Milan Robocup Team 2003

Andrea Bonarini[1], Matteo Matteucci[1], Marcello Restelli[1], and Domenico Sorrenti[2]

[1] Politecnico di Milano Artificial Intelligence and Robotics Lab
Department of Electronics and Information
Politecnico di Milano, Milan, Italy
{bonarini,matteucc,restelli}@elet.polimi.it
[2] Department of Information, Systems and Communication
Università di Milano - Bicocca, Milan, Italy
sorrenti@disco.unimib.it

Abstract. We present some research issues faced by our multi–university team. We have faced almost all the aspects of the design and implementation of RoboCup MSL robots. In this paper, we focus on the behavior-based fuzzy control architecture and the aspects related to artificial vision and self-localization. We propose a fuzzy cognitive model to integrate coordination, planning and reactive behaviors in a team of cooperating robots. Behavioral modules are used as high-level macro-actions that compose structured plans defined by a flexible multi-agent coordination system. The control model is designed to be tuned and adapted on-line so that the team strategies and the role of robots in the control schemata can be automatically modified to face different opponent teams, and changes in robot performances. The use of a unifying cognitive model provides an effective tool for seamless integration of heterogeneous members in the team, gaining as much as possible from different skills and competencies. As far as vision is concerned, we have worked on the design of specific omnidirectional sensors, in particular on the design of mirrors that could implement the desired distribution of resolution. Finally, we present our self-localization tool able to merge information coming from different sensors.

1 Introduction

The Milan Robocup Team (figure 1) has faced in the last years many research issues, including: catadioptric sensor design [6], image analysis for omnidirectional vision [1], sensor fusion [3] self-localization [8], behavior definition and management [2], team coordination [4], learning and adaptation [3]. Moreover, we have faced a large number of technical problems to implement our robots in many fields, from Mechanics to Control, to Electronics, to Sensors, to Artificial Intelligence and Soft Computing. We only mention here the development from scratch of *Reseghè*, our omnidirectional robot for 2003, equipped by 4 omni–wheels, and able to change shape to face the game situations.

In this paper, we focus on the three aspects we have faced this year that are more interesting for the research results produced, namely: the control architecture, the implementation of omnidirectional vision sensors, and self-localization.

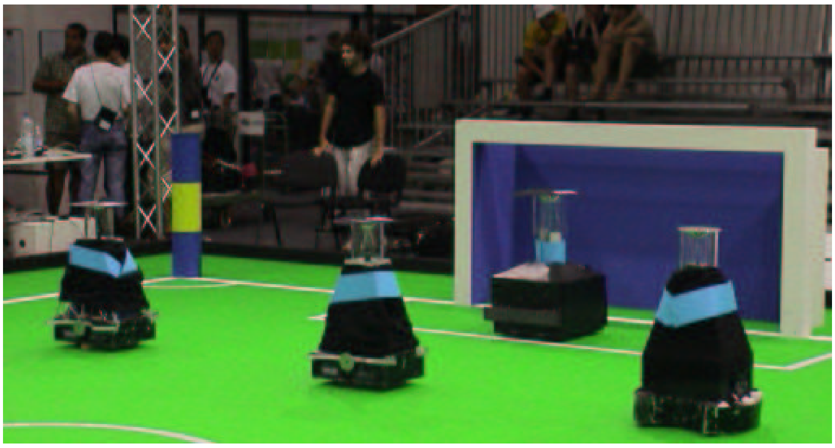

Fig. 1. The MRT RoboCup team on the field

2 Control Architecture

In this section we present the model we adopt to coordinate behaviors within and among robots. The main goals for the design of this model have been: uniform representation of concepts (percepts, goals, ...), integration among planning and reactive modules, easy design of behavioral and strategic modules, easy application of learning and adaptation techniques to learn parts of the models and adapt both single robot and team behavior to dynamical environments. The model we have implemented fulfills all these requirements and we have adopted it in Robocup, in service applications, and in space applications, as well.

2.1 The model

We base our model on the representation of concepts as *fuzzy predicates* [5], i.e., logical predicates that map real-valued variables into the truth interval [0..1]. Fuzzy predicates give the possibility to interpret in linguistic and conceptual terms both goals, and data coming from sensors and teammates. Each agent builds a world map by merging data from different sources. The map manager can answer queries by evaluating fuzzy predicates [3] on data contained in the map.

