B-Human

Team Description for RoboCup 2012

Thomas Röfer[1], Tim Laue[1], Judith Müller[1],
Colin Graf[2], Arne Böckmann[2], Thomas Münder[2]

[1] Deutsches Forschungszentrum für Künstliche Intelligenz,
Cyber-Physical Systems, Enrique-Schmidt-Str. 5, 28359 Bremen, Germany
[2] Universität Bremen, Fachbereich 3 – Mathematik und Informatik,
Postfach 330 440, 28334 Bremen, Germany

1 Introduction

B-Human is a joint RoboCup team of the Universität Bremen and the German Research Center for Artificial Intelligence (DFKI). The team consists of numerous students as well as three researchers. The latter have already been active in a number of RoboCup teams such as the GermanTeam and the Bremen Byters (both Four-Legged League), B-Human and the BreDoBrothers (Humanoid Kid-Size League), and B-Smart (Small-Size League).

We entered the Standard Platform League already in 2008 as part of the BreDoBrothers, a joint team of the Universität Bremen and the Technische Universität Dortmund, providing the software framework, state estimation modules, and the get up and kick motions for the NAO. For RoboCup 2009, we discontinued our Humanoid Kid-Size League activities and shifted all resources to the SPL, starting as a single location team after the split-up of the BreDoBrothers. Since the split-up, the team B-Human has won every tournament it participated in. In 2009, 2010, and 2011, we won the RoboCup as well as the RoboCup German Open. This year, we already won the RoboCup German Open in Magdeburg, and we hope to be able to continue our success by also winning the RoboCup in Mexico City.

This team description paper is organized as follows: Section 2 presents all current team members, followed by short descriptions of our publications since the last RoboCup in Sect. 3. Our major development, the ability to play on a field without unique cues, is described in Sect. 4. This is accompanied by several developments regarding walking and proprioceptive sensing in Sect. 5 as well as multiple changes to our infrastructure in Sect. 6.

2 Team Members

B-Human currently consists of the following people who are shown in Fig. 1:

Fig. 1. The team B-Human at the RoboCup German Open 2012

Students. Andreas Stolpmann, Arne Böckmann, Danny Zitzmann, Malte Jonas Batram, Marcel Steinbeck, Michel Bartsch, Nico Lehmann, Robin Wieschendorf, Simon Taddiken, Thomas Münder.

Senior Students. Alexander Fabisch, Alexander Härtl (missing in the photo, because he took it), Colin Graf, Felix Wenk, Katharina Gillmann, Ole Jan Lars Riemann, Thijs Jeffry de Haas, Tobias Kastner.

Staff. Judith Müller, Thomas Röfer (team leader), Tim Laue.

3 Publications since RoboCup 2011

As in previous years, we released our code after RoboCup 2011 – together with a comprehensive documentation [1] – to the public on our website `http://www.b-human.de/en/publications/`. Up to date, we know of eight teams that based their RoboCup systems on one of our code releases (NimbRo SPL, BURST, NTU Robot PAL, AUTMan Nao Team, Austrian Kangaroos) or used at least parts of it (Cerberus, RoboEireann, MRL SPL).

Already before 2011, the B-Human software included sophisticated solutions for most relevant subtasks of playing robot soccer, such as vision, state estimation, and walking. Therefore, the development towards RoboCup 2011 did not focus on replacing specific low-quality components, but was guided by an overall goal: eliminating game delays by more efficient actions and faster reactions to game state changes. In [2], we summarize some of the developments that had the most impact regarding our goal: different ball models and corresponding co-

operative ball tracking and retrieval strategies, a path planner as well as new approaches for tackling situations.

While the quality of matches between the teams in the RoboCup Standard Platform League has increased a lot, there are still certain situations that prevent the game from progressing. One of the most severe ones is when a team loses track of the ball, because it cannot score goals or prevent the opponent team from scoring goals without knowing where the ball is. In [3], a method is presented to quickly find the ball again by searching the least-recently observed parts of the pitch. A consistent model shared by all robots of the team to identify these parts of the field is explained, as well as the procedure to coordinate the observation among the teammates, such that a varying number of robots can participate in the process.

