








RoboCupSimData: Software and Data for Machine Learning from RoboCup Simulation League

Olivia Michael¹ , Oliver Obst¹  , Falk Schmiddsberger² ,
and Frieder Stolzenburg² 

¹ Centre for Research in Mathematics, Western Sydney University,
Locked Bag 1797, Penrith, NSW 2751, Australia
O.Obst@westernsydney.edu.au

² Automation and Computer Sciences Department,
Harz University of Applied Sciences,
Friedrichstr. 57–59, 38855 Wernigerode, Germany

Abstract. The main goal of this work is to facilitate machine learning research for multi-robot systems as they occur in RoboCup, an international scientific robot competition. We describe our software (a simulator patch and scripts) and a larger research dataset from games of some of the top teams from 2016 and 2017 in Soccer Simulation League (2D), where teams of 11 agents compete against each other, recorded by this software. We used 10 different teams to play each other, resulting in 45 unique pairings. For each pairing, we ran 25 matches, leading to 1125 matches or more than 180 h of game play. The generated CSV files are 17 GB of data (zipped), or 229 GB (unzipped). The dataset is unique in the sense that it contains local, incomplete and noisy percepts (as sent to each player), in addition to the ground truth logfile that the simulator creates (global, complete, noise-free information of all objects on the field). These data are made available as CSV files, as well as in the original soccer simulator formats.

Keywords: Mobile robotics · Multi-robot systems · Simulation · RoboCup · Reinforcement learning · Self-localization

1 Introduction

RoboCup is an international scientific robot competition in which teams of multiple robots compete against each other. The RoboCup soccer leagues provide platforms for a number of challenges in artificial intelligence and robotics research, including locomotion, vision, real-time decision making, dealing with partial information, multi-robot coordination and teamwork. In RoboCup, several different leagues exist to emphasize specific research problems by using different kinds of robots and rules.

There are different soccer leagues in the RoboCup with different types and sizes of hardware and software: small size, middle size, standard platform league, humanoid, 2D and 3D simulation [7]. In the 2D soccer simulation league [1], the emphasis is on multi-robot teamwork with partial and noisy information in real-time. Each of the robots is controlled by a separate program that receives sensor information from the simulator as an input and, asynchronously to sensor input, also decides on a next action that is sent back to the simulator, several times a second. This complexity of the environment, with continuous state and action spaces, together with the opportunity to compete against each other, makes RoboCup soccer an interesting testbed for machine learning among many other applications.

To assist automated learning of team behavior, we provide software to generate datasets from existing team binaries, as well as a large dataset generated using 10 of the top participants in RoboCup 2016 or 2017. While it is possible to use the simulator directly for machine learning, it is not easily possible to use the system to learn from existing teams. Our software allows to record data that facilitates this by recording additional data that is not normally available from playing other teams directly. For this, we modified and extended the simulator software to record data from each robots local perspective, i.e., with the restricted views that depend on each robots situation and actions, and also include the sensor noise. In addition, for every step of the game (100 ms), we recorded ground truth information (such as positions and velocities) of all objects on the soccer field, as well as basic actions of each robot. This ground truth information is usually recorded in a binary format logfile, but not available to teams during a match. We also provide scripts to translate these logfiles and the local player information into comma-separated values (CSV) files.

Using this software, we created a dataset from all pairings of the selected teams with 25 repetitions of each game, i.e., 1125 games in total. With 11 robots in each team, a single game dataset consists of 22 local views plus a global (ground-truth) view. These views are made available as CSV files, in addition to the original logfiles, that include additional sensors and actions of each robot recorded as text files.

The provided data are useful for various different tasks including imitation learning (e.g., [3]), learning or testing of self-localization (e.g., [11]), predictive modeling of behavior, transfer learning and reinforcement learning (e.g., [15]), and representation learning for time series data [9]. The next sections describe the environment, robots, and data in more detail.

2 Description of the Software Environment

The RoboCup Soccer Simulation Server `rcssserver` [10] is the software used for the annual RoboCup competitions that have been held since 1997. It is hosted at github.com/rcsoccersim/. We used the `rcssserver` version 15.3.0 to create our software and the data. The simulator implements the (2D) physics and rules of the game, and also handles the interface to the programs controlling each player.

