Context and motivation in coordinating fuzzy behaviors

Andrea Bonarini, Giovanni Invernizzi, Thomas Halva Labella, Matteo Matteucci
Politecnico di Milano AI and Robotics Project
Dipartimento di Elettronica e Informazione – Politecnico di Milano
E-mail: {bonarini, matteucc}@elet.polimi.it *

Abstract

We propose an architecture to implement coordination among fuzzy behaviors for autonomous robots, in real–time tasks. We consider a four–layer cognitive reference model to define the knowledge flow from sensor to behavior. Behaviour coordination is obtained by fuzzy context conditions evaluated on information provided by intelligent sensors (1^{st} layer) and a world modeler (2^{nd} layer). In our architecture it is possible to define *activation conditions* and *motivations* to coordinate fuzzy behaviors (3^{rd} layer) to achieve *local goals.* More complex tasks with multiple, interacting goals, can be accomplished reasoning on abstract, symbolic models (4^{th} layer). We adopt this system in different applications such as robotic soccer (Robocup), document delivery in office environments, and surveillance.

Keywords. *Behavior-based Robotics, Cognitive Robotics, Fuzzy Systems, Robotic Architecture.*

1 Introduction

A general architecture for the definition of the cognitive apparatus for autonomous systems [10] consists of a four–layered model. We consider this model as a reference for the definition of our behavior architecture and we focus in this paper on the implementation of the knowledge processing level where inference on *symbolic concepts*, derived from lower levels and represented by fuzzy predicates, produces actions for an autonomous agent. We have implemented this mechanism by independent modules each corresponding to a *fuzzy behavior.*

The *activation conditions* for each behavior are fuzzy predicates which should be verified in order to enable the corresponding behavior; we call these predicates *CANDO conditions.* Coordination among behaviors enabled at same time is implemented by a different set of predicates which represent *motivations* to actually execute the action proposed by each inference module (*WANT conditions*).

We propose to consider both CANDO and WANT conditions since they correspond to semantically different aspects. CANDO conditions are intrinsically related to the behavior definition: if they are not verified the behavior activation does not make sense. WANT conditions are context dependent, they can represent either goals for the agent or environmental situations.

We represent symbolic concepts by fuzzy models to face the issue of uncertain perception; a fuzzy model implements a classification of the available information and knowledge in terms of fuzzy predicates, which have been demonstrated to be a powerful and robust representation paradigm [7][12].

We adopt this architecture on different robots involved in a wide variety of tasks, such as: soccer playing in Robocup [2], document delivery in an office environment, and surveillance.

2 The reference model

In this section, we summarize the main features of the above mentioned cognitive reference model, highlighting how such model can be instanced in a complete architecture for autonomous robot control.

The four layers of the cognitive model are:

- *data layer:* extracts from raw data streams basic features as a basis of eventual understanding; this layer is the only agent's interface to the environment (the other layers work on features extracted by this one)
- *concept layer:* interprets and abstracts features to basic concepts and notions which are at a higher level of representation (symbolic concepts)

*This research has been partially supported by the MURST Project CERTAMEN.

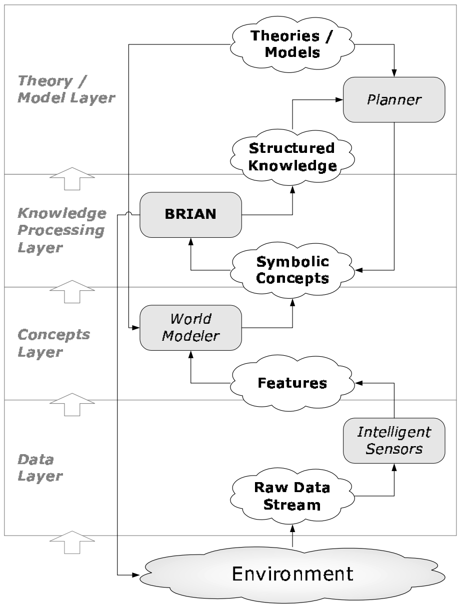

Figure 1: The agent architecture

- *knowledge processing layer:* processes symbolic concepts to obtain more complex information to be used for planning actions, communicating and other reasoning activities
- *theory/models layer:* contains structured abstract models and theories

Our architecture is mapped on this cognitive model (Figure 1).

