Team Chaos 2004

K. LeBlanc[1], S. Johansson[2], J. Malec[3], H. Martínez[4], and A. Saffiotti[1]

[1] Center for Applied Autonomous Sensor Systems
Örebro University, S-70182 Örebro, Sweden

[2] Department of Software Eng. and Computer Science
Blekinge Institute of Technology, S-37225 Ronneby, Sweden

[3] Department of Computing Science
Lund University, S-22100 Lund, Sweden

[4] Department of Information and Communication Engineering
University of Murcia, E-30100 Murcia, Spain

Abstract. "Team Chaos" (formerly Team Sweden) is a multi-university team which has been competing in the 4-legged robot league of RoboCup since 1999. This paper shortly describes the Team Chaos entry for RoboCup 2004. The most distinctive points of our team are: (i) a general, principled architecture for autonomous systems, (ii) hierarchical fuzzy behaviors for fast incremental development of robust behaviors, (iii) fuzzy landmark-based localization for efficient, fault-tolerant self-localization. The main improvements for 2004 are: (i) improved vision-based perception, (ii) natural landmarks (corners on field) for self-localization, and (iii) improved cooperative perception and behavior.

1 Introduction

"Team Chaos" is a cooperative effort which involves universities in Sweden and abroad. In 2004, the sites of activity are: Örebro University (coordinating node), Lund University, the Blekinge Institute of Technology, and the University of Murcia in Spain. Team Chaos is a follow-up of Team Sweden, which was created in 1999 and has participated in the 4-legged league of RoboCup ever since. The distributed nature of the team has made the project organization demanding but has resulted in a rewarding scientific and human cooperation experience.

We had two main requirements in mind when we started to work on RoboCup. First, we wanted our entry should effectively address the challenges of uncertainty in this domain, where perception and execution are affected by errors and imprecision. Second, it should illustrate our research in autonomous robotics, by incorporating general techniques that can be reused on different robots and in different domains.

While the first requirement could have been met by writing *ad hoc* competition software, the second one led us to develop principled solutions that drew upon our current research in robotics, and pushed it further ahead.

Team Leader: Kevin LeBlanc (`kevin.leblanc@aass.oru.se`)
Web (includes publications): `http://www.aass.oru.se/Agora/RoboCup/`

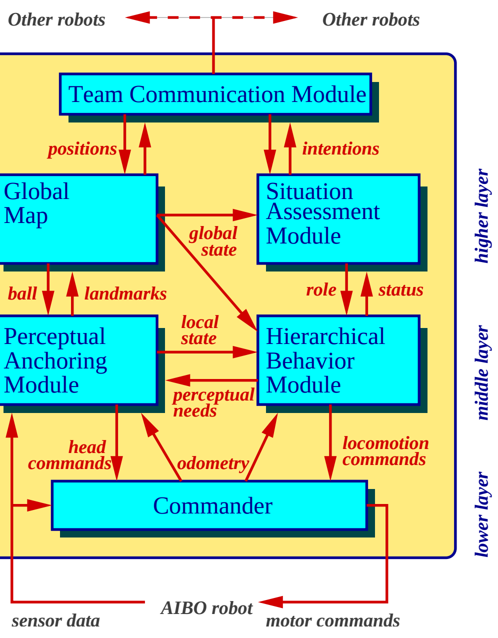

Fig. 1. The variant of the Thinking Cap architecture used by Team Chaos.

2 Architecture

Each robot uses the layered architecture shown in Fig. 1. This is a variant of the Thinking Cap architecture. [5] The lower layer (implemented in the CMD, or Commander) provides an abstract interface to the sensori-motoric functionalities of the robot. The middle layer maintains a consistent representation of the space around the robot (through the PAM, or Perceptual Anchoring Module), and implements a set of robust tactical behaviors, implemented in a hierarchical manner (in the HBM, or Hierarchical Behavior Module). The higher layer maintains a global map of the field (kept in the GM, or Global Map), and makes real-time strategic decisions based on the current situation (situation assessment and role selection is performed in the SAM, or Situation Assessment Module). Radio communication is used to exchange position and coordination information with other robots (via the TCM, or Team Communication Module).

[5] The Thinking Cap is an framework for building autonomous robots jointly developed by Örebro University and the University of Murcia. See `http://www.aass.oru.se/~asaffio/Software/TC/` `http://ants.dif.um.es/~humberto/robots/tc2/`.

