed to be played by people, which supposedly affects their performance. In any case, there is an independent goal of designing the AI system that contributes to the overall user enjoyment. Arguably, this problem is even more important for actual game development than the task of creating a strong AI system [38].

Team sports are a good testbed for investigating such non-skill related traits of AI systems as well. There is extensive literature on factors making team sports exciting for both athletes and spectators, for which there are many good examples [17, 19]. Likewise, there is a general understanding of what constitutes fun in the context of an AI system for a sports video game [20]. In particular, we often observe that people prefer playing against other people, because people behave in a certain “human-like” way that is perceived as inherently enjoyable [21]. Striving for a human-like behavior can be a legitimate goal for a sport game AI system, as important and challenging as a highly skilled behavior.

Human-likeness of AI behavior is the key goal for the present research. So far there is no ultimate solution for building human-like agents. The basic method to obtain human-like believable behavior is to analyze actual human behavior patterns and subsequently implement them in an AI system. Even hand-coded algorithms can implement specific behavioral patterns, considered “human-like” by their creators [22]. Purely manual methods of behavior creation, primarily based on finite-state machines and behavior trees, are still popular, but they are often characterized as requiring too much effort, while the resulting computer-controlled agents still can be distinguished from humans by the observers. A greater interest is evoked by the methods that can automatically construct agents’ knowledge by observing behavior of human players. So-called cognitive structures are the subject of various research works [23-25]. Cognitive architectures are usually responsible for high-level reasoning and strategic decision making. Their adopters often attribute human-likeness of the obtained AI behavior to human-likeness of decision-making processes implemented in cognitive architectures: typically, they are based on psychophysiological models of reasoning, and this fact presumably makes AI-controlled game characters believable.

The approach described in the present work is closer to another method: learning by observation [20]. It is a method of behavior creation by watching and analyzing behavioral patterns of existing game characters, usually human-controlled. Such an approach can be implemented with machine learning methods and can be used to obtain both believable and effective elements of behavior. In our case, behavior patterns come from real soccer players. Data-driven technologies are becoming widespread due to abundance of player-generated data, the ease of data delivery and increasing computational performance available at consumer level hardware. As an example of a relatively simple commercial game AI based on a pure data-driven approach, we can name Driveatar system of Forza 5 racing game [26]. Driveatar collects “behavior profiles” of human players and stores them in the cloud. These profiles are used to obtain believable AI drivers for the subsequent races.

In the task of designing a soccer AI based on human behavior it is also possible to use data obtained from human players playing soccer video games. However, this approach is currently impossible due to absence of soccer video games where every on-field role (such as a defender or a goalkeeper) is performed by a separate human. Evidently, such experience does not appeal to people in the context of video games. In most soccer implementations, a human player controls the player currently possessing the ball or the defender closest to the ball if the opposing team is attacking.

Chapter 4. Available Dataset

Using digitalized data recorded from real soccer matches allows to make key contribution to believability features of the system under development. At the present time, sport recordings are actively used by scientists for various types of data analysis, by sport clubs for studying statistical indicators, and by broadcasting companies to enhance spectator experience.

4.1 J1 TRACAB Dataset

For this work, I initially used digitalized soccer recordings obtained with TRACAB technology [27] that relies on video streams obtained from six still video cameras installed in a stadium. These streams are processed with tracking software that identifies players, ball, and referee at any game moment. This system based on SAAB — a proprietary military technology for data processing developed in Sweden [28]. Later this technology was adopted at FIFA World Cup 2010 as an official player tracking system. Furthermore, this technology was also adopted in major European soccer leagues, such as Premier League (the U.K.), Bundesliga (Germany), and La Liga (Italy).

The resulting data files contain various values describing game states and consist of colon-separated main chunks, representing integer or string values. Strings can be either plain strings or arrays of objects represented as strings. In the latter case, array elements are separated with semicolons. Individual properties of each object are separated with commas.

The dataset of real soccer recordings consists of three main chunks. The first is an integer chunk containing the cardinal frame number of the current frame, uniquely generated by the tracking system for each frame. The second chunk is an array of 29 objects that represent players, reserve players and the referee. The last chunk is an array of one object: the ball.

The **chunk 1** is just integer data. The data type of **chunk 2** is string-represented array of up to 29 objects. Each object contains the properties shown in Table 4.1. The data type of **chunk 3** is a string-represented array of one object. This object contains the properties shown in Table 4.2

Table 4.1: TRACAB dataset description (chunk 2)

Properties	Valid values	Details
Team	0, 1, or 3	”1=Home team, 0=Away team, 3=Referee Other values are used for internal purposes.
System target ID	1 to 29	Jersey numbers are 1 - 99. Jersey -1 is ”unassigned” except for team 3 (the referee)
Jersey Number	-1 or 1 to 99	
Position X	-5250 to 5250	
Position Y	-3400 to 3400	
Speed	0 to infinity	

Table 4.2: TRACAB dataset description (chunk 3)

Position X	-5250 to 5250	
Position Y	-3400 to 3400	
Position Z	0 to infinity	Note that the position of the ball center is displayed. Normal Z position for a ball on ground is thereby 10 cm.
Speed	0 to infinity	
Ball owning Team	'H' or 'A'	"H"=Home team, "A"=Away team
Ball status	"Alive" or "Dead"	"Alive"=In play, "Dead"=Not in play

The data is gathered with a frame rate of 25 frames per second.