2.2 Mr.BRIAN: the behavior manager

A robot is governed by a set of behavior modules [2], presently implemented as a set of fuzzy rules each, and supported by the fuzzy behavior management system *Mr.BRIAN* (*Multi–level ruling BRIAN Reactively Implements AgeNts*). Fuzzy predicates in the antecedents of the rules are evaluated to weight the actions proposed by the consequents.

A set of *CANDO conditions* is associated to each behavior; they are represented by fuzzy predicates whose truth value is taken as an estimation of the applicability of the behavior in the faced situation. If applicability is above a given threshold, the rules of the behavior module are triggered and the corresponding actions are proposed. Each action is associated to a weight computed as fuzzy composition between the applicability and the weight coming from the fuzzy rule evaluation. In general, several behaviors are activated at a time, so we may have many proposed actions. These are weighted by the *WANT conditions*, fuzzy predicates, associated to each behavior, that state the opportunity to do the actions proposed by a behavioral module. WANT conditions include fuzzy predicates evaluated on the present state of the world, and also conditions coming from the coordination module described below. At the end of the evaluation process, the proposed, weighted actions are composed to produce the actual commands to the actuators. In Mr.BRIAN can also define a hierarchy among behavioral modules, so that modules at a higher level can reason on the actions proposed by modules at lower levels to eventually modify them. For instance, the AvoidObstacle behavior module reasons about the actions proposed by GoToBall, or by KeepBallInField, and modifies them to achieve the goal of avoiding perceived obstacles, by keeping the behavior of the robot as close as possible to the original intentions.

2.3 SCARE: the coordination module

SCARE, (*SCARE Coordinates Agents in Robotic Environments*) [4] coordinates the activity of the agents within the team. It assigns agents to *activities*. An activity is either a *job* or a *schema*. A job is a simple activity with a given goal which may be achieved, in principle, in different ways, by different behavior modules. A schema is a complex activity, made of temporally and/or logically constrained jobs, which can be assigned to different agents. To each activity are associated fuzzy predicates to evaluate different aspects: when an activity can take part to the assignment process (*cando*), whether the situation gives to the agent good possibilities of success (*chance*), whether the activity is useful to the team in the present situation (*utility*), when the activity can be considered as successful (*success*), when it has to be considered failed(*failure*). SCARE ranks the activities by evaluating these fuzzy predicates by querying the world map manager, and also by considering a set of parameters that describe synthetically the characteristics of the different agents. One of the goals of SCARE is to make the most of the available resources. One of our research goals is to integrate heterogeneous robots in an effective team. After a first *decision making phase*,