By default, players use a 90° field of view and receive visual information every 150 ms. Throughout the game, this frequency can actively be changed by each player individually to 75 ms, 150 ms, or 300 ms, by changing the field of view to 45° , 90° , or 180° , respectively. Visual information is transmitted in the form of (Lisp-like) lists of identified objects, with the level of detail of information depending on object distances. Potential objects include all other players on the field, the ball, and landmarks like goal posts, flags, and side lines. Each player also receives additional status information, including energy levels, referee decisions, and the state of the game, every 100 ms. Each robot can issue parameterized actions every 100 ms, to control its locomotion, the direction of its view, and its field of view. A more detailed description of the information transmitted can be found in the simulator manual [6].

3 Overview on the Provided Data

In robotics, data collections often comprise lidar data recorded by laser scans. This is very useful in many applications, e.g., field robotics, simultaneous localization and mapping (SLAM) [16], or specific purposes such as agriculture [4]. Other datasets have been collected, e.g., for human robot interaction [2] or human-commentated soccer games [5]. In many contexts, there is not only one but several robots which may be observed. The data from RoboCup that we consider here include information about other robots in the environment and hence about the whole multi-robot system.

Data from multi-agent systems like the RoboCup or the multi-player real-time strategy video game StarCraft [8] provide information on (simulated) environments as in robotics. However, in addition, they contain data on other agents and thus lay the basis for machine learning research to analyze and predict agent behavior and strategies important for multi-robot systems. To provide a diverse dataset, we include several teams from the last two RoboCup competitions, allowing for different behaviors and strategies.

Perception and behavior of each robot during a game depends on the behavior of all other robots on the field. Game logfiles, i.e., files containing ground truth information obtained from recording games, can be produced from the simulator and are recorded in a binary format. Access to individual player percepts, however, is only possible from within the player code. To learn from behavior of other teams, it is useful to use the exact information that individual players receive, rather than the global (and noise-free) information in recorded logfiles. We therefore modified the simulator to additionally also record all local and noisy information as received by the robots on the field in individual files for each player. This information is stored in the same format as it is sent to players. We also provide code to translate these individual logs into CSV files that contain relative positions and velocities (cf. Sect. 6).

We chose ten of the top teams from the RoboCup 2D soccer simulation world championships 2016 in Leipzig, Germany (CSU_Yunlu, Gliders, HELIOS2016, Ri-one) and 2017 in Nagoya, Japan (CYRUS, FRA-UNited, HELIOS2017,

HfutEngine, MT, Oxsy). Team binaries including further descriptions can be downloaded from archive.robocup.info/Soccer/Simulation/2D.

We played each team against each other team for 25 times, resulting in 1125 games. Generated CSV files from one match vary in size (approx. 200 MB), in total we collected about 229 GB of CSV files (17 GB of data zipped). For each game, we also recorded the original logfiles including message logs. We also generated files with ground truth data as well as local player data in human-readable format. Finally, we made our generating scripts available (cf. Sect. 6), so that they can be used to reproduce our results or to produce additional datasets using other robotic soccer teams. There is also a smaller subset of 10 games where the top-five teams play against each other once (163 MB CSV files plus 217 MB original logfiles). Our software and data is available at

bitbucket.org/oliverobst/robocupsimdata/

with a detailed description of the ground truth and the local player datasets.

4 Description of the Ground Truth Data

According to rules of the world soccer association FIFA, a soccer pitch has the size of 105 m \times 68 m. This is adopted for the RoboCup soccer simulation league. Nevertheless, the physical boundary of the area that may be sensed by the robots has an overall size of 120 m \times 80 m. For the localization of the robot players, the pitch is filled with several landmark objects as depicted in Fig. 1: flags (f), which are punctual objects, lines (l), the goal (g), and the penalty area (p). The origin of the coordinate system is the center point (c). The pitch is divided horizontally (x -direction) into a left (l) and right (r) half and vertically (y -direction) into a top (t) and a bottom (b) half. Additional numbers (10, 20, 30, 40, or 50) indicate the distance to the origin in meters. Since every soccer game takes place on the same pitch, there is only one file with information about the landmarks for all games that lists all these coordinates, given as a table in CSV format, with name `landmarks.csv`. For example, the row `f r t,52.5,34` says that the right top flag of the pitch has the (x, y)-coordinates (52.5, 34).