The dataset used in this work is collected and provided by Data Stadium Inc. [36]. The dataset consist of 5 games of 6 teams. The game data originate from J1 League matches (Japanese top division soccer league). All games were played in the 2011 season, and conventional statistical data (including team formations) is available.

This dataset has no high-level information about the actions that take place on the game field. In particular, it lacks information about events such as player movements, passes and shots on goal. For the purposes of the present work the absence of these elements is not significant. However, these events are necessary for decision making step that will be described in next chapter.

In the process of case extraction it is important to keep appropriate player ordering. Such measure helps to increases a chance to find a matching case with similar meaning. If wrong players will be paired, false negative results will appear during search and discovered cases may not be useful. (this step is called "role-alignment" in [1]).

Since this dataset contained information about players, it was possible to identify roles manually using match details available in internet. Thus, no automated role-alignment is necessary.

4.2 STATS Dataset

Another available dataset consists of game records gathered by STATS.COM [30]. This dataset consists of series of 7500 sequences with total time equal to 45 games. Every sequence starts when some team obtains the possession of the ball and ends when a team loses the ball.

Each sequence contains a segment of tracking data corresponding to actual game play from a professional European soccer league. The format of each sequence is as follows:

- Each sequence is a matrix with 46 columns. Each row contains 23 pairs of (x, y) coordinates of 22 players from both teams and the ball at frequency of 10 frames per second.
- In the first 22 columns are 11 (x, y) pairs of defense team. The following 22 columns are coordinates of attacking team (defined as the team with consecutive possession of the ball). The last 2 columns are coordinates of the ball.
- Each set of 22 columns for both attacking and defending team consist of (x, y) pair for the goalkeeper, followed by 10 consecutive (x, y) pairs for the other 10 teammates. The identities and teams vary from sequence to sequence. However, within each sequence, the identity is consistent. Thus concretely, out of the 46 columns from each sequence, we know that the first 2 columns represent the coordinate of defense team's keeper. Columns 2 to 22 contain 10 consecutive (x, y) pairs of other defensive players. Columns 23 and 24 carry x and y coordinates of the attacking team's keeper. Columns 25 to 44 contain 10 consecutive (x, y) pairs of other attacking players. Columns 45 and 46 carry x and y coordinates of the ball.
- The coordinates originally belong to the [-52.5 meter, +52.5 meter] range along the x-axis, and [-34 meter, +34 meter] range along the y-axis, with the very center of the pitch being [0,0]. So, for example, to normalize the data to the range [-1, +1], one can simply divide the x-columns by 52.5 and y-columns by 34 (this effectively will re-scale the pitch, which roughly corresponds to soccer field of size 105mx70m, from a rectangular box to a square box).

The coordinates were also adjusted so that the attacking team will moves from left to right, meaning the defending team defends the goal on the right-hand side. In aggregate, the data set amounts to equivalently and approximately 45 games worth of playing time, with redundant and "dead" situations removed.

A major drawback of this dataset is the absence of important details. There is no knowledge about which teams take part in the matches, and player names and their roles are also unknown. As in TRACAB dataset, there are no details about events such as passes, and shots on goal.

4.3 Data Limitations and Visualization

Due to limited availability of digitalized sport records (recent wide spreading of recording technologies, issues related with acquiring such data) this research gain important additional goal: learning from *limited-size* datasets. This fact is important to note since conventional

machine learning methods usually operate with datasets of large sizes, typically containing from 6×10^4 to 9×10^9 records.

The problem of limited datasets becomes especially significant if the goal is to design a virtual team that imitate behavior style of some real-life team. Another obstacle for gathering a consistent dataset is related to changes of team members, injuries, red cards etc.

For visual confirmation of results and checking semantic value of cases a visualization tool was used.