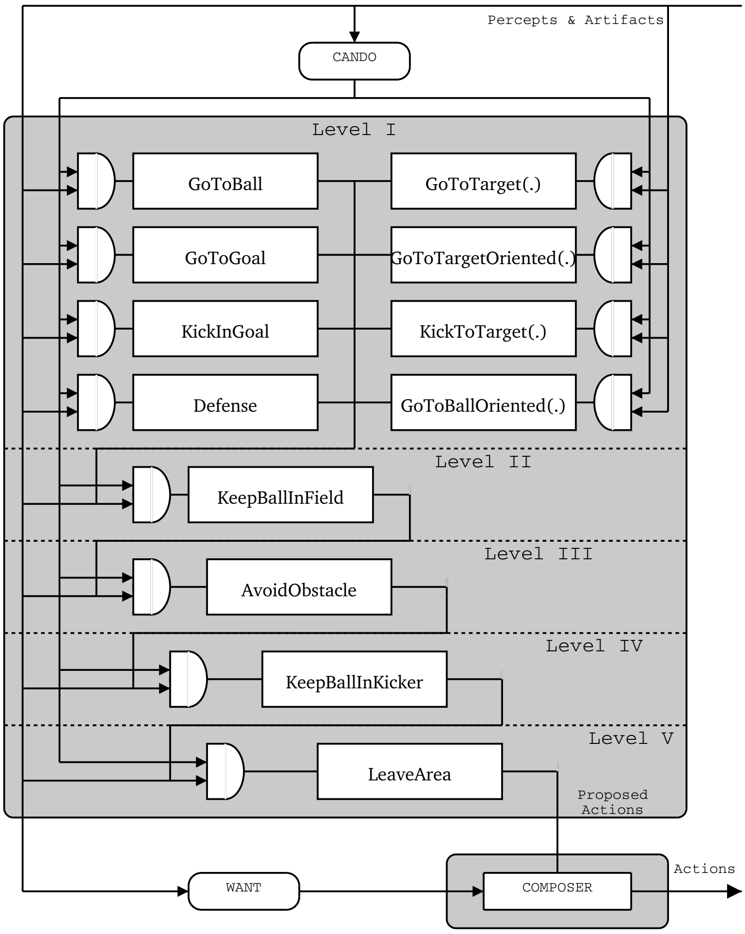

Fig. 2. The behavior configuration used in RoboCup

we have a rank of activities and potential assignments. Then, a *coordination phase* selects the best assignments considering also that schemata need to be assigned to appropriate sets of agents. At the end of this process, each agent is assigned to a job and new goal predicates are produced to drive the behavior system according to the the SCARE decisions. A copy of SCARE is instantiated on each agent. When agents cannot communicate, the local SCARE instance cannot assign activities to other agents and can only coordinate activities for the single agent. As soon as some communication link becomes active, one of the communicating instances of SCARE takes the role of leader and works for all the connected agents. This mechanism adapts the coordination activity to face the problems arising with degradation of communication links, which have been experienced also in a quite structured environment such as Robocup.

As a last point, we mention that SCARE has been designed to be adapted on-line to a dynamical environment. We have already written that the characteristics of agents influence the activity assignment process. This implies that the coordination system changes its behavior when agents change characteristics, for instance, when their speed decreases due to battery consumption, or when they are no longer able to kick due to lack of compressed air. Other decisions depend on predicates evaluated on the opponent's performance, and this gives the ability to adapt to different kind of opponents. For instance, if the opponents are faster than us, and the ball is quite far away, it's better to take a good defensive position instead of running hopelessly to the ball, but if they are quite slow, it makes sense to try to get the ball possess.

3 The vision sensor

We have panoramic vision systems on all our 2003 robots, as in the previous years.

We developed software for mirror design and machined different mirrors. The main scientific contributions are in the "design after the accuracy requisites of the application" area, both at the pixel and the part level [6]. We claim this area started after our previous work. We have designed an isometric mirror (see the produced image in figure 3 preserving lines and angles on the field plane, but we have not used it in the competitions since it had required t0 redesign the image analysis system, and this couldn't be done on time. Thus, we have only redesigned the old mirrors according the new principles, but also keeping the old image processing approach, briefly described here below.

The design of the mirrors we used this year takes into consideration two main requirements:

- the capability of perceiving the flag posts for localization purposes
- high accuracy close to the robot

We used two-part mirrors: the profile starts in the middle with an arc of circumference that joins a segment. The diameter of the mirror is of 10cm and its top part is placed at a height of 78cm. Our omnidirectional vision sensors (see image in figure 4are able to perceive a point up to a distance of 720cm and a height of 40cm. In this way, our robots can perceive the first color transition of at least two flag posts from everywhere in the field. We impose that the join between the two parts reflects points at a distance of 350cm from the ground, in order to achieve a good balance between accuracy near the robot and the possibility to perceive far objects. The arc of circumference is designed so that the robot is reflected only in a small area of the image, so that we can exploit the rest of it to get higher resolution.

3.1 Image processing

For what concerns perceptions our system is currently well suited for spatial and temporal constancy of lighting conditions. It performs a color classification,

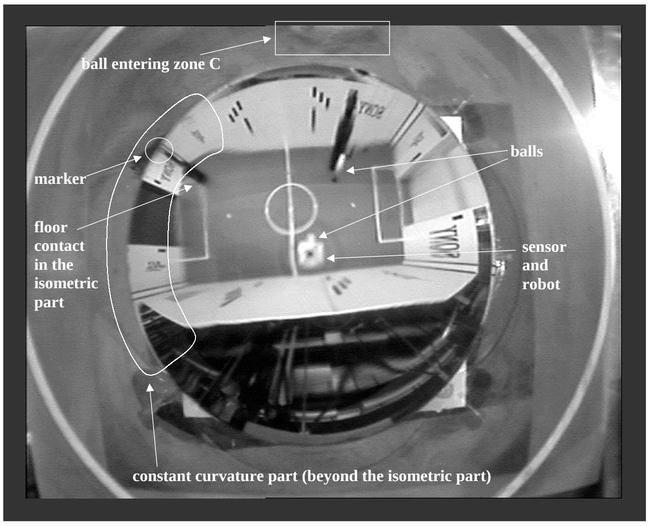

Fig. 3. The Image taken by the isometric mirror. Notice that the straight lines are straight and square angles appear as square angles.

which is similar to what other teams do. This step, still not published, allows maximum flexibility in setting up the mapping between the color space and the color labels. By maximum flexibility we mean that the area in the color space which is mapped to a single color can have any arbitrary shape, approximated by hypercubes as small as desired. This means that it is not constrained to be, e.g., an angular sector in the hue-saturation space.

