B-Human

Team Description for RoboCup 2016

Thomas Röfer[1,2], Tim Laue[2], Jesse Richter-Klug[2], Felix Thielke[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 University of Bremen and the German Research Center for Artificial Intelligence (DFKI). The team was founded in 2006 as a team in the Humanoid League, but switched to participating in the Standard Platform League in 2009. Since then, we participated in seven RoboCup German Open competitions, the RoboCup European Open, and in seven RoboCups. We always won the German/European Open and became world champion four times.

This team description paper is organized as follows: In Sect. 2, our new image preprocessing is described. Section 3 presents our approach for detecting the black and white ball. In Sect. 4, a the new class of measurements is introduced for self-localization. Finally, Sect. 5 summarizes this paper.

1.1 Team Members

B-Human consists of the following people, most of whom are shown in Fig. 1:

Team Leaders / Staff: Tim Laue, Thomas Röfer.
Students: Yannick Bülter, Mathis Engelbart, Miguel Kasparick, Mohamadreza Amir Khostevan, Daniel Krause, Jonas Kuball, Lam Duy Le, Andre Lübken, Florian Maaß, Andre Mühlenbrock, Tim Müller, Lukas Post, Jesse Richter-Klug, René Schröder, Leonid Schwenke, Andreas Stolpmann, Alexander Stöwing, Kannan Thambiah, Felix Thielke, Alexis Tsogias.
Associated Researchers: Udo Frese, Judith Müller, Dennis Schüthe, Felix Wenk.

1.2 Publications Since RoboCup 2015

As in previous years, we released our code after the RoboCup 2015 together with a detailed description [8] on our website and on GitHub (`https://github.com/`

Fig. 1: The B-Human team after winning the RoboCup European Open 2016

`bhuman/BHumanCodeRelease`). Up to date, we know of 21 teams that based their RoboCup systems on one of our code releases (AUTMan Nao Team, Austrian Kangaroos, BURST, Camellia Dragons, Crude Scientists, Edinferno, JoiTech-SPL, NimbRo SPL, NTU Robot PAL, SPQR, UChile, Z-Knipsers) or used at least parts of it (Cerberus, MRL SPL, Nao Devils, Northern Bites, NUbots, RoboCanes, RoboEireann, TJArk, UT Austin Villa).

At the RoboCup Symposium 2016, we will present a new kick engine [1] that is based on a a modified version of Dynamic Movement Primitives (DMPs) [4] to describe the kick trajectory. This formulation allows the robot to dynamically adapt to changing ball positions and to change the speed of a kick in order to play passes of different lengths. Especially the latter is a capability that the kick engine we currently use [6] does not have. The paper also describes a motor model for the NAO robot as well as a ZMP-based balancer. The implementation has already been used during the Corner Kick Challenge at RoboCup 2015, enabling us to play precise passes to the striker robot that scored multiple goals. A short description is given in our last code release document [8].

One team member has contributed to a paper that summarizes the Drop-In Competition and analyzes the different team strategies as well as the impact of the different scoring metrics [3].

2 Image Preprocessing

In previous years, our entire vision subsystem worked directly with the YUV422 images supplied by the cameras of the NAOs, viewing them as YUV444 images by accessing only every second Y value and skipping every odd numbered row. Most vision modules did not use the color values directly but looked up a color class for each pixel in a precomputed color lookup table which was calibrated beforehand using the color space HSI.

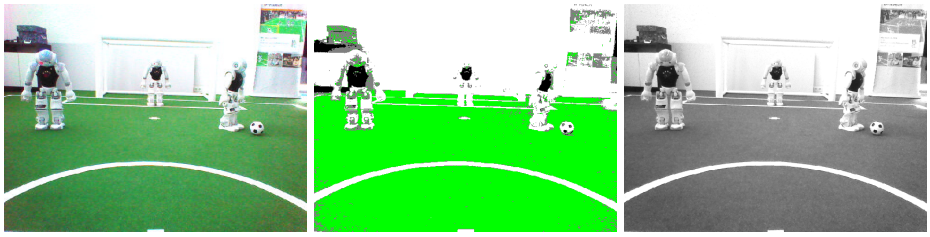


Fig. 2: The camera image (left) compared to the color-classified and gray-scaled images which now form the basis of all further image processing

Considering that at the RoboCup 2016, we will participate in the outdoor competition, in which games will be played under natural lighting [2], we chose a new approach to automatically detect the ranges of color values for the color classes to cope with different lighting conditions during a game. However, recalculating the precomputed color table for each frame would be computationally too expensive, so we chose to remove the color table altogether and instead convert each camera image to an image containing only the color classes of each pixel based on color value ranges determined in the color space YHS, which is the Y channel of the original image, the hue determined by the direction of the vector described by the U and V channels, and the saturation determined by the length of that vector.