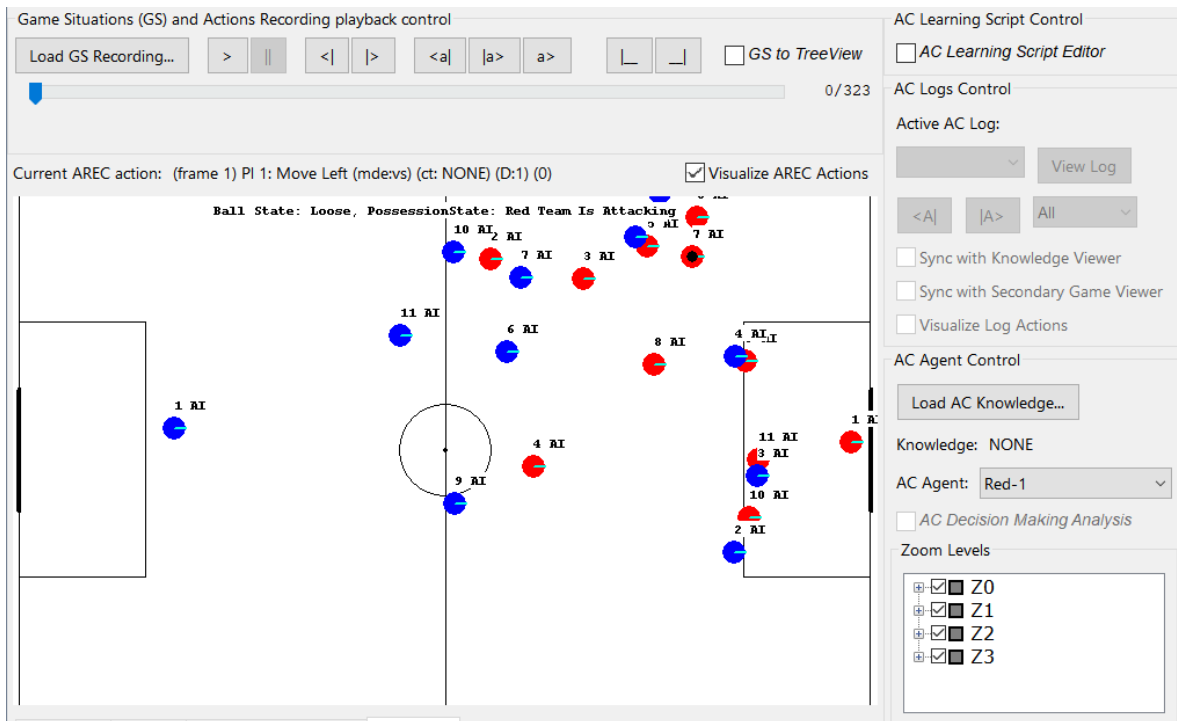


Figure 4.1: Real-world data visualization

Chapter 5. Method

5.1 Challenges

The dataset we currently use originates from real players records that originally was not created for the purposes similar to ours. So, it is necessary to perform a certain data preprocessing phase. During the subsequent steps, the AI system will be similar game situations (cases) in existing data structure to facilitate decision making.

Being multiagent and nontrivial, AI for game of soccer must deal with high-dimensional environment. High-dimensionality comes from a variety attributes that represent game state. Basic set of attributes includes coordinates of team and ball are 46 attributes. It worth to note other possible attributes like possession of ball and status of ball (in game or out of game) that are not continuous values, direction of players, speed of ball and/or players. Considering continuous nature of environment and high dimensionality, seeking of *exact* case in knowledge base becoming dramatically challenging for an AI system. Moreover, soccer game cannot be presented as only by combination of geometrical data. There is also a semantic component in every game situation. For example, one can get two situations with geometrically similar set of positions, but it may be wrong to correlate them because players being from different teams or having different roles significantly change semantic value of overall picture.

One of possible solution originates from “continuous environment” factor: an AI agent should not necessarily find an *exact* case. That means that it is possible to correlate two similar cases (within some limits) and consider them as one situation. Russel and Norvig [15] noted such principle as **limited rationality** — not pursuing perfect solution but accepting approximate “good enough” solution that still can satisfy goal. It is especially important when the system does not possess enough computational resources to process data. Musliner *et al.* [31] list requirements for mission-critical real-time AI systems (e.g. aircraft on-board control), some of whose are relevant for our work.. Among them are: acceptance of approximate solutions, precomputing (e.g. role-alignment in soccer), reduction of search variance and the principle of “graceful degrade” that aimed to support *smooth* and *seamless* system activity even in fault cases. In context of soccer AI it means that an agent should better generate a suboptimal decision rather than do nothing.

Search for inexact cases can be formulated as a range search task. That means that per single query an agent is supposed to retrieve a set of cases lying within acceptable limits. In the perfect case when all retrieved cases satisfy the desired conditions the agent could chose a random case, but usually it is necessary to drop cases irrelevant to the current game situation according to its semantics. For example, suppose the agent must make a decision about a pass in a situation where a certain opponent player stands between the passer and the receiver. Suppose that in a retrieved *similar* case, the opponent player is close to the original point, but not exactly in the middle between the passing team players, so its chance to intercept the pass is lower. Thus, the final decision will be influenced by soccer-specific factors.

In general, the desired agent must satisfy the following requirements:

- Fast single query. Time of reaction should be fast enough for supporting uninterrupted participation of the agent in game process. Referring to related research for RoboCup [32], one can state that making decision with frequency ≈ 10 time per second enough for effective reaction.
- Reasonable memory consumption. Considering application field of desired system (video games), memory consumption must be reasonable for consumer level of hardware.

5.2 Agent design

Mozgovoy and Umarov[34] define three possible approaches to the design of a soccer AI system:

- **Player with ball.** Only the player currently possessing the ball is controlled by a machine learning-based AI system. Other players are controlled by a simpler rule-based AI.
- **“Puppet master”.** The whole team is considered as one unit. Action of such unit is composed of all team members’ actions.
- **Players as separate agents.** An independent machine learning-based AI system is attached to every team member.