Intelligent sensors implement the data layer. For instance, omnidirectional vision [4] provides relative positions and distance of colored objects in the environment in our Robocup application, and the angular position of vertical edges in our service robotics applications.

These features are interpreted in terms of fuzzy, ground predicates to generate the symbolic concepts of the second layer (e.g., "the ball is close", "the door is on the left"). This is done according to a cognitive background model which is conceptually part of the fourth layer. A *World Modeler* operates at the second layer to produce a conceptual model of the environment and to infer symbolic concepts from data features using theoretical models (4^{th} layer).

Complex predicates are built from the ground ones to describe less basic aspects such as: "I have the possess of the ball", "I'm in a good position to kick". These predicates, together with the ground ones are used as CANDO and WANT conditions, and to infer actions, by the behavior engine *BRIAN* (*Brian Reacts by Inferring ActioNs*).

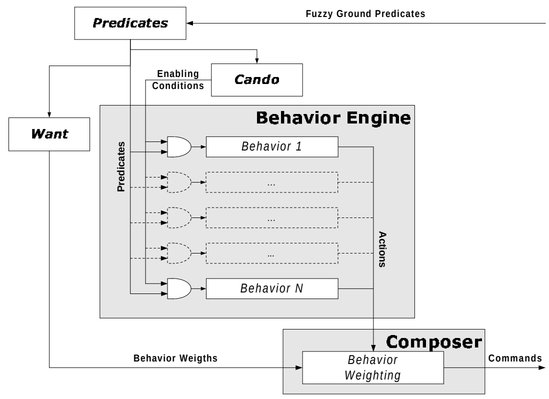

Figure 2: The behaviour management system

The *Planner* module reasons on models on the 4^{th} layer to produce goals as symbolic concepts for BRIAN, which are included in the behavior motivations. These models are abstract maps, graph representations, or state diagrams, used to forecast or simulate future actions.

In this architecture, BRIAN implements local, reactive behaviors whose activation is ruled also by strategic directions coming from the Planner through the WANT conditions. In other terms, BRIAN implements a kind of local reasoning influenced by context information and strategic information, in turn implemented by the planner (global reasoning). This, while generating strategic directions, should solve possibly arising conflicts between multiple goals.

3 BRIAN: the behavior management system

BRIAN, our *behavior management system*, shown in details in figure 2 uses fuzzy predicates to represent the activation conditions, the motivation conditions and internal knowledge. Fuzzy predicates are a general and robust [7] modeling paradigm, close to the designer's mental models [13]. Other researchers [12][8] have adopted them to implement control systems for autonomous robots for analogous reasons. Fuzzy predicates may represent aspects of the world, goals, and information coming from other agents. Their general shape consists of a fuzzy variable name, a label corresponding to a fuzzy set defined on the range of the variable, a degree of matching of the corresponding data to the mentioned fuzzy set, and a reliability value to take into account the quality of the data source. For instance, we may have a predicate represented as

$$< BallDistance, VeryClose, 0.8, 0.9 >$$

which can be expressed as: "the ball is considered very close, with a truth value of 0.8 (coming from the fuzzyfication of the incoming data, namely the real-valued distance from the ball), and a reliability value of 0.9,which qualifies the data as highly reliable".

We consider ground and complex fuzzy predicates. *Ground fuzzy predicates* range on data available to the agent, and have a truth value corresponding to the degree of membership of the incoming data to a labeled fuzzy set. This is equivalent to classify the incoming data into categories defined by the fuzzy sets, and to assign to this classification a weight between 0 and 1. Fuzzy ground predicates are defined on features elaborated by the world modeler and goals from the planner; the reliability of data is provided, respectively, by the world modeler (basing on feature analysis), and the planner stating the goals.

