The 10th Workshop on Humanoid Soccer Robots
15th IEEE-RAS International Conference on Humanoid Robots, Seoul, Korea, November 2015

Fuzzy-Based Classification of Game Situations

Matthias Hofmann and Jan Gelhaar*

***Abstract*— Robot soccer has seen impressive progress in the last decade. One of the most important topics to advance in this area is a reliable classification of game situations. This is required to develop and execute higher-level tactics and strategies by robot soccer teams. This paper presents the toolbox *FuzzyLabs* that is domain-independent, extendable and user-friendly. We utilize it to build a fuzzy-based rule classification based on expert knowledge for distinguishing game situations in robot soccer. Our study shows promising results.**

I. Motivation

Developing higher-level strategies becomes more and more important in the robot soccer community. For more sophisticated team play, e.g. applying tactical decisions, reliable categorization of game situations is desirable. Thus, the goal of this paper is twofold: On the one hand, we show the functionality of the provided software toolbox *FuzzyLabs*, consisting of the components *FuzzyController*, *FuzzyDesigner*, and *Game Situation Editor*. On the other hand, we employ these tools for the detection of a game situation based on fuzzy logic.

The remainder of the paper is structured as follows. Section II describes related work in this area. We present the software toolbox in section III. We describe a case study in section IV, and evaluate the fuzzy-logic approach in Section V. The paper concludes in section VI.

II. Related Work

The theorem of fuzzy logic was first introduced by Lofti A. Zadeh back in 1965[1]. The main difference to the conventional set theorem is that elements are not restricted to either belonging to a set, i.e. fuzzy variables belong to certain degrees to sets. A mapping of the fuzzy variable into the fuzzy set is called membership function. The membership functions of traditional sets are mapping from the super set into the set of $0; 1$, while the membership functions map from the super set into the interval [0;1].

Like with traditional sets, it is possible to describe propositional formula with the typical logic operators, like conjunction, disjunction, negation, and implication. In the last decades a bunch of different semantics for the logical operators have been proposed, with which the expressions can be evaluated. Further discussion of the different semantics can be found in [2], [3].

There are many approaches to solve classification problems. Most of them origin from the area of machine learning, more specifically supervised learning. Supervised learning is the machine learning task of inferring a function from labeled training data. Popular techniques in supervised learning that have been successfully used in the robot soccer community are decision tree learning [4], support vector machines, and neural networks [5]. Moreover, case-based reasoning has been successfully applied to robot soccer [6] as well. In contrast to the mentioned approaches, we design a fuzzy-based inference system based on a labeled data set. The data set has been collected empirically from domain experts.

III. Software Toolbox

In this section, we present *FuzzyLabs*, a powerful toolbox for processing fuzzy logic. While there already exist many open source implementations in the area fuzzy logic, we decided to build a new system from scratch. There are two main reason for this: On the one hand, we wanted to make sure that all concepts of fuzzy systems are addressed in the toolbox. On the other hand, we recognized that there is a need for an infrastructure which enables the easy and fast development and evaluation of fuzzy symbols with rich Graphical user interface support. *FuzzyLabs* consists of three components, the *FuzzyStudio*, the *FuzzyController*, and the *Game Situation Editor*. These components are presented in the following subsections.

A. The FuzzyController

The *FuzzyController* can be seen as a black box processing system for fuzzy logic. The *FuzzyController* is capable of the evaluation of fuzzy rules according to popular strategies presented in [2]. It receives a set of numeric values, processes those values according to a specified strategy against a set of fuzzy rules, and finally computes a numeric result which represents a defuzzified value. The controller must be initialized with the desired evaluation strategies and the definition of the fuzzy logic which includes fuzzy rules, and the definition of the fuzzy symbols which are used by those rules. Additionally, a mapping, defining which numeric variables are valid and assigned to which fuzzy variable is needed for the definition. By initializing the controller with the definition and the strategies, any dependencies among the fuzzy rules are resolved and an execution plan is created for the inference run. The execution plan consists of several iterations, with which the rules are divided into disjunctive subsets, that will be used by each iteration.

The first iteration always represents the fuzzification of the numeric values into fuzzy values and therefore no rule gets applied in this step. The result of every iteration is called a *scope* which is as set of current values for the fuzzy variables. These will be used for the next iteration. With each

Robotics Research Institute, TU Dortmund University, , Germany
`{forename.surname}@tu-dortmund.de`

subsequent iteration a new scope will be created, holding the new values for the fuzzy variables, which were calculated using the current subset of fuzzy rules. After the last iteration the fuzzy variables that have been marked for defuzzification in the definition of the fuzzy logic are converted back into numeric values, according to the specified defuzzification strategy.