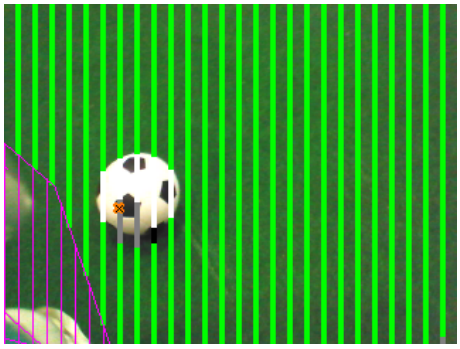
The color value ranges for classifying colors, which for now are only white, black, green, and none, may be either detected at runtime for each image or calibrated beforehand with similar results as the old approach.

While creating the color classified image, a gray-scaled image, which contains only the Y channel of the camera image, is created simultaneously. That way, vision modules can now choose to work on a gray-scaled image, a color classified image, or a combination of both. The new images compared to the camera image can be seen in Fig. 2.

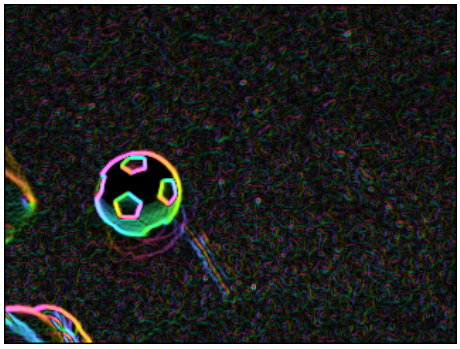
As our entire vision subsystem now only uses these two computed images, we chose to calculate them from full-resolution YUV422 camera images and reduce the resolution of the camera images from 1280×960 pixel for the upper and 640×480 pixel for the lower camera to 640×480 pixel for the upper and 320×240 pixel for the lower camera respectively.

3 Ball Detection

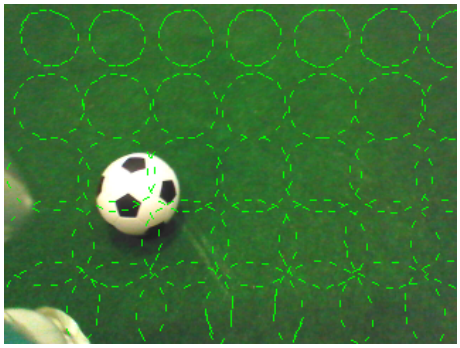
The introduction of the black and white ball is the major new challenge in the Standard Platform League in 2016. Until the RoboCup 2015, the ball was orange and rather easy to detect. In particular, it was the only orange object on the field. The new ball is mainly white with a regular pattern of black patches, just as a miniature version of a regular soccer ball. The main problem is that the field lines, the goals, and the NAO robots are also white. The latter even have



(a) Vertical scan lines and a detected ball candidate (the cross). Parts of the robot's body are ignored (bottom left).



(b) Contrast-normalized Sobel image. The colors indicate the directions of the gradients.



(c) Visualization of the search space for the ball contour. The actual search is only performed around the ball candidate, but in single pixel steps in both dimensions.

(d) The contour with the highest response (green) and the sample grid to check the ball pattern (pixels classified as black are shown in red, white pixels in blue).

Fig. 3: The main steps of the ball detection

several round plastic parts and they also contain grey parts. Since the ball is often in the vicinity of the NAOs during a game, it is quite challenging to avoid a large number of false positives.

We use a multi-step approach for the detection of the ball. First, the vertical scan lines our vision system is mainly based on are searched for ball candidates. Then, a contour detector fits ball contours around the candidates' locations. Afterwards, fitted ball candidates are filtered using some general heuristics. Finally, the surface pattern inside each remaining candidate is checked.

Searching for Ball Candidates. Our vision system scans the image vertically using scan lines of different density based on the size that objects, in particular the ball, would have in a certain position of the image. To determine ball candi-

dates, these scan lines are searched for sufficiently large gaps in the green that also have a sufficiently large horizontal extension and contain enough white (cf. Fig. 3a).

Fitting Ball Contours. As the position of a ball candidate is not necessarily in the center of an actual ball, the area around such a position is searched for the contour of the ball as it would appear in this part of the image given the intrinsic parameters of the camera and its pose relative to the field plane. The approach is very similar to the detection of objects in 3-D space using a stereo camera system as described by Müller et al. [5], but we only use a single image instead. For each ball candidate, a contrast-normalized Sobel image of the surrounding area is computed (cf. Fig. 3b). This contrast image is then searched for the best match with the expected ball contour (cf. Fig. 3c). The best match is then refined by adapting its hypothetical 3-D coordinates (cf. Fig. 3d).