Further table files provide information about the respective game. The names of all these files – all naming conventions are summarized in Fig. 2 – contain the names of the competing teams, the final scores for each team, possibly extended by the result of a penalty shootout, a time stamp (when the game was recorded), and some further identifier.

The central SoccerServer [6] controls every virtual game with built-in physics rules. When a game starts, the server may be configured by several parameters which are collected in one file with the identifier `parameters`. For example, the row `ball_decay,0.94` denotes that the ball speed decreases by the specified factor (cf. [14]). However, from a robotics point of view, most of the information in this file is not very relevant, like the stamina of the robots, the noise model, or the format of the coach instructions. We therefore skip further details here.

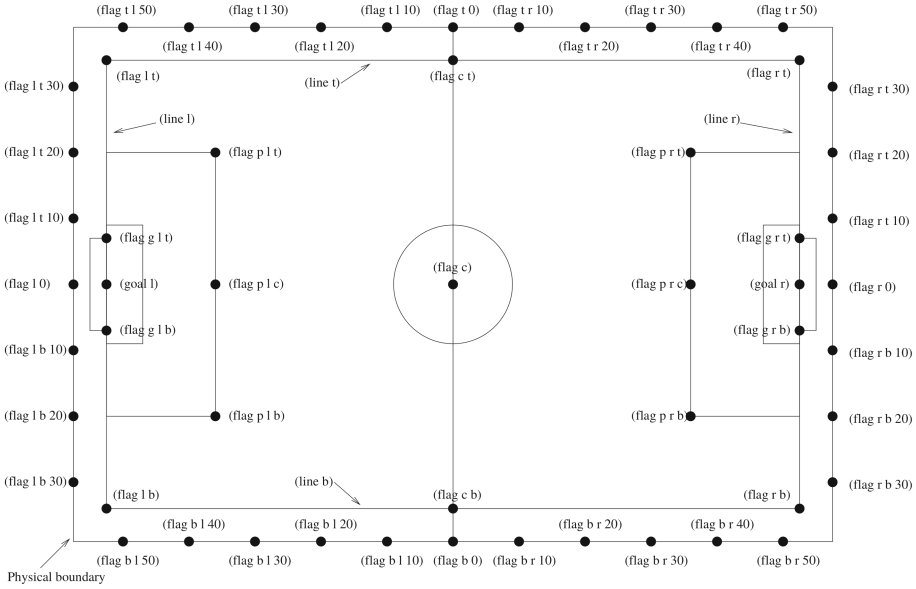


Fig. 1. Flags and lines in the robotic soccer simulation (cf. [6]).

1. landmarks: landmarks (static information for all games, 1 file)
2. game data: <time>-<team left>-<score left>-vs-<team right>-<score right>-<id>
 where <id> =
 - parameters (server configuration parameters, 1 file)
 - groundtruth (logfile information for each game, 1 file)
 - <team name>-<player number>-<suffix>
 where <suffix> =
 - landmarks (relative distances and angles to landmarks, 22 files)
 - moving (relative distances and angles to ball and other players, 22 files)

Fig. 2. Name conventions for the log and data files.

A soccer simulation game in the RoboCup 2D simulation league lasts 10 mins in total, divided into 6000 cycles where the length of each cycle is 100 ms. Simulations are different from each other even with the same teams, as noise is added to sensor data and player actions, and even attributes of players randomly vary between runs. Games are recorded in logfiles, which comprise information about the game, in particular about the current positions of all players and the ball including their velocity and orientation for each cycle. This information is collected for the whole game in a table with the identifier `groundtruth`. For each time point, the play mode (e.g. `kickoff`), the current ball position coordinates and its velocity is listed. Furthermore, the positions and velocities of each player of the left (L) and the right (R) team including the goalkeeper (G) is stated.

