Fuzzy Representations and Control for Domestic Service Robots in Golog

Stefan Schiffer[1] Alexander Ferrein[2] Gerhard Lakemeyer[1]

[1] Knowledge Based Systems Group
RWTH Aachen University, Aachen, Germany
{schiffer,gerhard}@cs.rwth-aachen.de
[2] Robotics and Agents Research Lab
University of Cape Town, Cape Town, South Africa
alexander.ferrein@uct.ac.za

Abstract. In the RoboCup@Home domestic robot competition, complex tasks such as "get the cup from the kitchen and bring it to the living room" or "find me this and that object in the apartment" have to be accomplished. At these competitions the robots may only be instructed by natural language. As humans use qualitative concepts such as "near" or "far", the robot needs to cope with them, too. For our domestic robot, we use the robot programming and plan language Readylog, our variant of Golog. In previous work we extended the action language Golog, which was developed for the high-level control of agents and robots, with fuzzy concepts and showed how to embed fuzzy controllers in Golog. In this paper, we demonstrate how these notions can be fruitfully applied to two Robocup@Home scenarios. In the first application, we demonstrate how qualitative fluents based on a fuzzy set semantics can be deployed. In the second program, we show an example of a fuzzy controller for a follow-a-person task. While these programs have to be regarded as a proof-of-concept for the possibility to integrate qualitative concepts into Readylog beneficially for such applications, we aim at implementing these programs on our domestic robot platform in the future.

1 Introduction

Classical applications for approaches to cognitive robotics and reasoning about actions are delivery tasks, where the robot should deliver a letter or fetch a cup of coffee. In these domains, it becomes obvious that solving such tasks deploying reasoning and knowledge representation is superior to, say, reactive approaches in terms of flexibility and expressiveness. An even more advanced application domain is RoboCup@Home [12, 13]. As a distinguished league under the roof of the RoboCup federation the robots have to fulfil complex tasks such as "*Lost&Found*", "*Fetch&Carry*", or "*WhoIsWho*" in a domestic environment. In the first tasks the robot has to remember and to detect objects, which are hidden in an apartment, or has to fetch a cup of coffee from, say, the kitchen and bring it to the sitting room, while in the latter the robot needs to find persons

Proceedings of SIMPAR 2010 Workshops
Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS
Darmstadt (Germany) November 15-16, 2010
ISBN 978-3-00-032863-3
pp. 183-192

and recognise their faces. The outstanding feature of these applications is that they require integrated solutions for a number of sub-tasks such as safe navigation, localisation, object recognition, and high-level control (e.g., reasoning). A particular complication is that the robot may only be instructed by means of natural interaction, e.g., speech or gestures. Human-robot interaction is hence largely based on natural language. For example, in the *Fetch&Carry* task it is allowed to help the robot with hints like "The teddy is near the TV set".

Humans make frequent use of qualitative concepts like *near* or *far*, as the example shows. It would be desirable that the robot could interpret these concepts and cope with them. When reasoning techniques are deployed to come up with a problem solution for these domestic tasks, also these mechanisms need to be able to cope with those qualitative concepts. But even as logic-based reasoning approaches make inherently use of qualitative concepts, the rest of the complex robot architecture does not. Hence, one needs to bridge the gap between the qualitative high-level control and the quantitative robot control system.

In this paper, we show how this gap can be bridged for domestic robot applications. We extended the logic-based high-level robot programming and plan language Readylog [2, 3] with so-called *qualitative fluents* describing properties of the world based on fuzzy set theory [4] and integrated fuzzy control techniques into the robot control language [5]. This enables us (1) to map qualitative predicates to quantitative values based on a well-defined semantics, and (2) to combine fuzzy control and logic-based high-level control. In the sequel, we show how these concepts can be used beneficially to formulate compact solutions for tasks such as *Fetch&Carry*. While we only give a preliminary specification here, for our future work we aim at deploying these programs to our domestic robot platform, which participated successfully at RoboCup@Home competitions in the past [10, 11].