As soccer is fundamentally a game of spatial tactics, one would expect to deal with high-dimensional geometric data like team members and ball. That is why we decided to focus on kd-tree as a relatively simple, straightforward and effective algorithm that allows to retrieve close points in a multidimensional space. Algorithms based on kd-trees are widely used in the domain of computer graphic, for instance, in tasks like ray tracing or color reduction; however, they can also be applied for case-based reasoning tasks, as suggested in [10]. This method possesses a number of attractive features: it gives us an explicit criterion of similarity between the current onscreen game situation and game situations in the training dataset; it is computationally inexpensive; it allows us to see specific base cases for each decision, thus helping to fine tune and improve the system. A kd-tree based algorithm mostly corresponds to the **“Puppet master”** approach.

TRACAB dataset contain details about competing teams and players, which allows to perform manual role alignment. Since STATS dataset is totally anonymous, role alignment was necessary before experimenting. As a preliminary step, a simple naive algorithm was used. Firstly, we compare player positions relative to the goal along X-axis. The players located closer to the goal are labeled as defenders, and so on. Their in-field roles (e.g. left or right defender) are assigned similarly by comparing their positions along Y-axis.

5.3 Case extraction

We investigated how easy and computationally demanding is to find a base case in the training data for the given onscreen situation. In one of our previous projects [34], we tried to base decision making on players’ local contexts without relying on the complete team data, which greatly reduces problem dimensionality.

One of the key reasons to use k-d trees is the ability to perform search within a specified range in multidimensional data. Since we are dealing with floating-point numbers (coordinates), searching for exact cases is unreliable and (as shown in Chapter 3) often unnecessary. The k-d tree-based method is fast enough to search closest matches for complete 23-element vectors, containing the coordinates of all soccer players and the ball. As a result, a kd-tree contains a set of 46-dimensional points. The construction complexity for a kd-tree is $O(N \log N)$, and memory consumption is $O(N)$.

Since the original framerate is too high for the purposes of decision making, we reduced it to one frame per three seconds. Thus, our resulting training set (4 matches) consists of ≈ 7500 frames, and the test set (1 match) contains ≈ 1800 frames. For the STATS dataset we decided to extract one frame per second. In that case training set was based on 104728-record training set, while the test set contained 27442 records.

Adding new elements to a balanced k-d tree takes $O(\log_2 N)$ time, where N is the number of elements in the tree. In our case, each element is a vector of 46 floating-point numbers (two coordinates per each game object).

Querying an axis-parallel range in a balanced kd-tree has the complexity of $O(N^{1-\frac{1}{k}} + m)$, where m is the number of reported points, and k is the dimension of the k-d tree (in our case, $k=46$) [33]. To increase search speed, we retrieve at most 10 matching cases. In practice it keeps search time under 1ms on a conventional Intel i7-based laptop PC for our dataset.

5.4 Global and local contexts

Finding optimal range values is not a trivial task. In general, decision making in soccer is affected both by the local context (especially in the area around the ball) and the global situation on the field. The global context helps to identify “the big picture”: whether a certain team is attacking or defending, how close the ball is to the opposing team’s goal line, or whether the attacking team is about to play a set piece.

Global context can be identified with a relatively low-resolution matching, i.e., with higher range values and low precision of extracted cases. The local context determines specific actions of individual players, and thus requires much higher precision around the ball area. For example, matching defenders’ positions precisely is vital for a successful attack, while the location of own goalkeeper is nearly irrelevant.

Therefore, it is reasonable to test the outcomes of k-d tree queries both for the whole soccer field and for the limited area within the given radius around the ball.

In order to perform query in local context within desired radius we check whether players positions situated within a certain circle. The center of this circle is the position of the ball. For the players within this radius a regular range is applied, for the players outside this radius the range value is equal to the football field size, which means that the matching player can be found anywhere.

Chapter 6. Results

A range search procedure considers two points matching if the Euclidean distance between them is smaller than the given range value. By specifying shorter ranges, we can find closer matches at the higher risk of retrieving no results at all.

6.1 Experiments with dataset from TRACAB

Global (whole-field) queries for our test set yield the probability of finding a match within 0.5-1.0% for the range of 8 meters, and reaches 50% for the ranges of 12 meters and above. The results of local context queries are shown in the Table 6.1.

Table 6.1: Successful local context queries (%) TRACAB

Range, m	Radius around the ball, m					
	4	5	6	7	8	9
1	37.4	27.5	21.2	17.8	15.6	13.5
2	55.0	42.0	32.1	25.2	20.1	18.0
3	70.3	56.0	43.7	33.7	27.3	22.7
4	82.0	69.9	56.5	46.0	37.2	29.9

For the strictest local context query (range = 1m, radius = 9m), there is a 13.5% chance to find a base case for decision making. If no results are found, we can repeat the query with relaxed conditions by providing higher range and/or radius values until some case is found.

To analyze practical examples of identified matches, we used a soccer simulation engine to visualize results. Fig. 6.1 shows overlapped images of a query situation and the corresponding retrieved matching result for a global search procedure (range = 8m).