Our approach is based on a specific image sub-sampling scheme [1], that is well suited and natural for omnidirectional images. The samples lay along many circles, at the same orientation, at different distances from the image symmetry center; it is a regular sampling in polar coordinates. Samples are taken as "receptors", i.e., cross shaped sets of pixels (3 x 3 each), whose color information is averaged. This provides also a sort of filtering of noise in the image.

The color classification takes place on the sub-sampled image only. After this delicate step we have a two-fold processing. The first one aims is based on conventional (color) connectivity analysis, aiming at the detection of some feature of the play, like goals, black objects (robots, referee, robot attendants, etc.), ball. The second one, which has been introduced this year, aims at detecting some playground features which are useful for self-localization. It is similar to the processing presented by [9], in the fact that it performs a radial scan of the image, at different directions. This comes natural for us, given the sub-sampling scheme adopted. The performance of our image processing system resulted "not satisfying" on field "A" in Padova 2003 (where we had all matches), we think

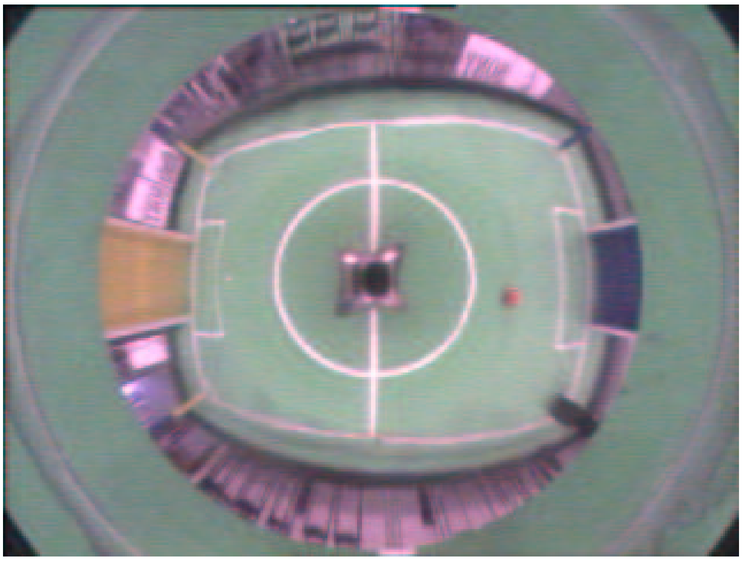

Fig. 4. The Image taken by the mirror implemented in 2003.

this excessively bad performance to be due to the differences of lighting condition across the field.

4 The self–localization module

In past work [8], we have studied and developed a self-localization method, which is independent from the specific sensor used for feeding data to it. Our localization system has three main components: the map of the environment, the perceptions and the localization engine. The environment is represented by a 2D geometrical map that can be inserted in the system through a configuration file or can be built by an automatic system, as in [7]. The map is made up of simple geometrical primitives, like points, lines, circles, etc., that we call *types*. For each primitive a list of attributes must be specified, which describe its main characteristics. Moreover, for each map element we have to specify the sensors able to perceive it.

On the other hand, we have sensor data and, possibly, other localization applications. Both produce perceptions, intended to be everything that provides information (even partial) about the the robot pose. Each perception is characterized by type, attributes, and the sensor that has perceived it. These data are useful in order to reduce the number of comparisons between perceptions and map elements. Moreover, for each perception we have both an estimate of the noise as well as the relative weight of the perception, which will be used during the voting process.

The localization engine takes in input the map of the environment as well as the perceptions, and outputs the estimated pose(s) of the robot (if the environment and/or the perceptions have inherent ambiguities).