A *complex fuzzy predicate* is a composition (obtained by fuzzy logic operators) of fuzzy predicates. Complex fuzzy predicates extend the basic information contained in ground predicates into a more abstract model. In RoboCup, for instance, we can model the concept of ball possession by the $HaveBall$ predicate, joining by the AND operator the ground predicates

$< BallDistance, VeryClose >$ and

$< BallDirection, Front >$,

the first deriving from the fuzzyfication of the ball distance perception, the second from the fuzzyfication of the perception of ball direction.

We define for each behavior module a set of predicates that enable its activation: the *CANDO preconditions*. The designer has to put in this set all the conditions which have to be true, at least to a significant extent, to give sense to the behavior activation. For instance, in order to consider to kick a ball into the opponent goal, the agent should have the ball control.

Another set of fuzzy predicates is associated to each behavior module: the *WANT conditions*. These are predicates that represent the motivation for the agent to activate a behavior in relation with a context. They may come either from the environmental context (e.g., "there is an opponent in front of me"), or from strategic goals (e.g., "I have to score a goal"). All these predicates are composed by fuzzy operators, and contribute to compute a motivational state for each behavior. The agent knows that it could play a set of behaviors (those enabled by the CANDO conditions), but it has to select among them the behaviors consistent with its present motivations. We have two possibilities, to be selected according to the specific situation: we can either combine the actions proposed by the selected behaviors, weighted by the respective motivations, or select the best motivated behavior, and the corresponding action. The first approach is followed by the majority of the proposed fuzzy behavior management systems presented in literature [11] [12] [8], since it is analogous to the traditional way of composing output from fuzzy rules; an analogous method is adopted also in other non–fuzzy architectures [1]. However, at least in principle, while designing (or learning) behaviors, all the possible interactions with other behaviors should be taken into account since the vectorial combination of two actions may produce undesired effects (think, for instance, at the action that can result from the combination of the actions proposed by the *AvoidObstaclesFromLeft* and the *AvoidObstaclesFromRight* behaviors). In principle, this design approach is in contrast with the behavior–independency principle, fundamental in the behavior–based approach to robot control design. The second approach, that is selecting the action proposed by the best fitting behavior, prevents the possibility to pursuit multiple goals in parallel. Both the approaches can be implemented in our architecture; we decided to design the behaviors we will show in section 5 adopting a mid-way approach: we compose the actions proposed by each behavior, but we carefully designed the motivation conditions in order to avoid the activation of two incompatible behaviors at the same time.

4 Related work

Our behavior architecture is quite different from the original Brooks's [6] one: our behavior management system works on fuzzy predicates obtaining the capability of coordinating the concurrent execution of several behaviors. We consider the adoption of CANDO and WANT predicates as an alternative choice to the implementation of the subsumption architecture. It is more general than Brook's proposal and also more effective in strongly dynamic environments. In our implementation the enabling connections among behaviors are context dependent, so relationships among behaviors are not rigidly defined, but we can adapt the emerging global behavior, depending also on external conditions and motivations.

Another reference we have considered is Arkin's schema-based behavior architecture [1]. It is possible to map the basic principles of our and Arkin's approaches into each other. The main difference is the fuzzy model we put at the basis of our architecture, whereas the analogous features are represented by Arkin by different tools such as a *gain* value to weight each behavior contribution, and their representation in terms of potential fields. However the gain has a different meaning, stating the a priori relevance of a behavior with respect to another, while we blend (or select) behaviors according to their condition values and context.

Another difference between our architecture and both Brooks' and Arkin's is the presence in ours of a world modeler that interfaces the external world with the behavior module. In the mentioned architectures, world modeling is embedded in each behavior definition. We have implemented such a module to achieve efficiency and to provide a unified interface from the environment to the behaviors.