Fig. 6.2 shows an example result of a local query (range = 1m, radius = 9m). In both figures, players painted with a solid fill belong to the query situation, while semitransparent-filled players represent a match found in the k-d tree.

The general procedure for experiment with TRACAB dataset is as follows:

- One of five games used as test set. It allows to imitate a complete soccer match.
- The rest four matches accumulated in one solid knowledgebase (kd-tree data structure).
- The same approach is used for every match (for the purpose of cross-validation).

It worth to note that every experimental data setup provides approximately the same results. Thus, being representative, only one table has been included in the report.

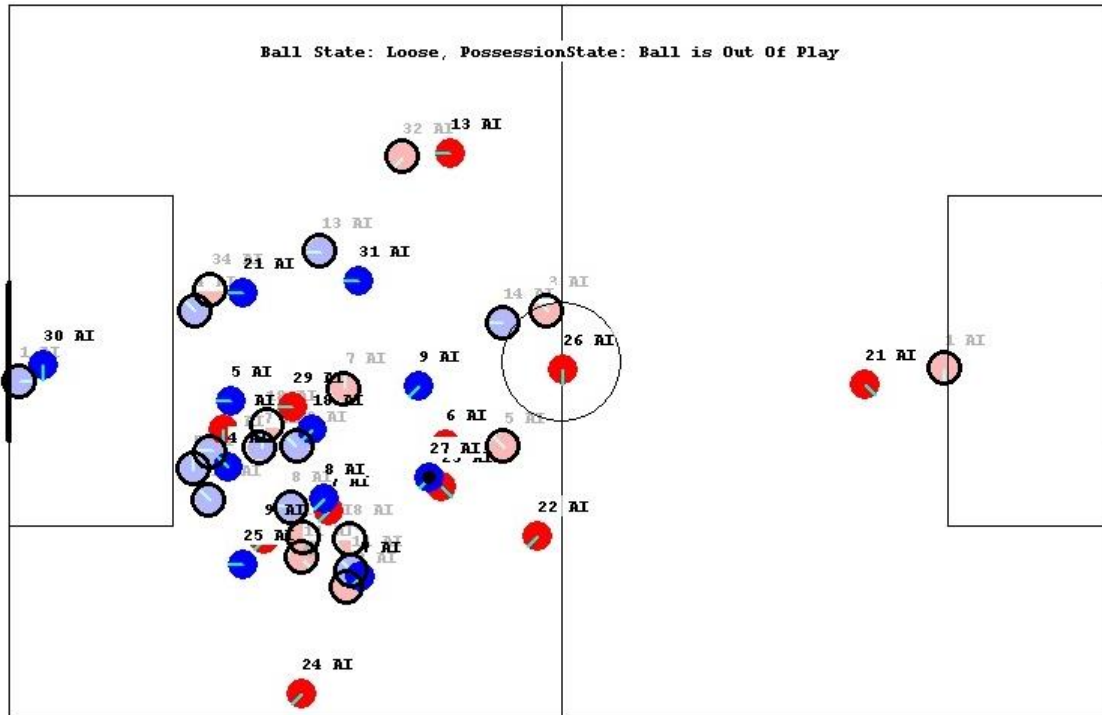


Figure 6.1: Situations matched as a result of a global query (range = 8m)

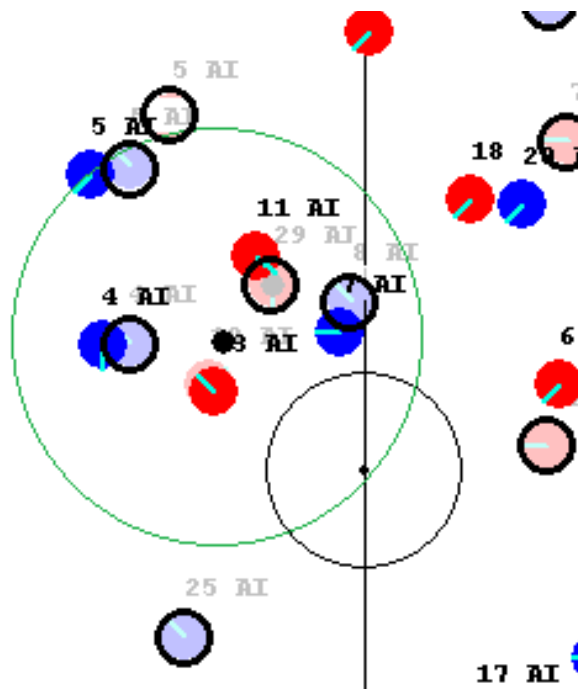


Figure 6.2: Situations matched as a result of a local query (range = 1m, radius = 9m)

6.2 Experiments with dataset from STATS

The dataset provided by STATS includes more records that influence results positively. For global context queries for our test set yield the probability of finding a match around 0.02% for the range of 4 meters, and it reaches 42% for the ranges of 8 meters and above. Fig. 6.3 shows result of global search with range 4 meters. Grey ellipsoids help to recognize pairs from current game situation and matched one.