3 Motion control

The Commander module accepts locomotion commands from the HBM in terms of linear and rotational velocities, and translates them to an appropriate walking style. This simplifies the writing of motion behaviors, which are easily portable between different (legged and/or wheeled) platforms. The Commander also implements several types of kicks. This module also controls head movement, and implements scan and look commands.

Our walking styles are based on the code for parametric walk created by the University of New South Wales (UNSW)[3], suitably re-tuned to best fit our needs. In addition, we have implemented a number of different kicks, as well as combined grab-and-kick and turn-and-kick motions.

4 Behaviors and behavior selection

The HBM implements a set of navigation and ball control behaviors realized using fuzzy logic techniques, and organized in a hierarchical manner [4, 6]. For instance, the following set of fuzzy rules implement the "GoToPosition" behavior.

```
IF (AND(NOT(PositionHere), PositionLeft))   TURN (LEFT);
IF (AND(NOT(PositionHere), PositionRight))  TURN (RIGHT);
IF (OR(PositionHere, PositionAhead))        TURN (AHEAD);
IF (AND(NOT(PositionHere), PositionAhead))  GO (FAST);
IF (OR(PositionHere, NOT(PositionAhead)))   GO (STAY);
```

More complex behaviors are achieved by combining simpler ones using fuzzy meta-rules which activate concurrent sub-behaviors. Behaviors also incorporate perceptual rules used to communicate the perceptual needs of active behaviors to the PAM.

5 Roles and role selection

Each robot can assume one of four different roles: *attack*, *defend*, *support* or *ball-control*. The first three roles are chosen by the SAM module depending on the current field situation; more than one robot can have the same role. Role selection is relatively consistent since the robots share the same information about the positions of objects in the field (this is explained in the section on information sharing). The fourth role, *ball-control*, is negotiated between the robots using wireless communication; at most one robot can be in *ball-control* mode at any time. There is also a pre-empting mechanism in place, to avoid having a robot which can not reach to ball prevent others from getting it; this also avoids situations where a robot who is obviously in a better position with respect to the ball is not able to take advantage of that position.

6 Perception

The locus of perception is the PAM, which acts as a short term memory of the location of the objects around the robot. At every moment, the PAM contains the best available estimates of the positions of these objects. Estimates are updated by a combination of three mechanisms: by *perceptual anchoring*, whenever the object is detected by vision; by *odometry*, whenever the robot moves; and by *global information*, whenever the robot re-localizes. Global information can incorporate information received from other robots (e.g. the ball position).

The PAM also takes care of selective gaze control, by moving the camera according to the current perceptual needs, which are communicated by the HBM in the form of a degree of importance attached to each object in the environment. The PAM uses these degrees to guarantee that all currently needed objects are perceptually anchored as often as possible (see [5] for details).

Object recognition in the PAM relies on three techniques: *color segmentation* based on a fast region-growing algorithm; *model-based region fusion* to combine color blobs into features; and *knowledge-based filters* to eliminate false positives. For instance, a yellow blob over a pink one are fused into a landmark; however, this landmark may be rejected if it is, for example, too high or too low relative to the field.

This year, significant improvements have been made to the PAM. Previously, seeds for the growing algorithm were obtained by hardware color segmentation on YUV images, taken directly from the camera. Seeds are now chosen by performing software thresholding on the images, which are first converted to HSV. This allows for a very portable and robust color segmentation, which works even in the face of changing lighting conditions.

In addition to the normal objects in the RoboCup domain, we also detect natural features, such as field lines and corners. Currently we use corners comprised of the (white) field lines on the (green) carpet. Corners provide useful information, since they can be classified by their type, and they are relatively easy to track (given the small field of view of the camera). Natural feature detection is performed using three techniques: *corner detection* based on changes in the direction of the brightness gradient; *color-based filtering* for filtering out corners that aren't on the carpet; and *corner grouping* for associating the detected corners with the natural features for which we are looking ([8]).

The first step in our algorithm is to segment the raw image using a simple thresholding algorithm. The result of this step on a sample image is shown in Fig. 2(b). The next step is to apply the Seed Region Growing (SRG) algorithm, which starts from a number of seed pixels (which are "safe" pixels of each color), and grows the region until a discontinuity in the color space is reached. The result of the growing step is shown in Fig. 2(c). Corners are detected using a gradient-based method, and a color filter is applied to determine which corners are on the carpet. Fig. 2(d) shows the corners obtained from the gradient-based method (in white), and those which survived the color filtering process (in black).