The rest of this paper is organised as follows. In Section 2, we give a brief introduction to the robot programming and planning language Readylog and the situation calculus, which Readylog is based on. We recapitulate previous work on integrating fuzzy sets and fuzzy control structures into Golog in Section 3, before we show our qualitative domain description in Section 4. In particular, we define necessary qualitative predicates for the domestic service robotics domain and define fuzzy control structures to enable the robot to cope with qualitative predicates. We conclude with Section 5.

2 The Situation Calculus and Golog

2.1 The Situation Calculus

The Situation Calculus [7] is a second order logical language with equality which allows for reasoning about actions and their effects. The world evolves from an initial situation due to primitive actions. Possible world histories are represented by sequences of actions. The situation calculus distinguishes three different sorts: *actions*, *situations*, and domain *objects*. A special binary function

Proceedings of SIMPAR 2010 Workshops
Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS
Darmstadt (Germany) November 15-16, 2010
ISBN 978-3-00-032863-3
pp. 183-192

symbol $do : action \times situation \rightarrow situation$ exists, with $do(a, s)$ denoting the situation which arises after performing action a in situation s. The constant S_0 denotes the initial situation, i.e., the situation where no actions have occurred yet. The state the world is in is characterised by functions and relations with a situation as their last argument. They are called *functional* and *relational fluents*, respectively. The third sort of the situation calculus is the sort *action*. Actions are characterised by unique names. For each action one has to specify a *precondition axiom* stating under which conditions it is possible to perform the respective action and an *effect axiom* formulating how the action changes the world in terms of the specified fluents. For space reasons we will not go into the formal details here. To get an idea, it is enough to know that actions can only be performed when they are possible, i.e., their precondition axiom holds, and their effects are manifested in the environment. Finally, we need a so-called *basic action theory* [9] (BAT), which is a set of logical sentences containing axioms about action preconditions and effects, axioms about what is true in the initial situation and some further foundational axioms. For details we refer to [8, 9].

2.2 Readylog

READYLOG [2,3] is our variant of GOLOG [6] and also makes use of Reiter's BATs as described above. It has imperative control constructs such as loops, conditionals, and recursive procedures, but also less standard constructs like the nondeterministic choice of actions. READYLOG extends Golog by numerous features. For specifying the behaviours of an agent or robot the following constructs exist: (1) sequence $(a; b)$, (2) non-deterministic choice between actions $(a|b)$, (3) solve a Markov Decision Process (MDP) ($solve(p, h)$, p is a GOLOG program, h is the MDP's solution horizon), (4) test actions ($?(c)$), (5) event-interrupt ($waitFor(c)$), (6) conditionals ($if(c, a_1, a_2)$), (7) loops ($while(c, a_1)$), (8) condition-bounded execution ($withCtrl(c, a_1)$), (9) concurrent execution of programs ($pconc(p_1, p_2)$), (10) probabilistic actions ($prob(val_{prob}, a_1, a_2)$), (11) probabilistic (offline) projection ($pproj(c, a_1)$), and (12) procedures ($proc(name(parameters), body)$).

3 Qualitative Fluents and Fuzzy Controllers in Golog

In this section, we briefly go over our previous work on integrating fuzzy fluents and fuzzy controllers into Golog. For technical details we refer to [4, 5].

3.1 Fuzzy Fluents

The essence of qualitative representations is to find appropriate equivalence classes for a number of quantitative values and to group them together in these qualitative classes. Fuzzy set theory seems appealing as it avoids sharp boundaries of the classes: a quantitative value can be, for instance, in two classes at the same time, the transition between two neighbouring classes can be designed as being smooth. This characteristic can avoid problems every roboticist already

Proceedings of SIMPAR 2010 Workshops
Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS
Darmstadt (Germany) November 15-16, 2010
ISBN 978-3-00-032863-3
pp. 183-192