HEIR [9] is an architecture whose accent is on the role played by different kind of knowledge: symbolic knowledge is used to reason and plan actions, whereas diagrammatic knowledge is used to maintain a representation of the situation. A behavior level is also present, but behaviors are dedicated to the low-level implementation of actions, able to react to eventual, unexpected events. We also have modules in charge of planning and modeling the environment, but our behaviors play a more complex role, actually implementing also complex activities, triggered by the context (evaluated on the world model) and by the goals defined by the planner.

In our architecture we start from Saffiotti's approach [12][8]to the use of fuzzy logic in robotics [11], which tries to face the problem of designing an effective controller for mobile robots by combining goal-specific strategies by resolution of conflicts between multiple objectives. We keep separate the CANDO from the WANT conditions, due to their different semantics, while in [12] they are put together in the *desirability* parameter.

5 Experimental results

We have applied our approach both in service robotics, and in robot soccer playing. Here, we focus on this last, since a Robocup match provides a rich and challenging environment where our approach clearly shows its effectiveness. Our players Rullit and Rakataa (see figure 3) are part of the Italian national soccer team ART (Azzurra Robot Team). We implemented ten fuzzy behaviors whose composition enables the robots to play effectively in a match, fighting for the possess of the ball, avoiding other robots and fouls, kicking in the opponent goal, and taking a defense behavior when the own goal-keeper has problems.

We can identify subsets of behaviors which are intended to be activated in sequence, and their conditions have been designed to avoid interference. For instance, *AlignToBall* aligns the robot to the direction of the line between the ball and the opponent goal (all our robots have an omnidirectional vision system, so they almost always know where the ball and the goal are); *GoToBall* brings the robot on the ball when it is in the forward direction. The CANDO conditions for *AlignToBall* include the fact the the ball is visible and that the robot is not aligned nor controls it. Notice how these are essential conditions for the behavior. Among the WANT conditions for *GoToBall* we have that the ball should be in the forward direction. The *AlignToBall* behavior tends to make this condition true, and, when this is the case, the context is favorable to the activation of *GoToBall*. Both the behaviors cooperate to bring the agent in a position from where it can take the ball and bring it towards the goal. The *GoToGoal* behavior has among its CANDO conditions predicates corresponding to have the control of the ball and to be aligned to ball and goal: so it can be activated only when both the other mentioned behaviors have achieved their goals.

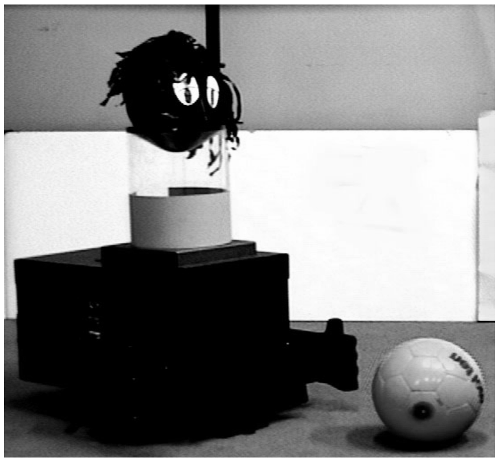

Figure 3: Our robot Rullit

We have also defined another kind of behaviors to take care of the integrity of the robot: these inhibit the others in order to handle critical situations. They are devoted to solve possible problems such as the presence of obstacles or walls in the desired directions. As mentioned above, we have designed the behavior conditions to implement exclusive activation. In particular, all the WANT conditions of the incompatible behaviors contain the fact that none of the avoid behaviors should be active. In this way, if the robot has to avoid something, it does this without interference with the other behaviors.

Interesting behaviors emerge from the interaction of behaviors belonging to the two sets. For instance, we have seen Rakataa dribbling a couple of opponents due to the appropriate switching between *AvoidObstacle*, *Align* and *GoToGoal*. The co-operation of these behaviors made the robot react, when facing an opponent, by throwing the ball aside (obtained by a fast rotation decided by the *Avoid* behavior in order to get around the obstacle) and running to catch it again (composition of *Align* first and *GoToGoal* then).