When developing software for autonomous robots, the aspect of behavior engineering is, among tasks such as sensing, state estimation, and motion control, of major importance. Current solutions range from basic behavior-based approaches to sophisticated reasoning systems, in each case depending on the complexity of the robot's task as well as the available amount of computing time. In [4], the behavior specification language *b-script* is presented that allows describing hierarchical agent behaviors with imperative programming patterns using the concept of generators. However, B-Human currently uses another behavior specification language that is based on hierarchical state machines (see Sect. 6.1).

4 Playing with Two Yellow Goals

For the SPL Open Challenge 2011, we already presented a preliminary solution for playing on a field with two yellow goals, as described in [1]. For the 2012 competitions, we use the same approach but implemented several refined heuristics as described in the following subsections. The approach still does not require any additional detections of certain features in the field's surrounding.

4.1 Changes to Existing Modules

For self-localization, B-Human uses a combination of a particle filter and a Kalman filter. The former computes a global position estimate; the latter performs local tracking for refining the global estimate (cf. [1]). Both components needed to be modified for the new field layout. In a first step, the measurement models of both filters have been adapted by adding two more yellow goal posts and removing the blue posts accordingly. In addition, the particle filter's sensor resetting mechanism needed to be adapted in similar manner. These changes could already allow a proper position tracking in many situations.

Nevertheless, the particle filter's sensor resetting mechanism might generate samples that are at a totally wrong position due to the ambiguity of the goal posts that are used for computing the new samples. In the worst case, which occasionally occurs, these samples outstandingly match the perceptions made in

the following execution cycles. Consequently, a majority of all samples will be replaced by samples at the resetting position. Fortunately, a Standard Platform League field with uniform goal colors is twofold point symmetric with reference to the field's center. This means that even if the sensor resetting causes a wrong position, this position is not arbitrary but symmetric to the previous position. By assuming that no teleportations to symmetric positions occur in reality, we can deal with this kind of symmetry in a straightforward manner by computing a mirrored robot pose whenever the sample set represents the symmetric counterpart.

However, this approach is not failsafe as events such as persistent collisions with other robots close to the field center might confuse a robot to such an extent that it looses track of its orientation. Furthermore, a correct initialization is required when a robot starts to play. Solutions to these two problems are described in the following subsections.

4.2 The Own Side Model – A Reliable Base for Position Tracking

Whenever a robot starts to play, the field half, in which it is located, can be inferred from the rules:

- Before the game, after the halftime break, and after a timeout, the teams position their robots next to their own half for automatic placement. When the game state changes to *READY*, a robot must be in its own half.
- All robots must be in their own half when the game starts, i. e. the game state changes from *SET* to *PLAYING*.
- After a penalty, a robot is always placed in its own half.

In some of these situations, even more detailed position information is available, e. g. the goalkeeper is inside in penalty area when the game state changes from *SET* to *PLAYING*, the possible positions for reentering the field after a penalty are specified in detail, and if the opponent team has kickoff, a minimum distance from the opponent half is guaranteed. By considering a robot's walked distance through integrating odometry offsets over time, it is possible to infer the correct field side over a longer time period. In many matches, for instance, the goalkeeper does not move far enough to have any opportunity to enter the opponent half and thereby always knows its current field half.

The information provided by this model allows the particle filter as well as the module that handles the recovery from kidnapping (cf. 4.3) to resolve the field's twofold point symmetry in a number of game situations. As this approach relies on the rules, we have to expect the referees to apply them correctly.

4.3 Computation of Side Confidence for Kidnapping Recovery

The previously described approach allows a correct pose estimation in situations following certain game events. However, in the course of play, the certainties derived from these heuristics vanish. As aforementioned, certain game events

might interfere the localization process and cause uncertainty about the correct position alternative. To recover from such situations, we introduced the concept of side confidence.

As long as the own side model is confident about the current field half, the side confidence remains high. Hence, the particle filter remains in its normal tracking mode. After having walked over a certain distance and thereby lost the absolute confidence about the own side, events such as collision with other robots (cf. 5.1) decrease the side confidence over time. Falling down close to the field center even causes a total loss of confidence.