The *FuzzyController* offers the possibility to read the internal states of the inference mechanism, i.e. the scopes and meta-information of the iterations. This can be effectively used for debugging purposes and improvements of the fuzzy logic with a tool, which was also developed by the project group, the *FuzzyStudio*.

B. FuzzyStudio *- An Editing and Evaluation Program for Fuzzy Logic*

The *FuzzyStudio* application serves two points. Firstly, it makes it possible to define the fuzzy logic for the controller in a WYSIWYG[1]-fashion. Secondly, it enables debugging and evaluation of this logic by allowing an investigation of the controller and the observation of the inference mechanism on close.

The *FuzzyStudio* application is generally designed as a plug-in system for defining and evaluating fuzzy logic. All required fuzzy variables, values, and rules can be defined and then evaluated by manually providing numeric input values for the controller, and then examining the output results. Besides manually entering the input values, it is possible to provide these values with user defined plug-ins that can be mounted into the application. For instance, it is possible to create a plug-in that provides real-time sensor data directly transfered from the robots. The system is written in C-Sharp, and utilizes Microsoft's Windows Presentation Foundation (MFC). Figure 1 shows the application.

C. A Flexible Game Situation Editor

Since we need a set of game situations for the evaluation of our approach, we designed the flexible game situation editor. It is implemented as a plugin-in for the *FuzzyStudio* component. This plug-in lets the user define new game situations by simply moving the players and the ball on the field. Multiple game situations can be saved. The situation editor is able to send numeric values of a selected situation directly to the fuzzy controller. The graphical user interface of the editor is shown in figure 2.

IV. Classifying Game Situations - A Case Study

This section describes how the *FuzzyStudio* component can be used in robot soccer for classification of game situations. We describe the input of the fuzzy logic, and how we model the game situations.

[1]What you see is what you get

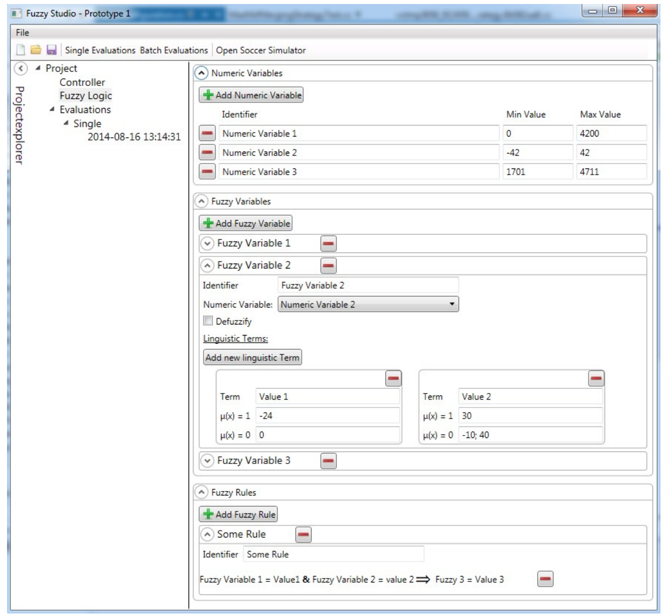

Fig. 1. Graphical user interface of the FuzzyStudio application

Fig. 2. The situation editor PlugIn

A. Representation of the World State

A system controlling the behavior of an agent needs a model of the current world state to select and perform actions. Many software frameworks for robot soccer provide symbols with a low level of abstraction, e.g. whether the ball is seen by the agent, or which robot is next to the ball.

Hence, the goal of this work is it to bring more abstraction into the world state, i.e. a symbolic representation of a current game situation. Thus, we categorize game situations into the following entities:

- `Critical Defense`: A critical defense situation occurs when the opponent team is in an advantageous position so that scoring a goal soon is very likely.

- `Defensive Situation`: This is a defensive situation in which the opponent team has possession of the ball and is approaching the goal.
- `Open Game`: An open game depicts a situation, which can result in an attack by either team. An example of this might be a situation, in which no player is currently near the ball.
- `Attacking Situation`: Analogous to the defensive situation, the attack situation describes that the own team is now in ball possession and is advancing on the opponent's goal.
- `Critical Attack`: This situation is analogous to the critical defense.

Since the propositions are vague, we use fuzzy logic for classification. Fuzzy logic offers the possibility to express simple if-then rules for inferring game situations. Another beneficial characteristic of fuzzy logic is a good interpolation behavior, which should counteract flickering or rapid jumps of the calculated values, and result in stable interpretation of the current game situation.

B. Modeling Game Situations with Fuzzy Logic

One main challenge when modeling the game situations of a soccer game is the limitation of the state space. If the developed fuzzy logic aims to cover all possible scenarios in a soccer game, the user ends up with an exponentially growing number of rules for each additional input variable. In order to shrink the number of fuzzy rules to a manageable magnitude, the fuzzy rules are organized hierarchic.