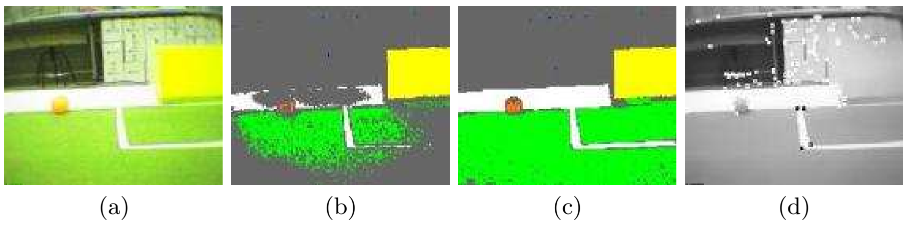

Fig. 2. (a) Raw image. (b) Segmented image (software thresholding). (c) Segmented image after region growing. (d) Gradient-based corner detection (white) and corners after color filtering (black).

7 Self-Localization

Self-localization in the 4-legged robot league is a challenging task: odometric information is extremely inaccurate; landmarks can only be observed sporadically since a single camera is needed for many tasks; and visual recognition is subject to unpredictable errors (e.g., mislabeling). To meet these challenges, we have developed a self-localization technique based on fuzzy logic, reported in [1]. This technique needs only qualitative motion and sensor models, can accommodate sporadic observations during normal motion, can recover from arbitrarily large errors, and has a low computational cost. The result of self-localization is used to make strategic decisions inside the SAM, and to exchange information between robots in field coordinates.

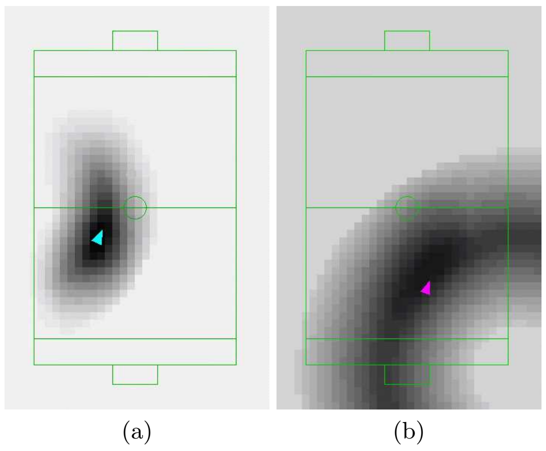

Fig. 3. Self localization grids.

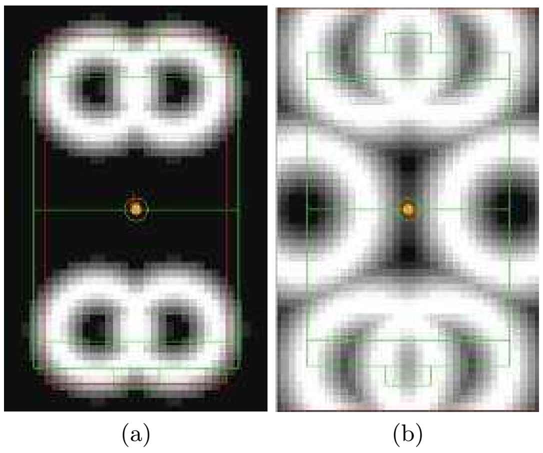

Fig. 4. Position grid induced by observation of two different types of corners. Due to symmetry of the field, the center of gravity is close to the middle of the field.