Having reached a state of uncertainty, a robot needs to refer to its teammates to regain confidence as we do not use any additional features outside the field. Similar to our Open Challenge 2011 demonstration, the ball is used as an additional point of reference. Whenever a robot perceives the ball, this observation is set in relation to the fused ball model of its teammates as well as to the mirrored alternative. If a robot's perceptions are consistent with those of its teammates over time, the robot's confidence increases. In contrast, if the perceptions are consistent with the mirrored alternative, the self localization modules are requested to switch to the alternative position estimate. For these comparisons, only teammates that are confident about their own positions are considered.

In general, these mutual agreements about the current ball position sustain the team's self-localization throughout a whole match.

4.4 Adaption of Behaviors

Besides the aforementioned enhancements of the self-localization components, some adaptions of the robot behaviors had to be made, too. One example for the normal game behavior is the head control: As ball observations are used to overcome kidnappings, it is important for all robots to look at the ball regularly. If a robot has not seen the ball for some time and is not able to find it at the position communicated by its teammates, it will look at the mirrored ball position on the field.

A special situation, which can be considered to demonstrate the major weakness of our approach, is a robot that has a low side confidence and is alone on the field, i. e. without any information source for kidnapping recovery. If one of our robots is in such a situation, it will ask the referees for side information. This is realized by a special behavior that kicks the ball out of the field. Subsequently, according to the rules, the ball should be placed on the field 1m closer to our own goal. By perceiving the ball position again, the orientation can be inferred.

5 Sensing and Motion

5.1 Detecting Physical Contact with Other Robots

Detecting other players is is very important for playing soccer. While team mates can communicate their position on the field through the wireless, other robots

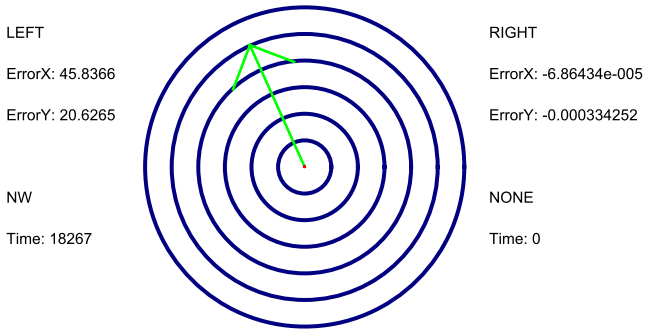

Fig. 2. Visualization of an arm collision.

have to be detected using sensors. Robots can be detected visually, with sonars, and physically. B-Human uses all of these methods. Visual detection works in longer ranges and uses the team markers [5], but it can also be used in short range to detect the opponent's feet in tackling situations [2]. Ultrasound only works in a range between 25 cm and 1.2 m, but the sensors are prone to be affected by crosstalks. The very short range is covered by detecting physical contacts to other robots using feet and arms.

Foot Contact. The foot contact detector utilizes the foot bumpers to detect whether the robot hit an obstacle while walking. Internally it calculates the *bumps per second (bps)* by maintaining a ring-buffer that stores bumps and non-bumps. The buffer represents one second (100 entries). A foot contact is detected whenever the bps rise above a certain threshold. This very simple approach allows for a very good foot contact detection without any false positives.

The behavior has been modified to react to the foot contacts, i. e. the robot walks a few steps backwards when it is not fighting for the ball.

Arm Contact. Arm contacts are detected by calculating the average difference between the requested and the actual shoulder joint angles over 100 frames (1second). If the average error rises above a certain threshold, an arm contact is detected. The direction is encoded using a simple compass. The compass values are deduced from the differences in forward/backward and sideways directions by looking at their signs (see Fig. 2).

Although it would be possible to calculate the exact push direction from the differences, the simple compass representation has been chosen because the behavior does not need finer-grained information.

5.2 Detecting Ground Contact

Detection whether the robot has contact is important for three reasons. First the gyroscopes of the NAO have to be continuously calibrated to compensate

for their bias drift. The assumptions made by the calibration procedure would be violated if the robot would be picked up and carried around. Therefore, without ground contact, no automatic inertia measurement sensor calibration is performed. Second, detecting the loss of ground contact is the only way the recognize that a robot was manually placed in the set state. Since manual placement means that the robot has to reestablish its position, it is advantageous to know that it will now take a longer distance to walk until it can enter the opponent half (see Sect. 4.2). Finally, it is convenient if a robot stops moving when it is lifted up by the referees.

