B-Human 2021 – Playing Soccer Out of the Box

Thomas Röfer[1,2], Tim Laue[2], Arne Hasselbring[1], Lukas Malte Monnerjahn[2], Nele Matschull[2], Lukas Plecher[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
{thomas.roefer,arne.hasselbring}@dfki.de
{tlaue,lumo,ne_ma,plecher}@uni-bremen.de

Abstract. In 2021, our team B-Human participated in two events in the RoboCup Standard Platform League, namely the German Open Replacement Event and the actual RoboCup. We won both competitions. In both events, the biggest scientific challenge was to be able to play soccer on foreign robots in remote locations with the human team members neither being on site nor having direct access to these robots. We present our approach to a robot soccer system that only requires a fully automatic pre-game extrinsic camera calibration, but otherwise works out of the box. This paper focuses on the automatic calibration and some aspects of our lighting-independent computer vision system.

1 Introduction

RoboCup is a competition. Thus, teams tune their systems for maximum performance regarding speed and precision. Although overall success depends on sophisticated algorithms, the actual performance often hinges on a multitude of suitable parameters. For this reason, during a RoboCup event, one can constantly observe participants on the fields, adapting their software for particular robots and the current environment. In general, the following robot software aspects are affected the most:

Vision System: Many efficient robot vision approaches strongly depend on the current lighting conditions and require a proper preset. However, when playing next to huge windows, the interplay of sun and clouds might change these conditions quite quickly and – combined with a fixed vision parameter preset – cause effects such as overexposure, cast shadows, and changes of the color saturation.

Camera Parameters: For a precise projection of objects detected in an image to robot-relative positions on the field plane, a calibration of each individual robot's cameras is necessary. In general, the production tolerance is so significant that without any calibration, reliable self-localization and object tracking are not possible.

Joints and Motions: Different robots are in different states of wear. Thus, two robots of the same model often cannot carry out the same motion pattern, such as walking or kicking, in the same way. Furthermore, the structure and state of the field surface is not the same on every field and might strongly affect the reliability of any motion.

In 2021, no regular RoboCup competitions were held at any central place. Instead, most leagues decided to hold virtual and decentralized alternative competitions [8,10]. In the Standard Platform League, two competitions were held: the *German Open Replacement Event (GORE)* as well as the *RoboCup 2021 Virtual*. As all teams use the same robot platform, which is currently the NAO by SoftBank Robotics, for both events a unique setup was possible: Full games as well as different technical challenges were held on real robots but inside labs and on robots that were not directly accessible by the teams playing. Software and (sometimes) robots traveled, but people did not.

Such kind of remote setup poses significant challenges to the aforementioned tuning aspects as time and possibilities for interaction are extremely limited. In this paper, we describe our methods for autonomous calibration and lighting independent vision that we applied successfully at RoboCup 2021 as well as at the German Open Replacement Event. The third aspect – robust motion – has been addressed in a separate paper [5].

The remainder of this paper is organized as follows: Section 2 describes the competitions we participated in, along with their particular challenges. Afterwards, our approach for autonomous calibration is presented in Section 3. Our developments that allow calibration-free robot vision are described in Section 4 and Section 5 respectively. Finally, our results in 2021 are summarized in Section 6.

2 Challenges of the 2021 Competitions

As mentioned before, in 2021, the Standard Platform League held two competitions that required teams to run their software on robots that were not their own and to let them play on pitches to which the teams did not have direct access to.

As the annual RoboCup German Open was canceled in 2021, the Standard Platform League community organized its own competition, which was called *German Open Replacement Event (GORE)*. The event was held in May 2021 at two places – Technical University of Dortmund and University of Bremen – in parallel. The participating teams sent some of their robots to one of the places to form two robot pools that consisted of 16 robots each. No participants traveled with their robots. To hold full 5 vs 5 games, two hours before each game, every playing team was assigned five random robots from a pool. These could have been some of the own robots, but not necessarily so. Within these two hour time slots, a team had to deploy its software, configure each robot, and perform necessary calibrations and tests. This process had to be fully remote, the actual

handling of the robot hardware was executed by local assistants. Given such a short period of time and the large number of robots, any automatisms as well as calibration-free implementations are highly beneficial. A detailed description of the GORE is given by Laue *et al.* [2]. The applied rules, which are mainly based on the standard game rules, featuring some extensions for hardware safety, are described in the GORE rule book [7].

RoboCup 2021 was a virtual event, too. The Standard Platform League competition consisted of four separate challenges, the results of which were combined to determine an overall winner. Two of these challenges were fully local: the Passing Challenge and the Obstacle Avoidance Challenge. Teams performed these challenges on their own robots in their own labs and streamed their attempts. As such a setting allows full control over the robots and their environment, automatic calibration as robust algorithms are less important and can be compensated by hand tuning details. However, the remaining two challenges, namely the Autonomous Calibration Challenge and the 1vs1 Challenge, were held remotely and posed challenges similar to those of the GORE. In both cases, the robot software was run on foreign robots at places that were not accessible by the participants. In the Autonomous Calibration Challenge, a robot calibration had to be performed fully autonomously. Subsequently, the robot had to navigate to multiple targets and detect two balls on the pitch, in all cases as precise and fast as possible, making the quality of the calibration a decisive factor. The 1vs1 Challenge was a minimalistic version of robot soccer, played with one robot per team. For this challenge, the calibration had to be done within 20 minutes and allowed fully automatic procedures as well as a semi-automatic execution with the help of a local volunteer. A detailed description of all rules is given in the official rule book [9].