This way, we extract a set of new fuzzy values from the initial input values that have a slightly higher level of abstraction. Those abstract variables get used in the next stage to result into a new set of values with an even higher abstraction level. Although this can be applied in a very granular fashion, the first version of the fuzzy logic uses only two stages for inferring the game situation.

In the first stage, the game situation in the immediate vicinity of the ball, the own goal and the opponent goal are calculated. After that, the three separate game situation are combined to infer an overall game situation. Figure 3 illustrates the different stages and levels of abstraction. Please note, that *XXCenterDistanceToXX* denotes the median distance rather than the average distance.

V. Evaluation

In order to effectively evaluate the suitability of the fuzzy logic approach, samples are needed, which are labeled with ground truth data by domain experts. For this the *Flexible Game Situation Editor* plug-in (see subsection III-C) was used to create 100 game situations. The game situations were reconstructed from video footage of recent games, or created by deliberately chosen positions for the players, or generated by randomly assigning player positions.

A total number of 14 experts annotated these sample situations. For this purpose, we used an annotation program that was implemented especially for this purpose. For validation, all situations were labeled twice by each expert. This results in a total number of 2800 labels. We observed that the labels of all users, beside some minor spikes, were consistent for each sample, and are therefore suitable for the evaluation of the fuzzy controller's performance.

After analyzing the created fuzzy logic rule base for inference of the game situation, it shows promising results. The fuzzy logic was evaluated against the ground truth data. 50 of 100 situations that were assessed by the controller lay within the standard deviation of the annotation by the project group participants.

Randomly picking a value would only have resulted in a 24% chance of lying inside the standard deviations. Although the fuzzy rule base is simple, this fact shows that the current approach has good potential for correctly detecting game situations. Furthermore, the calculated situation values showed a standard deviation of 15.5% from the average ground truth situation values, and a 16% standard deviation from the median values.

Figure 4 shows the results of the evaluation: The red bars represent the differences of the values calculated by the fuzzy logic, and the average value of the annotations by the experts. The blue stripes depict the standard deviations of the ground truth data. The values in the diagram are scaled to a range between 0 and 200, because 200 represents the maximum difference between two game situation values. This is due to the fact, that the fuzzy logic uses the value of -100 to represent a critical defense, while a most critical attack situation results in a value of 100.

VI. Conclusion and Future Work

In this paper, we presented the software tools *FuzzyStudio* and *FuzzyController*. They are domain-independent, highly customisable and extensible. Moreover, we have shown how the application of these toolboxes can contribute to solve the game situation detection problem. A first study shows promising results.

As shown in the previous sections, the fuzzy logic approach has a good chance of reliably classification of game situations. Therefore, next steps will be to incorporate the fuzzy inference system into a robotic framework to apply the system in a real-time application.

Furthermore, we aim to investigate whether fuzzy logic is a suitable approach to infer roles for each robot in a given game situation. This leads to specific actions assigned with the role, and would result in a fuzzy-driven behavior for robots.

References

[1] L. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, no. 3, pp. 338 – 353, 1965. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S001999586590241X

[2] J. Kahlert, *Fuzzy Control für Ingenieure*, ser. Fuzzy Control für Ingenieure: Analyse, Synthese und Optimierung von Fuzzy-Regelungssystemen. Vieweg, Braunschweig/Wiesbaden, 1995.

[3] B. Bosko, *Fuzzy Thinking: the new science of fuzzy logic*. Hyperion, New York, 1993.

[4] S. Konur, A. Ferrein, E. Ferrein, and G. Lakemeyer, "Learning decision trees for action selection in soccer agents," in *In Proc. of Workshop on Agents in dynamic and real-time environments*, 2004.