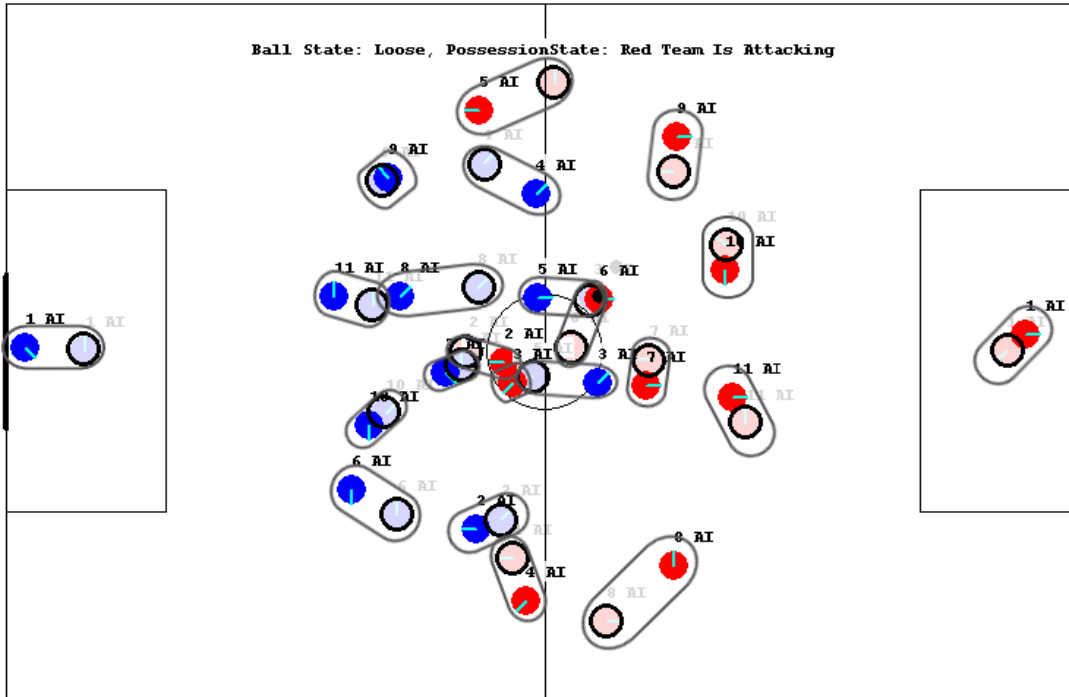


Figure 6.3: Situations matched as a result of a global query (range = 4m)

The results of local context queries are shown in the Table 6.2.

Table 6.2: Successful local context queries (%) STATS

Range, m	Radius around the ball, m					
	4	5	6	7	8	9
1	33.5	20.6	12.5	7.7	4.6	2.7
2	63.7	44.9	30.3	19.7	12.6	7.8
3	85.5	70.8	54	39.3	27.3	18.2
4	96.1	89.4	78.3	64.6	50.6	38

Fig. 6.4 shows an example result of a local query (range = 2m, radius = 9m). In both figures, players painted with a solid fill belong to the query situation, while semitransparent-filled players represent a match found in the k-d tree.

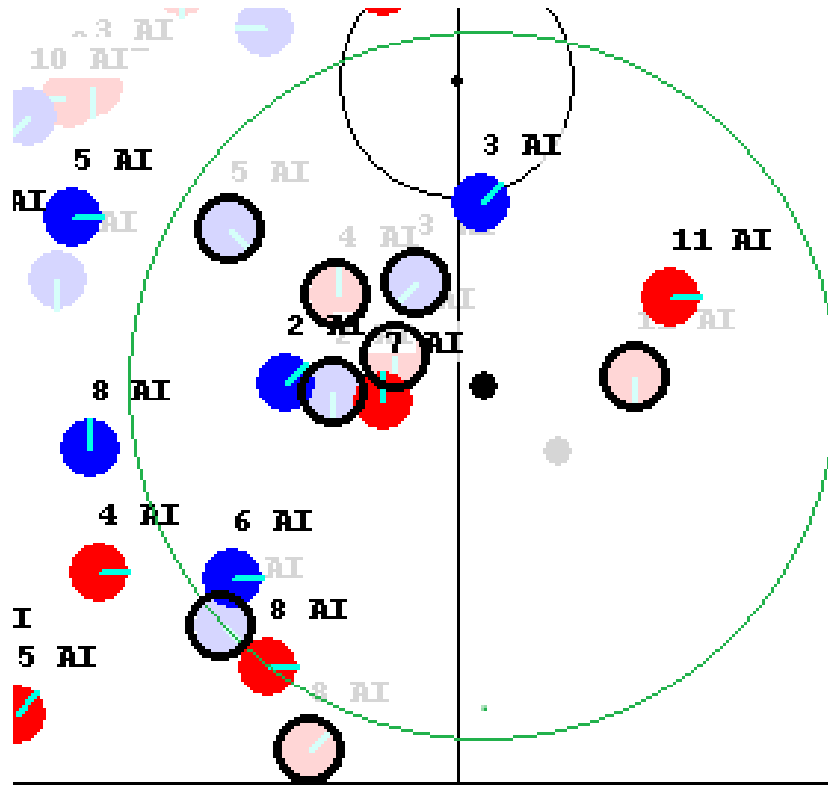


Figure 6.4: Situations matched as a result of a local query (range = 2m, radius = 9m).