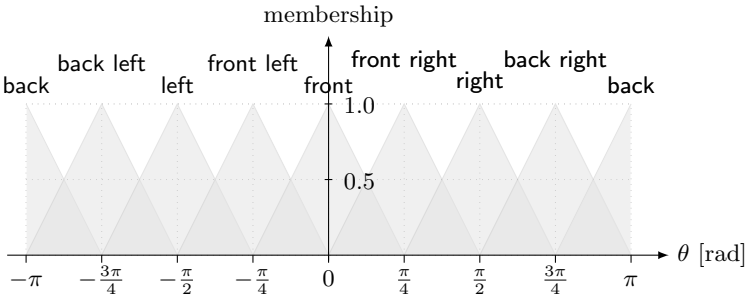


Fig. 1. Membership function for qualitative orientation at level 3

has experienced: sensor values oscillate between two categories resulting in awkward behaviour of the robot.

Our formalisation of fuzzy fluents is based on the idea to extend ordinary functional fluents with a degree of membership to a certain qualitative category. This means, for example, that the robot's distance to an obstacle can be *short* and *very short* at the same time. To account for the fuzzy border of the two categories, the degree of membership is, say, 0.5 for both categories. Another example is given in Fig. 1. The robot's orientation at an angle of $\frac{\pi}{8}$ is *front* and *front-right*, both with a membership degree of 0.5. Note that while we use triangular-shaped membership functions in this example, there are no limitations on the shape of the membership function. To use these fluents, one simply defines the different categories and membership values in the domain specification.

What is further needed in order to do reasoning with these kinds of fluents, is a routine that restores a quantitative value from a qualitative category, that is to *defuzzify* a category. In [4], we formalise a *centre-of-gravity defuzzifier* in the situation calculus. However, other defuzzifiers known from fuzzy set theory can easily be used as well.

3.2 Fuzzy Controller in Golog

Fig. 2 shows a schematic fuzzy controller. The quantitative sensor values $y(t)$, together with some reference input $r(t)$, which describes the vital state of the system, need to be fuzzified, i.e., the membership to a certain class needs to be determined. The *Inference Mechanism* uses these fuzzified input values together with a rule base of fuzzy rules to select the appropriate control output. The output as such uses fuzzy categories and thus must be defuzzified to serve as an input $u(t)$ for the real world (the control output). The output of the real world process serves as the sensor input for the next control step.

Proceedings of SIMPAR 2010 Workshops
Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS
Darmstadt (Germany) November 15-16, 2010
ISBN 978-3-00-032863-3
pp. 183-192

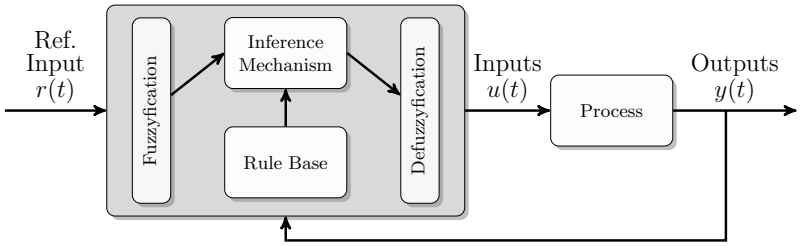


Fig. 2. General architecture of a fuzzy controller

To map this into Golog, we introduce a statement **fuzzy_controller** which takes a rule base as input and returns the control output (cf. also [5]):

$$
\begin{aligned}
&\textbf{fuzzy\_controller}(\\
&\quad \textbf{if } \phi \textbf{ then } \mathit{assign}(\mathfrak{f}, c_k); \cdots ;\\
&\quad \textbf{if } \psi \textbf{ then } \mathit{assign}(\mathfrak{g}, c_l);\\
&\quad \textbf{default}(\mathit{assign}(\mathfrak{f}, c_n); \mathit{assign}(\mathfrak{g}, c_m)))
\end{aligned}
$$