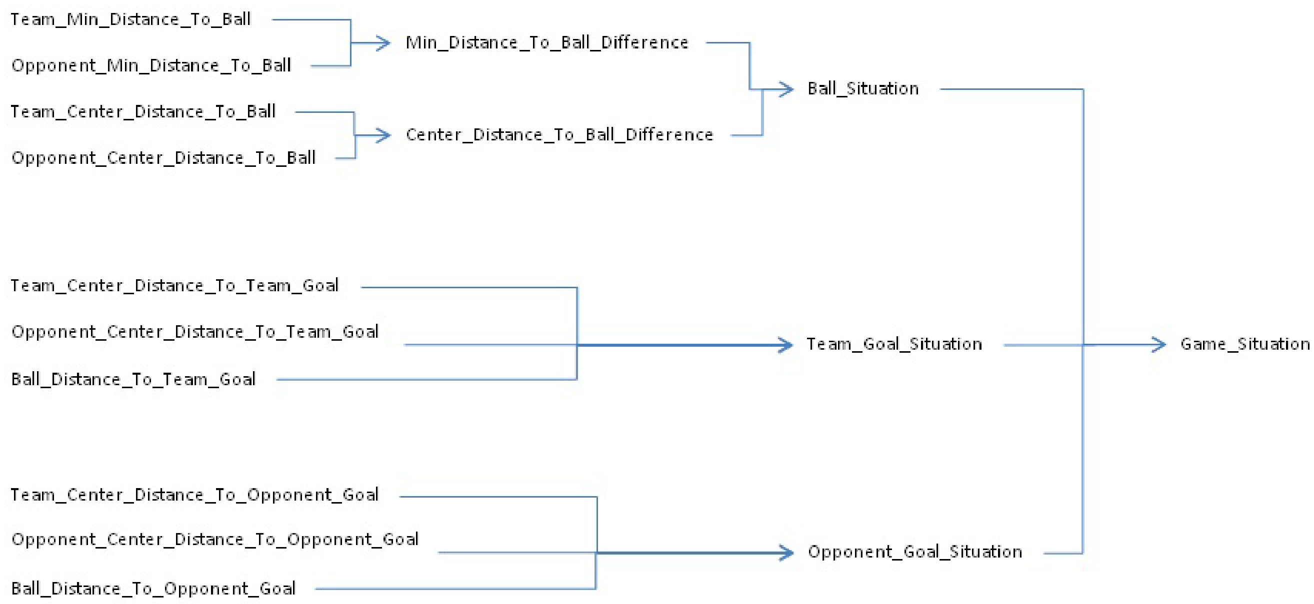

Fig. 3. Hierarchical fuzzy logic to infer game situations.

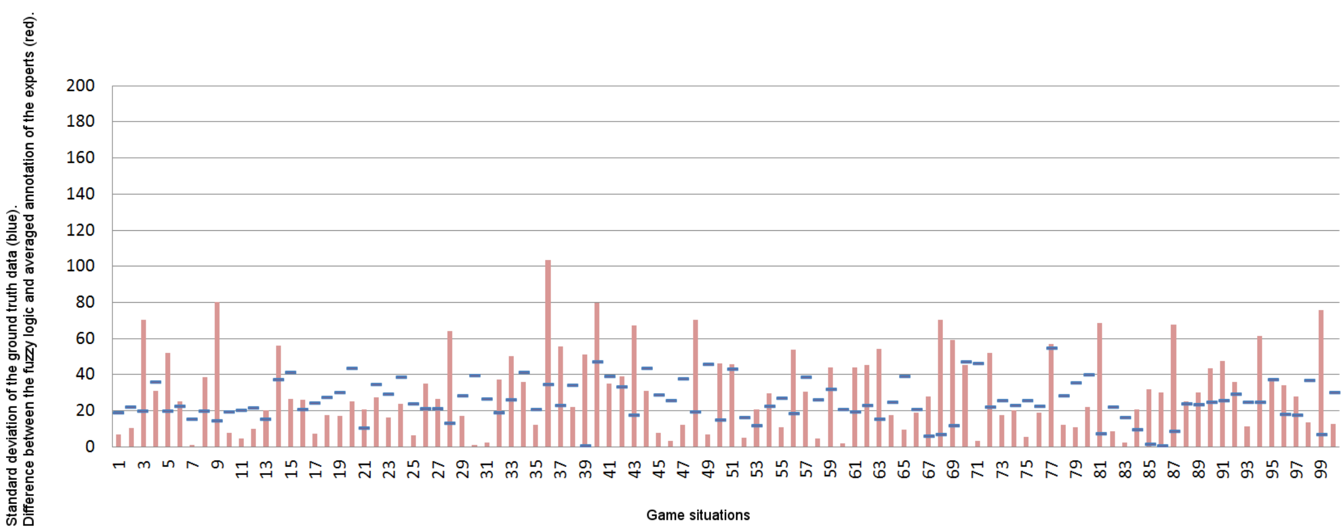

Fig. 4. Difference between the designed fuzzy logic, and the annotation by domain experts.

[5] B. Faria, L. Reis, N. Lau, and G. Castillo, "Machine Learning Algorithms Applied to the Classification of Robotic Soccer Formations and Opponent Teams," in *2010 Ieee Conference on Cybernetics and Intelligent Systems, Cis 2010*, 2010, pp. 344–349, citations: scopus.

[6] A. Karol, B. Nebel, C. Stanton, and M.-A. Williams, "Case based game play in the robocup four-legged league part i the theoretical model," in *RoboCup 2003: Robot Soccer World Cup VII*, ser. Lecture Notes in Computer Science, D. Polani, B. Browning, A. Bonarini, and K. Yoshida, Eds. Springer Berlin Heidelberg, 2004, vol. 3020, pp. 739–747. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-25940-4_73