Chapter 7. Discussion

Le *et al.* [1] consider an “average” team (the resulting model of their machine learning approach). They quantified how the players from real records differ from the “average” team. The average level of deviation across all players and teams is about ≈ 4 meters. As our global search with STATS dataset starts yielding results with the range of 4 meters, we consider obtained results as promising. For the proper evaluation of results, additional work on actual decision-making subsystem is necessary.

Comparing results from TRACAB and STATS it is possible to make the following observation: results scaling up with more data that ensures correctness. However, we believe that the same dataset still can yield better results if appropriate measures are applied. For instance, improving precomputing (role-alignment techniques) should increase the quality of found matches. STATS dataset allows to produce more results not only because of size: removing redundant and “dead” situations (with the ball out of field) was performed by its authors.

Local context search in its current implementation has a notable drawback. For the players located out of search radius the *range* parameter is set to a certain large number, which means the matching player can be found anywhere. There is a chance that the matching appears near to ball, which significantly changes game situation meaning.

As a further step, we are planning to apply a dynamic range method that takes into account a distance between the given player and the ball: faraway players will become less significant and get larger ranges.

Global search can be improved by applying more advanced role-alignment techniques. Without appropriate role-alignment even redundant dataset will not provide sufficient results.

To ensure results, we planning to apply Hungarian algorithm [37] that will serve as baseline test in role-alignment.

Chapter 8. Conclusion

We have applied a k-d tree-based approach for case retrieval in a soccer game. Our primary goal was to show the feasibility of this method both in terms of computational performance and its ability to find matching cases in the database. While our current datasets are too small to be used as a reliable source for decision making, their content is already sufficient for local context analysis of the field area surrounding the ball. Global reasoning is also possible: while the probability of finding a match with enough precision is not high, global context does not change rapidly, which makes frequent decision making unnecessary.

Apart from experimenting with larger datasets, role alignment, and actual decision making, we are planning to test the hybrid global/local approach, where an admissible range would increase for soccer field areas, located far away from the ball. The number of identified local matches can be increased by focusing on relative locations of the players around the ball rather than absolute coordinates on the field, which is done in the present version.

We believe that such an approach is useful spatial multidimensional data analysis. It is important to note that the complete solution for building an AI system cannot be based only on a pure kd-tree algorithm. It is necessary to enhance the basic algorithm with application-specific heuristics, like local context search.

References

- [1] H. M. Le, P. Carr, Y. Yue, P. Lucey. “Data-Driven Ghosting using Deep Imitation Learning”, in MIT Sloan Sports Analytics Conference, Boston, MA, March 3-4, 2017
- [2] P. Stone, R. S. Sutton, and G. Kuhlmann. “Reinforcement learning for RoboCup soccer keepaway”, in *Adaptive Behavior*, 13(3):165–188, 2005.
- [3] S. Liu, G. Lever, J. Merel, S. Tunyasuvunakool, N. Heess, T. Graepel. “Emergent Coordination Through Competition”, in *International Conference on Learning Representations* (2019). URL <https://openreview.net/forum?id=BkG8sjR5Km>
- [4] H. Kitano *et al.*, “RoboCup: A Challenge Problem for AI and Robotics” *Lecture Notes in Computer Science*, vol. 1395, 1998, pp. 1–19.
- [5] M. Prokopenko and P. Wang, “Disruptive innovations in RoboCup 2D soccer simulation league: from Cyberoos’ 98 to Gliders2016,” in *Robot World Cup, 2016*, pp. 529–541.
- [6] M. Riedmiller, T. Gabel, F. Trost, and T. Schwegmann, “Brainstormers 2D — Team Description 2008,” *RoboCup 2008: Robot Soccer World Cup XII, 2008*.
- [7] O. Michael and O. Obst, “BetaRun Soccer Simulation League Team: Variety, Complexity, and Learning,” *arXiv preprint arXiv:1703.04115*, 2017.
- [8] M. Sicart, “A tale of two games: football and FIFA 12,” in *Sports Videogames: Routledge*, 2013, pp. 40–57.
- [9] H. M. Le, Y. Yue, P. Carr, and P. Lucey, “Coordinated multi-agent imitation learning,” in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 2017, pp. 1995–2003.
- [10] S. Wess, K.-D. Althoff, and G. Derwand, “Using k-d trees to improve the retrieval step in case-based reasoning,” in *European Workshop on Case-Based Reasoning*, 1993, pp. 167–181.
- [11] B. G. Weber, P. Mawhorter, M. Mateas, and A. Jhala, “Reactive planning idioms for multi-scale game AI,” in *Proc. IEEE Symp. Comput. Intell. Games*, 2010, pp. 115–122.
- [12] J. K. Gupta, M. Egorov, M. Kochenderfer, “Cooperative Multi-agent Control Using Deep Reinforcement Learning. In *Autonomous Agents and Multiagent Systems*”, Springer, pp. 66– 83, 2017
- [13] N. Balachandar, J. Dieter, G. S. Ramachandran, “Collaboration of AI Agents via Cooperative Multi-Agent Deep Reinforcement Learning”, *arXiv:1907.00327*, 2019