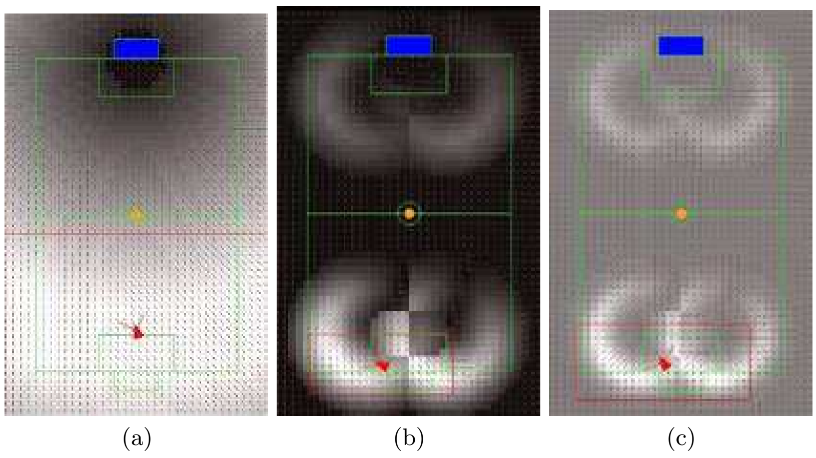

Fig. 5. Belief induced by the observation of (a) a net, (b) the first feature, and (c) the second feature. Initially the position of the robot is unknown (belief distributed along the full field).

This technique, implemented in the GM module, relies on the integration of approximate position information, derived from observations of landmarks and nets, into a fuzzy position grid — see Fig. 3. To include own motion information, we dilate the grid using a fuzzy mathematical morphology operator. Using this technique, our robots can maintain a position estimate with an average error of approximately $\pm 20\,cm$ and $\pm 10^{\circ}$ during normal game situations. Localization is done continuously during normal action. Stopping the robot to re-localize is only needed occasionally (e.g. in case of major errors due to an undetected collision).

The extension to our method which includes natural features as landmarks is described in [8]. The extended technique allows the robot to localize using only the nets and natural features of the environment (i.e. corners).

The observation of a landmark constrains the possible robot positions to be on a partial circle around the observed landmark (see Fig. 3 (b)). However, since our feature detector doesn't provide unique identifiers for corners, the possible robot positions induced by a corner observation is the union of several circles, each centered around a corner in the map. This can be seen in Fig. 4. It should be noted that the ability to handle this ambiguity efficiently is a primary advantage of our representation. The data association problem is automatically addressed in the fusion process (Fig. 5). Data association is a difficult problem, and many existing localization techniques are unable to adequately address it.

8 Information sharing

We use radio communication to exchange information between robots about the positions of objects in the field, in particular the ball. Information from other robots is fused using an original approach based on fuzzy logic, which is reported in [2]. In our approach we see each robot as an expert which provides unreliable information about the locations of objects in the environment. The information provided by different robots is combined using fuzzy logic techniques, in order to achieve agreement between the robots.

This contrasts with most current techniques, many of which average the information provided by different robots in some way, and can incur well-known problems when information is unreliable. Our method strives to obtain a consensus between data sources, whereas most other methods try to achieve a compromise or tradeoff between sources.

One of the major innovations of this approach is that it consistently maintains and propagates uncertainty about the robots own position into the estimates of object positions in the environment. Moreover, in constrast to many existing approaches, it does not require high accuracy in self-localization.

The schema used for cooperative ball localization is shown in Fig. 6. The self position grids for both robots are shown on the left, the ball location grids are shown in the middle, and the result of the fusion of the ball grids is shown on the right. Darker cells have higher degrees of possibility. The two triangles represent the robot's (defuzzified) estimates of their own positions. The yellow circles show the point estimates of the ball position.

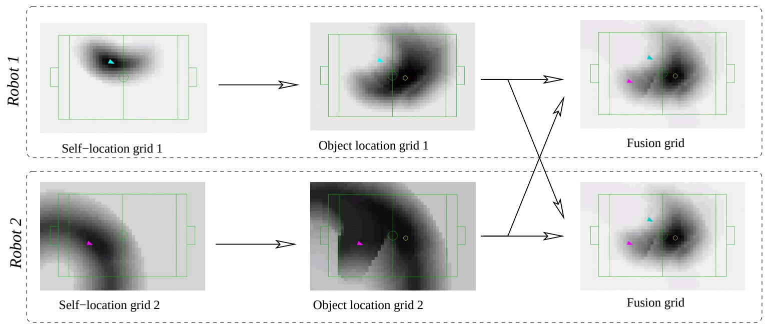

Fig. 6. Schema used for information fusion.