A fuzzy rule base in Golog is interpreted as follows. Each matching fuzzy rule will be replaced by its consequence, i.e., a special assignment statement, while non-matching ones contribute *nil*. The assignment statement $\mathit{assign}(\mathfrak{f}, c)$ mentioned above is a Golog action which assigns the qualitative category c to the fuzzy fluent $\mathfrak{f}$. As defuzzifier, we use the centre of gravity (*cog*). Depending on the assigned output category, control actions can be sent to the actuators. The condition of a rule can be a complex formula over fuzzy fluents stating for example: *is the object close and very close?* Sometimes, it may happen that no given rule in a controller block matches at all, nevertheless some output would be required. We therefore define an additional statement $\textbf{default}(\mathit{assign}(\mathfrak{f}, c); \ldots)$, which is interpreted in case the control output was the *nil* action after evaluating the rule base.

4 Applications in a Domestic Service Robotics Domain

In this section we give two examples for using fuzzy fluents and fuzzy controller in the domestic robot domain. We start with a brief description of the tasks. Before we show the example Golog programs, we define the required distance and orientation relations.

4.1 A Domestic Service Robotics Domain (RoboCup@Home)

In the RoboCup@Home competition *service and assistive robot technology* that is highly relevant for future personal domestic applications should be demonstrated. In the competition, the robots have to fulfil tasks such as:

***FollowMe!*:** the robot has to follow a human through the apartment;

Proceedings of SIMPAR 2010 Workshops
Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS
Darmstadt (Germany) November 15-16, 2010
ISBN 978-3-00-032863-3
pp. 183-192

***Fetch&Carry*:** a human names known objects and the robot needs to fetch them. The human may give hints such as: "The teddy is near the TV";

***Walk'n'Talk*:** in a guidance phase, a human instructor leads the robot around in an apartment and tells it certain landmarks such as "kitchen table", "TV set", or "fridge". In a second phase the robot is instructed to navigate to some of these just learnt places.

The rules of the RoboCup@Home competition state that a robot —to be successful in the competition— is to be endowed with a certain set of basic abilities, like navigation, person and object recognition, and manipulation. Furthermore, fast and easy calibration and setup is essential, as the ultimate goal is to have a robot up and running out of the box. Also, human-robot interaction has to be achieved in a natural way, i.e., interacting with the robot is allowed only using natural language (that is by speech) and gesture commands. As mentioned in the introduction, humans tend to make use of qualitative concepts such as *near* or *far*. With introducing suitable qualitative concepts, we bridge the gap between human and robot representations of domestic environments.

But not all parts of the solution of a domestic task require deliberation. For some decisions simple reactive controllers are sufficient. However, these reactive mechanisms also need to understand qualitative concepts. Here, we can make use of our embedding of fuzzy controllers in Golog. In the next sections, we show some specification examples.

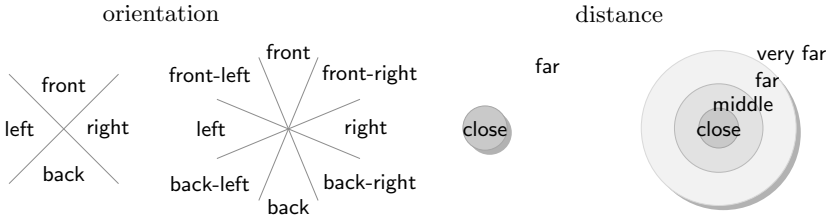
4.2 Qualitative Representations for Domestic Environments

One very important form of interaction between a human and a robot in the RoboCup@Home domain is to give the robot some hints where objects might be located. Based on Clementini, Felici, and Hernandez [1], we develop qualitative representations for positional and directional information that can be used to instruct the robot.

The position of a *primary object* is represented by a pair of distance and orientation relations with respect to a *reference object*. Both relations depend on a so-called *frame of reference* which accounts for several factors like the size of objects and different points of view. Different frames of reference can be classified into three basic types: they are *intrinsic*, if the relation is given by some inherent property of the reference object, *extrinsic*, if a relation is imposed by external factors such as motion, or *deictic*, if the orientation is given by the point of view from which the reference object is seen. In a domestic setting, we can likely infer the type of the frame of reference as well as a possible reference object from the context. If, for example, the user is referring to an object *to the left of the plant* we are able to instantiate the reference object with the plant, if the user is referring to something being *in front of the robot*, we conclude an intrinsic type and take the inherent *front* direction of *the robot*.