For example, the column with head `LG1 vx` contains the velocity of the left goalkeeper in x -direction. Finally, information about the robots body and head orientation and their view angle and quality is included. The absolute direction a player is facing (with respect to the pitch coordinate system) is the sum of the body and head direction of that player.

5 Description of Local Player Data

The visual sensor of the players reports the objects currently seen. The information is automatically sent to players every sense step with a frequency depending on the player view width and quality. By default it is set to 150 ms. Thus, in addition to the three files mentioned above, 44 more files are available for each game. For each of the altogether 22 robots (ten field players and one goalkeeper per team), two files with local player data is provided, hosting information about where the respective player sees the landmarks and moving objects, respectively.

The file with final identifier (suffix) `landmarks` provides the distances (in meters) and angles (in degrees) to the respective landmarks relative to the robot head orientation for each step. Analogously, the file with final identifier `moving` provides the actual relative distances and angles to the ball and all other players. Sometimes the player number or even the team name is not visible and hence unknown (u) to the robot. In this case, the respective piece of information is left out. If data is not available at all, then this is marked by `NAN` in the respective table element. The server also provides information about the velocity, stamina, yellow and red cards, and the commands (e.g. dash, turn, or kick) of the robots. In some cases there is also information about the observed state of other robots available, in particular, whether they are kicking (k), tackling (t), or are a goalkeeper (g).

6 Code

The soccer simulator communicates with players using text messages in form of lists, via UDP (see also Sect. 2). These individual messages can currently not be recorded, in contrast to the simulator logfiles. One option for developers of teams is to implement recording of messages that their own agents receive. But to collect data from other teams, the simulator software had to be modified instead. Our code contains patches to the simulator that allow recording of visual and body messages in individual files for each player. Messages are stored in their original format to keep the amount of processing during the game minimal. Running a simulation with our additional software will result in a number of recorded files:

- the visual and body messages (two files for each player). The file names follow the same naming convention as the game data CSV files (cf. Fig. 2) but use the `.rcv` suffix for visual messages and `.rcb` for body messages.

- a recording of the game (the ground truth in a binary format), using the suffix `.rcg`.
- commands from players as received by the simulator (in plain text), using the suffix `.rcl`.

Recording these files for all players on the field into files on a network drive can result in significant traffic, and it may be preferable to record onto a local drive and transfer the data after each match in order to reduce the impact of recording on the simulation.

Three different pieces of code are part of this project:

1. There is the `rcssserver-patch` written in C++ which modifies the RoboCup simulator to also log each player’s visual and body sensors.
2. To convert the simulator logfile (ground truth) into a CSV file, we provide `rcg2csv`, a C++ program that is built using the open source `librcsc` library (see osdn.net/projects/rctools/releases/p3777). Logfiles are recorded at regular intervals of 100 ms. Optionally, `rcg2csv` stores all simulation parameters in an additional CSV file.
3. To convert visual messages into CSV files, we provide a Python program `see2csv.py` that translates player visual messages into two files: a CSV file for moving objects (other players and the ball), and a CSV file with perceived landmarks.

7 Conclusions

RoboCup provides many sources of robotics data, that can be used for further analysis and application of machine learning. We released software to create research datasets for machine learning from released RoboCup simulation league team binaries, together with a large and unique dataset of 180 h of game play. This package allows a number of problems to be investigated: approaches to learning and test self-localization, predictive world-models, or reinforcement learning. The publicly available research dataset has proven itself instrumental as it has already been used as a testbed for time-series analysis and autoencoding (data compression) by recurrent neural networks [12,13] as well as for the analysis of soccer games with clustering and so-called conceptors [9].

Funding. The research reported in this paper has been supported by the German Academic Exchange Service (DAAD) by funds of the German Federal Ministry of Education and Research (BMBF) in the Programme for Project-Related Personal Exchange (PPP) under grant no. 57319564 and Universities Australia (UA) in the Australia-Germany Joint Research Cooperation Scheme within the project *Deep Conceptors for Temporal Data Mining* (Decorating).