Since we had a lot of trouble with failing force sensing resistors in the past, we decided not to use them to detect whether the robot is upright and has ground contact anymore. Instead, the robot can now be in one of two states: *contact* and *no contact*. It uses different conditions in each of these states to detect a state change. In the *no contact* state, the robots assumes that it is back on the ground when the orientation of the robot torso, computed from the acceleration sensor readings, is upright and when all sensor readings from the inertia measurement unit are nearly constant. In the *contact* state, ground contact loss is detected primarily through the estimate of the torso orientation that is computed to balance the walk and to compute the position of the camera relative to the field. The state changes when the torso orientation exceeds a certain threshold or when a sudden change of gravitational acceleration is measured with the z-axis accelerometer.

5.3 Improved Walk

Walking with the NAO is one of B-Human's ongoing research topics. We are using a computational inexpensive closed-loop method based on the 3D-LIPM which utilizes the inertia measurement unit and joint angle sensors of the NAO to react on perturbation and to correct the looseness of the model [6]. This year, the walk was tuned towards maneuverability and precision while maintaining the maximum walking speeds of previous years. In doing so, the time the robot takes to adjust its position to shoot the ball towards the goal could be reduced. Furthermore, the stability of the robot during the execution of in-walk kicks was improved by reworking the integration of kick motions into a walk cycle. That way, the balancing method used by the walk became slightly more effective while performing a kick.

6 Infrastructural Changes

6.1 Simplified Behavior Specification

In the past we used XABSL [7] to specify the behavior of the robots with a hierarchy of state machines. Since XABSL lacks support for complex data types and other high-level programming language features, we developed a new behavior specification language that is based on C++. Similar to XABSL, we

added the two keywords (*option* and *state*) to the C++ language to simplify the specification of state machines. The extended C++ code is translated back into ordinary C++ code using a specialized compiler. As a result, writing behavior specifications became more convenient. Unlike in XABSL, accessing data of other software modules does not require a symbol abstraction layer, which significantly reduces the coding overhead.

6.2 Logging

Since 2011 B-Human, uses a logging feature that is able to record a selection of the internal representations during a game. Based on those log files, it is possible to recreate a game in the simulator with very high accuracy. The files can also be synchronized to video footage from the game.

Internally, the B-Human system consists of the three processes Cognition, Motion, and Debug [1]. Until this year, the logging was tightly integrated into B-Human's debugging architecture, because the latter was already able to collect data from Cognition and Motion in the process Debug. However, sending representations between processes requires a certain amount of computation time. It turned out that it was not possible to log all relevant representations with the old approach while still running the Cognition process at maximum speed, i. e. at 30 Hz.

Therefore, the logger was rewritten and completely integrated into the Cognition process. It works by recording the state of each selected representation into a ring buffer. Since accessing the NAO's flash drive is very slow, the buffer is kept in memory until the game state changes to penalized or finished. Therefore, the buffer has to be large enough to record at least 10 minutes (i. e. 80MB at the RoboCup German Open 2012).

6.3 GameController in the Simulator

Since most of the behavior development is currently done in the simulator, it is important to be able to simulate entire 4 vs. 4 games (see Fig. 3). However, these only make sense if also the rules [8] are applied automatically. Since our code assumes that the robots are in legal kick-off positions when entering the playing state, it is important that this automatic refereeing also involves robot placement when necessary. The automatic referee currently supports the following functionality:

- When entering the *READY* state, all robots are unpenalized and placed at their return positions. The ready state ends after 45 seconds or if no robot has moved for two seconds.
- In the *SET* state, all robots that are in illegal positions are moved to the manual placement positions specified in the rules (and the mirrored ones). While manually placing the goal keepers is easy, finding suitable positions for the field players is more difficult, because the manually placed robots together with the robots that autonomously reached legal positions should

Fig. 3. A simulated game. The red and blue drawings on the field display the estimates of the current robot positions. The small dots above the robots show their current side confidence. Either a robot is certain to be in its own side (bright green, both goalies are certain), it is fairly sure about its playing direction (dark green), it is uncertain (yellow), or it is lost (red, not shown).

form an evenly distributed formation. The approach used for each robot is to first exclude all positions that would be closest to another robot of the same team. Among the positions that remain, the closest position is picked. This is repeated for each robot that has to be placed manually. There are no conflicts, because once a robot is placed manually, it is moved to its target position, so no other robot can be closer to that position afterwards. When robots are manually placed, they can recognize that they were lifted up. Besides moving robots, the ball is placed in the center of the field. The set state automatically ends after five seconds.