Orientation relations are used to describe where objects are placed relatively to each other. The frame of reference contains the *point of view* and is meant to fix the "front" side of the orientation relation. The point of view and the reference

Proceedings of SIMPAR 2010 Workshops
Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS
Darmstadt (Germany) November 15-16, 2010
ISBN 978-3-00-032863-3
pp. 183-192



(a) level 2 (b) level 3 (c) level 1 (d) level 3

Fig. 3. Different levels of granularity for orientation and distance according to [1]

object are connected by a straight line. The view direction is then determined by a vector from the point of view to the reference object. The location of a primary object is expressed with regard to the view direction as one of a set of relations. There are different levels of granularity. For example, for orientation on the first level, the point of view and the reference object are connected by a straight line such that the primary object can be to the left, to the right, or on that line. On the second level there would be four partitions, the third level would have eight, and so on, as shown in Figs. 3(a) and 3(b).

For the distance relation also different levels can be derived as shown in Figs. 3(c) and 3(d). An arbitrary level n of granularity with $n+1$ distinctions yields the set $Q = \{q_0, q_1, \ldots, q_n\}$ of qualitative distances. Given a reference object *RO* these distances partition the space around *RO* such that q_0 is the distance closest to *RO* and q_n the one farthest away. Further, a structure relation is needed in order to relate distances to each other (see [1] for details). As in the description of qualitative orientation, where the frame of reference was meant to fix the 'front' side of the reference object, we need to attend frames of reference for the distance relation, too. In the domestic settings we can define different distance relations according to: (1) external references such as the maximal size of the apartment: *"The plant is at the far end of the corridor"*; (2) intrinsic references used in relating objects to each other such as room or table: *"The cup is on the table close to the plate"* vs. *"The teddy is close to the TV"*; and (3) an appropriate distance system. In our domestic environment we suggest to make finer distinctions in the neighbourhood of the reference object than in the periphery. Hence, we can distinguish the scales *dist-scale* $\in \{\text{apartment}, \text{room}, \text{object}(o)\}$, where object o refers to objects such as *table*, or *bookshelf*.

Hence, we must provide a procedure *analyseHint*, which takes a hint given by the human instructor and distills the position of the object, the frame of reference as well as the scale from that hint. For instance:

- *"The plant is far on the left side of the corridor"*; the primary object is the plant, the point of view is the view point of the robot, the distance scale is set to the size of the corridor.
- *"The cup is on the table close to the plate"*; the primary object is the cup, the reference object is the plate, the distance scale is set to the size of the table. No orientation relation is given.

Proceedings of SIMPAR 2010 Workshops
Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS
Darmstadt (Germany) November 15-16, 2010
ISBN 978-3-00-032863-3
pp. 183-192

proc fetch_and_carry($object, hint$)
  analyseHint($hint$);
  $\pi(pos, for_\theta, for_{dist}).[ori\_type(for_\theta) \wedge dist\_system(for_{dist}) \wedge$
    $dist\_scale(for_{dist}) \wedge dist\_type(for_{dist}) \wedge object\_pos(pos)]?$;
  search($object$, pos, for_θ, for_{dist})
endproc

proc search($object$, pos, for_θ, for_{dist})
  solve(**while** ¬objectFound **do**
      pickBest($search\_pos$ = defuzzify(pos, for_θ, for_{dist}));
      lookForObjectAt($object$, $search\_pos$);
    endwhile, H) /* *end solve with horizon H* */
  pickup_and_return($object$);
endproc

Algorithm 1: A Readylog program making use of the qualitative notions for the "*Fetch&Carry*" test

– *"The teddy is close to the TV"*; the primary object is the teddy, the reference object is the TV, the distance scale should be set to the size of the room where the TV is located. Again, no orientation relation is given.