References

1. Akiyama, H., Dorer, K., Lau, N.: On the progress of soccer simulation leagues. In: Bianchi, R.A.C., Akin, H.L., Ramamoorthy, S., Sugiura, K. (eds.) RoboCup 2014. LNCS (LNAI), vol. 8992, pp. 599–610. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-18615-3_49
2. Bastianelli, E., Castellucci, G., Croce, D., Iocchi, L., Basili, R., Nardi, D.: HuRIC: a human robot interaction corpus. In: Calzolari, N., et al. (eds.) Proceedings of the Ninth International Conference on Language Resources and Evaluation, LREC 2014, pp. 4519–4526 (2014)
3. Ben Amor, H., Vogt, D., Ewerton, M., Berger, E., Jung, B., Peters, J.: Learning responsive robot behavior by imitation. In: International Conference on Intelligent Robots and Systems, IROS-2013, pp. 3257–3264, November 2013
4. Chebrolu, N., Lottes, P., Schaefer, A., Winterhalter, W., Burgard, W., Stachniss, C.: Agricultural robot dataset for plant classification, localization and mapping on sugar beet fields. *Int. J. Robot. Res.* **36**(10), 1045–1052 (2017)
5. Chen, D., Mooney, R.J.: Learning to sportscast: a test of grounded language acquisition. In: Proceedings of the 25th International Conference on Machine Learning (ICML) (2008). <http://nn.cs.utexas.edu/?chen:icml08>
6. Chen, M., et al.: RoboCup Soccer Server – for Soccer Server Version 7.07 and Later. The RoboCup Federation, February 2003. <https://sourceforge.net/projects/sserver/files/rcssmanual/manual-7.08.1/manual.pdf>
7. Kitano, H., Asada, M., Kuniyoshi, Y., Noda, I., Osawa, E., Matsubara, H.: RoboCup: a challenge problem for AI. *AI Mag.* **18**(1), 73–85 (1997)
8. Lin, Z., Gehring, J., Khalidov, V., Synnaeve, G.: STARDATA: a StarCraft AI research dataset. CoRR - Computing Research Repository abs/1708.02139, Cornell University Library (2017). <http://arxiv.org/abs/1708.02139>
9. Michael, O., Obst, O., Schmidberger, F., Stolzenburg, F.: Analysing soccer games with clustering and conceptors. In: Akiyama, H., Obst, O., Sammut, C., Tonidandel, F. (eds.) RoboCup 2017. LNCS (LNAI), vol. 11175, pp. 120–131. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-00308-1_10
10. Noda, I., Matsubara, H., Hiraki, K., Frank, I.: Soccer server: a tool for research on multiagent systems. *Appl. Artif. Intell.* **12**(2–3), 233–250 (1998)
11. Olson, C.F.: Probabilistic self-localization for mobile robots. *IEEE Trans. Robot. Autom.* **16**(1), 55–66 (2000)
12. Steckhan, K.: Time-series analysis with recurrent neural networks. Project thesis, Automation and Computer Sciences Department, Harz University of Applied Sciences (2018). (in German)
13. Stolzenburg, F., Michael, O., Obst, O.: Predictive neural networks. CoRR - Computing Research Repository abs/1802.03308, Cornell University Library (2018). <http://arxiv.org/abs/1802.03308>
14. Stolzenburg, F., Obst, O., Murray, J.: Qualitative velocity and ball interception. In: Jarke, M., Lakemeyer, G., Koehler, J. (eds.) KI 2002. LNCS (LNAI), vol. 2479, pp. 283–298. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45751-8_19
15. Taylor, M.E., Stone, P.: Transfer learning for reinforcement learning domains: a survey. *J. Mach. Learn. Res.* **10**, 1633–1685 (2009)
16. Tong, C.H., Gingras, D., Larose, K., Barfoot, T.D., Dupuis, É.: The Canadian planetary emulation terrain 3D mapping dataset. *Int. J. Robot. Res.* (2013). <http://asrl.utorias.utoronto.ca/datasets/3dmap/>