A second emergent behavior is a sort of defense behaviour made by the co-operation between *AvoidWall*,*AvoidObstacle* and *Defense* behaviors. When the ball is close to the wall with another robot that is trying to catch it, our robot stays still covering our goal watching the other robot and waiting that it resolves the situation, since *AvoidWall* prevents the robot to get close to the wall. Once the opponent brings the ball away from the wall, Rakataa is ready to close it on the wall again, by applying the *Defense* behavior.

The third emergent behavior we mention here makes the robot catch the ball near an opponent or a wall. It is obtained by the co-operation between *Align* and *AvoidWall* or *AvoidObstacle*. When the ball is close to a wall (without any other robot in the neighborhoods) or close to another robot, the switching between behaviors makes the robot approach slowly the ball with fast and impulsive rotations. When it reaches it, the result is a sort of kick with the robot's hands that throws the ball away from the obstacle.

6 Conclusion

We have presented in this paper a general behavioral architecture for autonomous robots based on fuzzy models. We have focused on the behavior management module, whose most relevant feature is the use of fuzzy enabling and context conditions. It has been designed to make it possible different types of interaction among behavior modules. Fuzzy models give robustness to the system with respect to imprecision in data acquisition and uncertainty. Moreover, it enables smooth transitions between different behaviors, a property highly desirable in learning and adaptation [3].

We believe that learning may support the development and adaptation of robotic agents. In particular, we have shown elsewhere [5] how it could be possible to adapt behaviors in a short time to new environments by tuning their context predicates. The architecture here presented is essential in this activity, since it makes distinct the different aspects (CANDO, WANT, and behavior code) which can be separately designed, learned or adapted.

References

[1] R. C. Arkin. *Behavior-Based Robotics.* MIT Press, Cambridge, MA, 1998.

[2] M. Asada, H. Kitano, I. Noda, and M. Veloso. Robocup: today and tomorrow – what we have learned. *Artificial Intelligence Journal*, 110:193–214, 1999.

[3] A. Bonarini. Reinforcement distribution to fuzzy classifiers: a methodology to extend crisp algorithms. In *IEEE International Conference on Evolutionary Computation – WCCI-ICEC'98*, volume 1, pages 51–56, Piscataway, NJ, 1998. IEEE Computer Press.

[4] A. Bonarini, P. Aliverti, and M. Lucioni. An omnidirectional vision sensor for fast tracking for mobile robots. *IEEE Transactions on Instrumentation and Measuring*, page In press, 2000.

[5] A. Bonarini and M. Matteucci. Learning context motivation in coordinated behaviors. In *Proceedings of IAS-6*, page In Press, Amsterdam, NL, 2000. IOS Press.

[6] R. A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robot Automation*, 2(1), 1986.

[7] G. J. Klir, B. Yuan, and U. St. Clair. *Fuzzy set theory: foundations and applicatons.* Prentice-Hall, Upper Saddle River, NY, 1997.

[8] K. Konolige, K. Myers, E. Ruspini, and A. Saffiotti. The saphira architecture: A design for autonomy. *Journal of Experimental and Theoretical Artificial Intelligence*, 9(1):215–235, 1997.

[9] M. Piaggio. HEIR — A non hierarchical hybrid architecture for intelligent robots. *Lecture Notes in Computer Science*, 1555:243–254, 1999.

[10] F. J. Radermacher. Cognition in systems. *Cybernetics and Systems*, pages 1 – 41, 1996.

[11] A. Saffiotti. The uses of fuzzy logic in autonomous robot navigation. *Soft Computing*, 1(1):180–197, 1997.

[12] A. Saffiotti, K. Konolige, and E. H. Ruspini. A multivalued-logic approach to integrating planning and control. *Artificial Intelligence Journal*, 76(1-2):481–526, 1995.

[13] L. A. Zadeh. Making computer think like people. *IEEE Spectrum*, pages 26–32, 1984.