With this procedure at hand, we can adopt our fuzzy fluents for the qualitative distance and orientation. The membership function for the orientation fluent was given in Fig. 1. We can define the membership function for distance in a similar way. In the next section, we give an idea of how these fluents can be used for programming the robot.

4.3 Qualitative Notions in High-level Programs

Now that we have proposed an initial modelling of qualitative representations of positional information in a domestic setting we show how we can make use of these representations within our existing high-level control mechanism. Algorithm 1 shows a slightly abstracted version of a Readylog control program for the *Fetch&Carry* task.

The procedure *fetch_and_carry* takes the object that should be fetched and a user hint as input. At first, the action *analyseHint* is executed. This is a complex action which involves natural language processing. From the user phrase, the frame of reference for orientation and distance as well as the distance scale is extracted (as pointed out in the previous section). The action's effects axioms are changing fluent values for the fluents describing the orientation's frame of reference, the distance system, the distance scale, the distance's frame of reference as well as the qualitative position of the reference object. The next statement in the program is a so-called "*pick*" statement (π) which is used to instantiate the free variables in the logical formula in the next test action (denoted by the ?). The whole construct can be seen as an existential quantifier, and the effect is that the variables pos, for_θ, for_{dist} are bound. The next step is to call

Proceedings of SIMPAR 2010 Workshops
Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS
Darmstadt (Germany) November 15-16, 2010
ISBN 978-3-00-032863-3
pp. 183-192

proc follow_me_rulebase
  fuzzy_controller(...;
    if $\text{is}_\star(\text{dist}_{\text{user}}, \text{close}, \text{speed}_{\text{user}}, \text{slow})$ **then** $assign(\text{speed}_{\text{robot}}, \text{slow})$;
    if $\text{is}_\star(\text{dist}_{\text{user}}, \text{far}, \text{speed}_{\text{user}}, \text{medium})$ **then** $assign(\text{speed}_{\text{robot}}, \text{fast})$;
    ...; **default**$(\text{speed}_{\text{robot}}, \text{medium})$)
) ;`/* end fuzzy_controller */`
  applySpeed()
endproc

Algorithm 2: A fuzzy controller for the "*FollowMe!*" test

the search routine with these parameters. The search involves the activation of decision-theoretic planning (*solve*) at a position where the object is meant to be according to the user's hint. The position is defuzzified, taking the frame of reference information into account. That is, the position based on the distance scales and the quantitative orientations given the points of view etc can now be calculated. The action *lookForObject* again is a complex action which actually tries to seek the object.

4.4 Domestic Golog Fuzzy Controllers

As detailed in Sect. 3.2 we integrated fuzzy controllers in Golog in [5]. If (a part of) a task does not require high-level decision making (decision-theoretic planning as used in the previous section), but can instead be solved with a reactive mechanism it may still be convenient to make use of the qualitative representations. One example in the domestic setting is the "*FollowMe!*" test. The control of the follow behaviour can be modelled quite straight-forwardly.

In the following we show a simple rule base that could be used to solve the *FollowMe!* task. The rule base for this test could look like Alg. 2. As we stated in Sect. 3, a rule base consists of a number of if-then rules where the antecedent as well as the consequence mention fuzzy fluents. So, the first rule reads as follows: "*if the distance to the user is close and its speed is slow, then set the robot speed to slow*", the second rule reads "*if the distance to the user is far and its speed is medium, then set the robot speed to fast*", where user is the person to be followed. The $is_\star$ predicate is defined in [5] and denotes the conjunction of the fuzzy fluents $dist_{user}$ and $speed_{user}$, If neither condition applies, the default speed selection is set to medium. Finally, the $speed_{robot}$ fuzzy fluent has to be defuzzified, that is, a quantitative value is calculated for the qualitative class. Then, we can apply the quantitative speed to the robot motors.