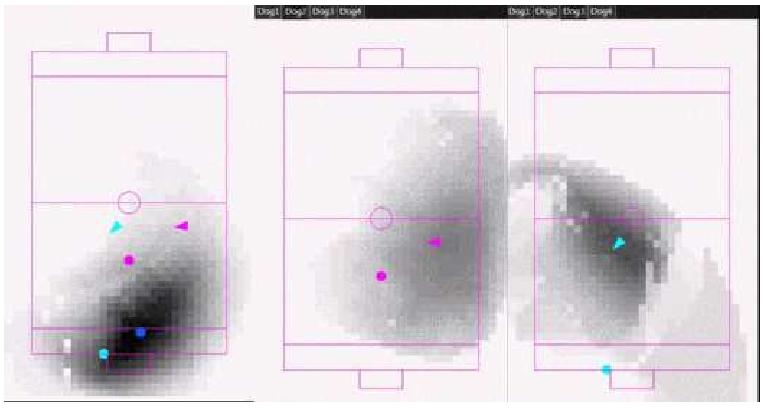

Fig. 7. Combining two imprecise estimates into an accurate one.

An example of our method can be seen in Fig. 7. The left window shows the ball grid resulting from the sharing of information. The three small circles near the bottom represent the point estimates of the ball position according to each robot (lighter circles) and as a result of the fusion (darker circle). The other two windows show the self-localization grids for robots 1 and 2, respectively. In this example, both robots happen to have a rather poor self-localization, which can be seen from the blurring of the two individual self-grids. Correspondingly, the individual estimates for the ball positions are relatively inaccurate, and quite different from each other. When intersecting the fuzzy sets, however, we obtain a fairly accurate fused estimate of the ball position (left window). Note that just averaging the estimates of the ball position produced by the two robots independently would result in an inaccurate estimate of the ball position.

9 Development Tools

Team Chaos is developing a suite of tools which can be used to tele-operate the robots, as well as debug, monitor, calibrate and tune the Team Chaos implementation. These tools allow online interaction with the various modules of the architecture. Communication between the robot and the host machine is implemented via a custom UDP protocol, which allows the user to dynamically modify various parameters of the system. This same protocol will also be used for information sharing, which is currently done through the TCP gateway. The configuration tool has proven to be very useful for calibrating the vision system, and for tuning various other sub-systems, for example the walking styles and kicks.

10 Conclusion

The general principles and techniques developed in our research have been successfully applied to the RoboCup domain. In particular, fuzzy logic was beneficial in writing robust behaviors, providing reliable self-localization, and achieving effective cooperative perception.

Acknowledgements

Team Chaos gratefully acknowledges the support of the Swedish KK foundation, the Spanish Seneca Foundation, “El Dulze” (Murcia, Spain), and the four participating universities.

References

1. P. Buschka, A. Saffiotti, and Z. Wasik. Fuzzy landmark-based localization for a legged robot. *IEEE/RSJ Int. Conf. on Intell. Robots and Systems* (IROS), 2000.
2. J.P. Canovas, K. LeBlanc, and A. Saffiotti. Robust multi-robot object localization using fuzzy logic. *Proc. of the Int. RoboCup Symposium*, Lisbon, PT, 2004. To appear.
3. B. Hengst, B. Ibbotson, P. Pham, and C. Sammut. Omnidirectional locomotion for quadruped robots. Birk, Coradeschi, Tadokoro (eds) *RoboCup 2001*, Springer-Verlag, 2002.
4. A. Saffiotti. Using fuzzy logic in autonomous robot navigation. *Soft Computing*, 1(4):180–197, 1997. Online at http://www.aass.oru.se/Agora/FLAR/.
5. A. Saffiotti and K. LeBlanc. Active perceptual anchoring of robot behavior in a dynamic environment. *IEEE Int. Conf. Robotics and Automation* (ICRA), 2000.
6. A. Saffiotti and Z. Wasik. Using hierarchical fuzzy behaviors in the RoboCup domain. In: Zhou, Maravall, Ruan (eds) *Autonomous Robotic Systems*, Springer-Verlag, 2002, pp. 235-262.
7. Z. Wasik and A. Saffiotti. Robust color segmentation for the RoboCup domain. *IEEE Int. Conf. on Pattern Recognition* (ICPR), 2002.
8. D. Herrero-Perez, H. Martinez-Barbera, and A. Saffiotti. Fuzzy Self-Localization using Natural Features in the Four-Legged League. *Proc. of the Int. RoboCup Symposium*, Lisbon, PT, 2004. To appear.