Filtering Ball Candidates. The fitting process results in a measure, the *response*, for how well the image matches with the contour excepted at the candidate's location. If this value is too small, the ball candidate is dropped. All candidates that fit well enough are processed in descending order of their response. As a result, the candidate with the highest response that also passes all other checks will be accepted. These other checks include that the ball radius found must be similar to the radius that would be expected at that position of the image. Our vision system also detects other robots on the field. If a ball candidate is inside the area of a detected robot and could not just be lying in front of it, it is also dropped. However, if a ball candidate is completely surrounded by green pixels and the response was high enough to exclude the possibility of being a penalty mark, the ball candidate is accepted right away, skipping the final step described below that might be failing if the ball is rolling quickly.

Checking the Surface Pattern. For checking the black and white surface pattern, a fixed set of 3-D points on the surface of the ball candidate are projected into the image (cf. Fig. 3d). For each of these pixels, the brightness of the image at its location is determined. Since the ball usually shows a strong gradient in the image from its bright top to a much darker bottom half, the pixels are artificially brightened depending on their position inside the ball. Then, Otsu's method [7] is used to determine the optimal threshold between the black and the white parts of the ball for the pixels sampled. If the average brightnesses of both classes are sufficiently different, all pixels sampled are classified as being either black or white. Then, this pattern is looked up in a pre-computed table to determine whether it is a valid combination for the official ball. The table was computed from a 2-D texture of the ball surface considering all possible rotations of the ball around all three axes and some variations close the transitions between the black and the white parts of the ball.

The approach allows our robots to detect the ball in distances of up to five meters with very few false positive detections.

4 Complex Field Features and Self-Localization

In the past, B-Human used goals as a dominant feature for self-localization. When the field was smaller and the goal posts were painted yellow, they were easy to perceive from most positions and provided precise and valuable measurements for the pose estimation process. In particular the sensor resetting part, i. e. the creation of alternative pose estimates in case of a delocalization, was almost completely based on the goal posts perceived. In 2015, we still relied on this approach, using a detector for the white goals [9]. However, as it turned out that this detector required too much computation time and did not work reliably in some environments (requiring lots of calibration efforts), we decided to perform self-localization without goals but by using complex field features derived from certain constellations of perceived field lines.

Field Features. The self-localization always used field lines, their crossings, and the center circle as measurements. Since 2015, these features are complemented by the perception of the penalty marks. All these field elements are distributed over the whole field and can be detected very reliably, provided a constant input of measurements in most situations.

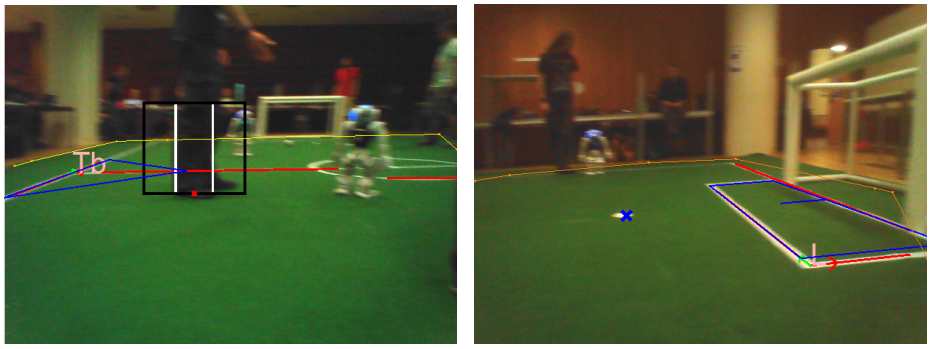
Built upon this, the perception of a new category of measurements, the so-called *Field Features*, has been implemented. They are created by combining multiple basic field elements in a way that a robot pose (in global field coordinates) can be derived directly. The handling of the field symmetry, which leads to actually two poses, is described in the following section.

The features that a currently computed are: the penalty area, the center circle (including the center line that provides the direction), the field corners, the center corners (where the center line touches the outer field lines, cf. Fig. 4a), and the goal frame on the floor. Some of these features can be determined by different field element constellations, e. g. the penalty area can be derived from a subset of its corners as well as from a penalty mark and the penalty area's line next to it (cf. Fig. 4b).

All considered lines are preprocessed by classifying them in short and long lines and by determining their relation to the field border (if available). The crossings of the lines are categorized as L / T / X on the one hand and in *big* / *small* on the other hand. In this context, *big* means that it is a crossing that results from the intersection of two long lines, such as field corners perceived from a longer distance.

Overall, this approach provides a much higher number of reliable pose estimates than the previous goal-based approach, as the field lines on which it is based can be seen from many perspectives and have a more robust context (straight white lines surrounded by green carpet) than the noisy unknown background of goal posts.