– In the *PLAYING* state, it is detected when the ball leaves the field. If it does so by crossing a goal line, the score is increased for the other team and the state switches to *READY* with the kick-off set for the team that received the goal. If the ball leaves the field somewhere else, it is determined which robot touched it last based on the underlying physics engine. The ball is then placed back according to the SPL rules. The game time is counted down in the *PLAYING* state. However, there is no automatic transition to the *FINISHED* state, because it should be possible to run longterm test games.

– It is also possible to manually penalize and unpenalize robots. They are put on the side when penalized and return at the correct positions when unpenalized. If several robots are penalized at same time, they are queued in the direction of their own goal – according to the rules.

7 Conclusions

The main focus of B-Human's work for 2012 was on playing soccer on a symmetric field with two yellow goals. Given that we only scored a single own-goal at the RoboCup German Open 2012, while kicking 55 times in the correct goal, one could argue that we were about 98% successful with our approach. By introducing the detection of foot contacts and improving the detection of arm contacts, we made our robots more perceptive to the presence of other robots in their immediate vicinity. Again, we improved the walk to be more stable and more precise in reaching target positions. Finally, some changes in the infrastructure allow for simpler behavior specification, better logging, and better behavior testing. This way, we hope to be prepared for the RoboCup 2012 in Mexico City.

References

1. Röfer, T., Laue, T., Müller, J., Fabisch, A., Feldpausch, F., Gillmann, K., Graf, C., de Haas, T.J., Härtl, A., Humann, A., Honsel, D., Kastner, P., Kastner, T., Könemann, C., Markowsky, B., Riemann, O.J.L., Wenk, F.: B-Human team report and code release 2011 (2011) Only available online: `http://www.b-human.de/downloads/bhuman11_coderelease.pdf`.
2. Laue, T., Röfer, T., Gillmann, K., Wenk, F., Graf, C., Kastner, T.: B-Human 2011 – eliminating game delays. In Röfer, T., Mayer, N., Savage, J., Saranlı, U., eds.: RoboCup 2011: Robot Soccer World Cup XV. Volume 7416 of Lecture Notes in Artificial Intelligence., Springer (2012) 25–36, to appear.
3. Wenk, F., Röfer, T.: Coordinated pitch observation for a humanoid robot soccer team. In: Proc. of the Fifth Workshop on Humanoid Soccer Robots in conjunction with the 2011 IEEE-RAS Int. Conf. on Humanoid Robots, Bled, Slowenia (2011)
4. de Haas, T.J., Laue, T., Röfer, T.: A scripting-based approach to robot behavior engineering using hierarchical generators. In: Proc. of the International Conference on Robotics and Automation (ICRA-2012), IEEE (2012) to appear.
5. Fabisch, A., Laue, T., Röfer, T.: Robot recognition and modeling in the robocup standard platform league. In Zhou, C., Pagello, E., Behnke, S., Menegatti, E., Röfer, T., Stone, P., eds.: Proceedings of the Fourth Workshop on Humanoid Soccer Robots in conjunction with the 2010 IEEE-RAS International Conference on Humanoid Robots. (2010)
6. Graf, C., Röfer, T.: A center of mass observing 3D-LIPM gait for the RoboCup Standard Platform League humanoid. In Röfer, T., Mayer, N., Savage, J., Saranlı, U., eds.: RoboCup 2011: Robot Soccer World Cup XV. Volume 7416 of Lecture Notes in Artificial Intelligence., Springer (2012) 101–112, to appear.
7. Loetzsch, M., Risler, M., Jüngel, M.: XABSL - A Pragmatic Approach to Behavior Engineering . In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2006), Beijing, China (2006) 5124–5129
8. RoboCup Technical Committee: RoboCup Standard Platform League (Nao) rule draft – 2012 rules, as of March 13, 2012 (2012) Only available online: `http://www.tzi.de/spl/pub/Website/Downloads/SPL_RuleDraft2012.pdf`.