5 Conclusions

In this paper, we presented an approach on how high-level robot controllers could deal with qualitative representations for domestic environments. For robot competitions such as RoboCup@Home this is useful, as the robot needs to be

Proceedings of SIMPAR 2010 Workshops
Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS
Darmstadt (Germany) November 15-16, 2010
ISBN 978-3-00-032863-3
pp. 183-192

instructed by a human operator by natural language. Having qualitative representations in place allows for more human-like instructions as humans tend to use qualitative (spatial) representations such as *far* or *left-of*. In our previous work, we defined qualitative fluents in the situation calculus based on fuzzy sets. This allows us to define qualitative fluents in a well-founded way. Particularly, it gives a semantics to derive quantitative values from qualitative categories and vice versa. Further, we proposed a semantics for fuzzy controller in Golog. Both, the definition of fuzzy fluents and fuzzy controllers, allows us to write programs mentioning qualitative values in a straight-forward way. For the RoboCup@Home tasks *Fetch&Carry* and *FollowMe!* we showed example implementations, how qualitative representations and fuzzy controllers could be beneficially deployed. While these programs only reflect first ideas of deploying fuzzy fluents and fuzzy controllers in domestic robot applications, we are aiming at implementing these controllers for our future work on our domestic robot platform.

References

1. Clementini, E., Felice, P.D., Hernandez, D.: Qualitative representation of positional information. Artificial Intelligence 95(2), 317–356 (1997)
2. Ferrein, A.: Robot controllers for highly dynamic environments with real-time constraints. Künstliche Intelligenz 24(2), 175–178 (2010)
3. Ferrein, A., Lakemeyer, G.: Logic-based robot control in highly dynamic domains. Robotics and Autonomous Systems, Special Issue on Semantic Knowledge in Robotics 56(11), 980–991 (2008)
4. Ferrein, A., Schiffer, S., Lakemeyer, G.: A fuzzy set semantics for qualitative fluents in the situation calculus. In: Proc. ICIRA-08. LNCS, vol. 5314, pp. 498–509. Springer (2008)
5. Ferrein, A., Schiffer, S., Lakemeyer, G.: Embedding fuzzy controllers into golog. In: Proc. of the IEEE Int'l Conf. on Fuzzy Systems (FUZZ-IEEE'09). pp. 498–509. IEEE (August 20-24 2009)
6. Levesque, H.J., Reiter, R., Lesperance, Y., Lin, F., Scherl, R.B.: GOLOG: A logic programming language for dynamic domains. J. of Log. Progr. 31(1-3) (1997)
7. McCarthy, J.: Situations, Actions and Causal Laws. TR, Stanford University (1963)
8. Pirri, F., Reiter, R.: Some contributions to the metatheory of the situation calculus. Journal of the ACM 46(3), 325–361 (1999)
9. Reiter, R.: Knowledge in Action. Logical Foundations for Specifying and Implementing Dynamical Systems. MIT Press (2001)
10. Schiffer, S., Ferrein, A., Lakemeyer, G.: Football is coming home. In: Chen, X., Liu, W., Williams, M.A. (eds.) Proc. Int'l PCAR Symposium. Univ. of Western Australia Press (2006)
11. Schiffer, S., Niemüller, T., Doostdar, M., Lakemeyer, G.: AllemaniACs@Home 2009 Team Description. In: Proceedings CD RoboCup 2009. Graz, Austria (2009)
12. van der Zant, T., Wisspeintner, T.: RoboCup X: A Proposal for a New League Where RoboCup Goes Real World. In: Bredenfeld, A., Jacoff, A., Noda, I., Takahashi, Y. (eds.) RoboCup. LNCS, vol. 4020, pp. 166–172. Springer (2005)
13. van der Zant, T., Wisspeintner, T.: Robotic Soccer, chap. RoboCup@Home: Creating and Benchmarking Tomorrows Service Robot Applications, pp. 521–528. I-Tech Education and Publishing (2007)

Proceedings of SIMPAR 2010 Workshops
Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS
Darmstadt (Germany) November 15-16, 2010
ISBN 978-3-00-032863-3
pp. 183-192