Localization Resetting. The self-localization is based on a particle filter with a low number of particles that each include an Unscented Kalman filter. Field



(a) Center corner: two long field lines intersect and represent a big T (marked *Tb*).

(b) Penalty area: a penalty mark and a close line allow the detection of this area.

Fig. 4: Two examples for field features, both are depicted as blue lines.

features can be used as measurements but not as a perception of relative landmarks, instead, an artificial measurement of a global pose is generated, reducing the translational error in both dimensions as well as the rotational error at once. Furthermore, no data association – in contrast to the basic field elements that are not unique – is required.

However, particles only cover the state space very sparsely. Therefore, to recover from a delocalization, it is a common approach to perform *sensor resetting*, i. e. to insert new particles based on recent measurements. The field features provide exactly this information and thus are used by us for creating new particles.

As false positives can be among the field features, e. g. caused by robot parts overlapping parts of lines and thereby inducing a wrong constellation of elements, an additional filtering step is necessary. All robot poses that can be derived from recently observed field features are clustered and only the largest cluster, which also needs to contain a minimum number of elements, is considered as a candidate for a new sample. This candidate is only inserted into the sample set in case it significantly differs from the current robot pose estimation.

To resolve the field's symmetry when handling the field features, we use the constraints given by the rules (e. g. all robots are in their own half when the game state switches to *Playing* or when they return from a penalty) as well as the assumption that the alternative that is more compatible to the previous robot pose is more likely than the other one. This assumption can be made, as no teleportation happens in real games. Instead, most localization errors result from situations in which robots lose track of their position and accumulate translational and rotational errors.

These changes in localization and perception – along with a growing number of robots that have a z-axis gyroscope – enabled us to reduce the number of *Leaving the Field* penalties from 15 (in seven games during RoboCup 2015) to 0 (in five games during the RoboCup European Open).

5 Summary

For RoboCup 2016, we have made several major changes regarding image preprocessing and perception. The aim of these changes is to improve the overall robustness of the system, particularly with regard to the ability to play in environments with unstable lighting conditions, such as the upcoming Outdoor Competition.

Some of these new features as well as the new ball detection, which is a must-have to be able to play at all, have already been used at the RoboCup European Open 2016 and contributed to our success in that competition. Thus, we are looking forward to a successful participation in the RoboCup 2016 in Leipzig!

References

1. Böckmann, A., Laue, T.: Kick motions for the NAO robot using dynamic movement primitives. In: RoboCup 2016: Robot Soccer World Cup XX. Lecture Notes in Artificial Intelligence, Springer (2017), to appear
2. Committee, R.T.: RoboCup Standard Platform League (NAO) rule book (2016), only available online: http://www.informatik.uni-bremen.de/spl/pub/Website/Downloads/Rules2016.pdf
3. Genter, K., Laue, T., Stone, P.: Benchmarking robot cooperation without pre-coordination in the RoboCup standard platform league drop-in player competition. In: Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 3415 – 3420 (2015)
4. Ijspeert, A.J., Nakanishi, J., Hoffmann, H., Pastor, P., Schaal, S.: Dynamical movement primitives: learning attractor models for motor behaviors. Neural computation 25(2), 328–373 (2013)
5. Müller, J., Frese, U., Röfer, T.: Grab a mug - object detection and grasp motion planning with the nao robot. In: Proceedings of the IEEE-RAS International Conference on Humanoid Robots (HUMANOIDS 2012), Osaka, Japan. pp. 349–356. IEEE (2012), http://www.informatik.uni-bremen.de/agebv2/downloads/videos/muellerhumanoids12.mp4
6. Müller, J., Laue, T., Röfer, T.: Kicking a Ball – Modeling Complex Dynamic Motions for Humanoid Robots. In: del Solar, J.R., Chown, E., Ploeger, P.G. (eds.) RoboCup 2010: Robot Soccer World Cup XIV. Lecture Notes in Artificial Intelligence, vol. 6556, pp. 109–120. Springer (2011)
7. Otsu, N.: A Threshold Selection Method from Gray-level Histograms. IEEE Transactions on Systems, Man and Cybernetics 9(1), 62–66 (1979), http://dx.doi.org/10.1109/TSMC.1979.4310076
8. Röfer, T., Laue, T., Richter-Klug, J., Schünemann, M., Stiensmeier, J., Stolpmann, A., Stöwing, A., Thielke, F.: B-Human team report and code release 2015 (2015), only available online: http://www.b-human.de/downloads/publications/2015/CodeRelease2015.pdf
9. Röfer, T., Laue, T., Richter-Klug, J., Stiensmeier, J., Schünemann, M., Stolpmann, A., Stöwing, A., Thielke, F.: B-Human team description for RoboCup 2015. In: RoboCup 2015: Robot Soccer World Cup XIX Preproceedings. RoboCup Federation, Hefei, China (2015)