- [14] Liu, S. et al. “Emergent coordination through competition”, In International Conference on Learning Representations (2019). URL <https://openreview.net/forum?id=BkG8sjR5Km>
- [15] S. J. Russell, P. Norvig, J.F. Canny, J. M. Malik, D. D. Edwards, *Artificial Intelligence: A Modern Approach, 3d Edition*. Englewood Cliffs: Prentice Hall (2010)
- [16] K. Dill, “What is game AI?”, Game AI pro, S. Rabin, Ed., pp.3–10, 2013
- [17] I. Oh, S. Rho, S. Moon, S. Son, H. Lee, and J. Chung, “Creating pro-level AI for real-time fighting game with deep reinforcement learning”, arXiv preprint arXiv:1904.03821, 2019
- [18] A. Correia and S. Esteves, “An exploratory study of spectators’ motivation in football,” *International Journal of Sport Management and Marketing*, vol. 2, no. 5-6, pp. 572–590, 2007.
- [19] T. K. Scanlan, P. J. Carpenter, M. Lobel, and J. P. Simons, “Sources of enjoyment for youth sport athletes,” *Pediatric exercise science*, vol. 5, no. 3, pp. 275–285, 1993
- [20] I. Umarov and M. Mozgovoy, “Creating Believable and Effective AI Agents for Games and Simulations: Reviews and Case Study.” *Contemporary Advancements in Information Technology Development in Dynamic Environments*, pp. 33- 57, 2014
- [21] P. Sweetser, D. Johnson, J. Sweetser, and J. Wiles, “Creating engaging artificial characters for games,” in *International Conference on Entertainment Computing*. [URL]. Available: <https://dl.acm.org/citation.cfm?id=958734>
- [22] M. Booth, The official Counter-Strike bot. <http://ww.gdcvault.com/play/1013625/>, 2004
- [23] D. Choi, P. Langley, “Evolution of the ICARUS cognitive architecture. *Cognitive Systems Research*”, 48, 25–38, 2018
- [24] J. Schrum, I. Karpov, R. Miikkulainen, “UT 2: Human-like Behavior via Neuroevolution of Combat Behavior and Replay of Human Traces”, *Proc. of IEEE Conference on Computational Intelligence and Games*, pp. 329-336, 2011
- [25] R. Le Hy, A. Arrigoni, P. Bessière, O. Lebeltel, “Teaching Bayesian Behaviours to Video Game Characters”, *Robotics and Autonomous Systems*, vol. 47, 2004, pp. 177-185.
- [26] S. McInnis, “How Forza 5 is Crowd-Sourcing Artificial Intelligence”, , [URL] <http://www.gamespot.com/articles/how-forza-5-is-crowd-sourcing-artificial-intelligence/1100-6409975/>, accessed: 2019.07.19
- [27] “Tracab,” <http://chyronhego.com/sports-data/tracab>, accessed: 2019-07-18.
- [28] “Saab,” <http://saabgroup.com/>, accessed: 2019-07-18.

- [30] “STATS” <https://www.stats.com/>, accessed: 2019-07-20
- [31] D. J. Musliner, J. A. Hendler, A. K. Agrawala, E. H. Durfee, J. K. Strosnider, and C. J. Paul., “The challenges of real-time AI”, Technical Report CS-TR-3290, UMIACS-TR-94-69. University of Maryland Institute for Advanced Computer Studies, 1994.
- [32] Stone, P., Sutton, R.: “Scaling reinforcement learning toward RoboCup soccer”, in Proceedings of International Conference on Machine Learning, pp. 537–544, 2001
- [33] J. L. Bentley, J. H. Friedman, “Data structures for range searching,” ACM Computing Surveys (CSUR), vol. 11, no. 4, pp. 397–409, 1979
- [34] M. Mozgovoy and I. Umarov, “Believable team behavior: Towards behavior capture AI for the game of soccer,” in 8th International Conference on Complex Systems, pp. 1554–1564, 2011
- [35] M. Mozgovoy and I. Umarov, “Behavior Capture with Acting Graph: A Knowledgebase for a Game AI System” in Databases in Networked Information Systems (DNIS): 7th International Workshop, S. Kikuchi, A. Madaan, S. Sachdeva, and S. Bhalla, Eds, Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 68–77, 2011
- [36] “Data stadium inc.” <https://www.datastadium.co.jp>, accessed: 2019-07-20.
- [37] H. W. Kuhn. “The Hungarian Method for the assignment problem.” Naval Research Logistics Quarterly, 1955.
- [38] Mozgovoy M., Preuss M., Bidarra R., Nakashima T., Harada T. *Team Sports for Game AI Benchmarking Revisited*. Unpublished manuscript, 2019