We have implemented an evidence accumulation method, where the main point is to accumulate the evidence while working at high resolution in order to get an accurate estimate. We divide the search space in subregions (called *cells*), to which we associate a counter, as usual in evidence accumulation methods. Being ours a localization problem for a mobile robot in 2D, the search-space is the 3D manifold $\mathbf{R}^2 \times \mathbf{SO}(2)$.

Each perception increases the counter associated to a cell if it exists some point internal to the cell, which is compatible with both the perception and the model. Then, on the basis of the votes collected by each cell, the system selects the ones which are more likely to contain the correct robot pose. This process is further iterated on the selected cells until at least one termination condition is matched. When this happens, the system has to provide its currently best solution(s). In order to accomplish this task we cluster the adjacent cells in the vote accumulator as long as voted cells are encountered. Then we compute the barycenter of the cluster as the weighted average of the cell centers. The weights are the votes received by each cell. This is the output of the system. It is supplied with an associated estimate of its accuracy. Moreover, the solution is output with an associated reliability, which is correlated to the received votes.

We have successfully validated this localization approach in the RoboCup middle-size league. The map of the environment considers the presence of the lines, the two doors and the four flag posts. The vision sensor produces six types of perceptions: white points, blue directions, yellow directions, generic corner directions, corner yellow-blue-yellow directions and corner blue-yellow-blue directions. We decided to use only the direction of the goals and of the flag posts, since the distances measured for these objects are affected by large errors.

We have placed our robot inside the field at known positions and then we have run the localization procedure. The required accuracy was set to $100mm$ and 1^o; in this way the system stops the iterations when the cell size is smaller than the setting. The evaluation of the errors are summarized in the following: average position error = $5.63cm$, maximum position error = $9.32cm$, average orientation error = 0.72^o, maximum orientation error = 1.41^o.

5 Conclusion

In this paper, we have presented the three main areas on which research activity have been done in Milan Robocup Team in 2003. We have updated our control architecture which is now able to manage multi-level behaviors, and to face communication failure. We have implemented new mirrors for our panoramic sensors to study the distribution of resolution in the omnidirectional image, and defined a mirror design approach supported by software tools. Finally, we have implemented a robust and effective self-localization system, able to fuse information coming from many different sensors, and providing a good precision in a relatively short time, and in a large set of situations. Among the other many minor results we have achieved, we have here only mentioned the development of our second omni-directional robot, able to change shape to face game situations.

References

1. A. Bonarini, P. Aliverti, and M. Lucioni. An omnidirectional vision sensor for fast tracking for mobile robots. *IEEE Transactions on Instrumentation and Measuring*, 49(3):509–512, 2000.
2. A. Bonarini, G. Invernizzi, T.H. Labella, and M. Matteucci. An architecture to coordinate fuzzy behaviors to control an autonomous robot. *Fuzzy Sets and Systems*, 134(1):101–115, 2003.
3. A. Bonarini, M. Matteucci, and M. Restelli. Anchoring: do we need new solutions to an old problem or do we have old solutions for a new problem? In *Proceedings of the AAAI Fall Symposium on Anchoring Symbols to Sensor Data in Single and Multiple Robot Systems*, pages 79–86, Menlo Park, CA, 2001. AAAI Press.
4. A. Bonarini and M. Restelli. An architecture to implement agents co-operating in dynamic environments. In *Proceedings of AAMAS 2002 - Autonomous Agents and Multi-Agent Systems*, pages 1143–1144, New York, NY, 2002. ACM Press.
5. G. J. Klir, B. Yuan, and U. St. Clair. *Fuzzy set theory: foundations and applicatons*. Prentice-Hall, Upper Saddle River, NY, 1997.
6. P. Lima, A. Bonarini, C. Machado, F. M. Marchese, C. Marques, F. Ribeiro, and D. G. Sorrenti. Omnidirectional catadioptric vision for soccer robots. *International Journal of Robotics and Autonomous Systems*, 36(2-3):87–102, 2001.
7. F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, 4(4):333–349, 1997.
8. M. Restelli, D. G. Sorrenti, and F. M. Marchese. Murea: a multi-resolution evidence accumulation method for robot localization in known environments. In *Proceedings of IROS 2002*, pages 415–420, Piscayaway, NJ, 2002. IEEE Computer press.
9. Mark Simon, Sven Behnke, and Ral Rojas. Robust real time color tracking. In M.Veloso P.Stone, editor, *RoboCup 2000: Robot Soccer World Cup IV*, pages 239–248, Berlin, D, 2000. Springer Verlag.