3 Autonomous Calibration

One standard step in robot vision is the transformation of a detected object's image coordinates into the robot's world coordinate system. To perform this step, information about the robot's kinematics, the camera's position and orientation with respect to the robot body, as well as certain camera parameters, such as opening angles and the optical center, are needed. Even after careful industrial-scale production, the deviations from the original specifications are often high enough to cause huge transformation errors. Thus, in general, a camera calibration is unavoidable for robust and precise robot perception.

We model the deviation of the camera poses from their specified values by six parameters: a roll and pitch offset in the body and for each of the two cameras. Translational errors are too insignificant to the projection error and the yaw angles cannot be calibrated due to lack of a global reference.

An important feature of the calibration is that it works without exact knowledge of the robot's pose. The parameters are derived from the known relative angles (either parallel or orthogonal) and distances between lines on the field. Specifically, we use the view of the goal area from the side, which gives two

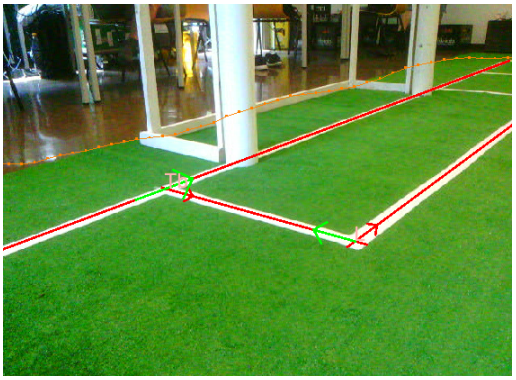


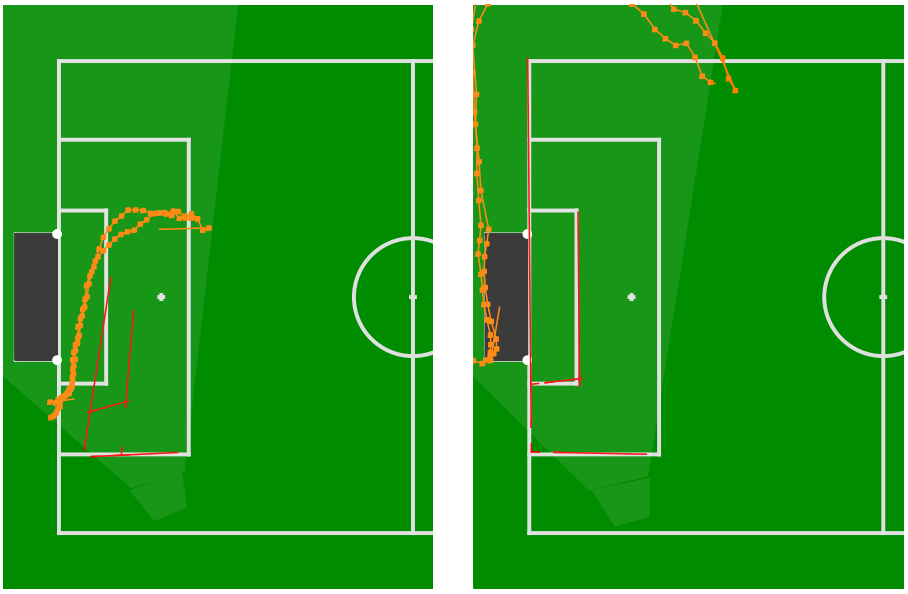
Fig. 1: The robot's view during the calibration, taken during our attempt in Hamburg.

right angles and one distance between two lines (see Fig. 1). The goal area is observed with three different head pan angles with each of the two cameras (this is necessary to distribute the error among the body and camera offsets), which gives 18 samples for 6 parameters in a least-squares optimization. To identify the goal area in the image without needing the yet uncalibrated projection to the field, we search for a short line that connects two long lines in the image, as this relation is true for the goal area only. For improved accuracy, the lines that are returned by our usual detection algorithm are refined by a Hough transform on a Sobel grayscale image before calculating the cost function.

In the autonomous mode required for RoboCup 2021, the robot is initially placed at one of two known positions at the side of the field. The self-localization is running using percepts that are projected using the yet uncalibrated camera pose, but no sensor resetting is performed in order to prevent the robot's pose estimate from jumping due to incorrectly perceived landmarks. The robot walks to a position from which the short line of the goal area can be observed well. It adjusts its head angles and its position if the necessary features are not detected within some time. After each successfully recorded sample, the robot also turns its head and body to observe the goal area from another angle. After all samples have been collected, the optimization process is started. We use a Gauss-Newton optimizer, but since the cost function is quite complex, the Jacobian is approximated by the numerical gradients. The optimization is done once the parameter change is sufficiently low for a number of optimization steps. Figure 2 shows the results of an automatic calibration performed at RoboCup 2021.

4 Preliminary Work on Deep Learning-based Robot Vision

Due to the increasing complexity of the Standard Platform League's environment – removing color-coded items and enforcing setups with as much natural light-



(a) Projected field lines before calibration (b) Projected field lines after calibration

Fig. 2: Results of the autonomous camera calibration on a robot of the HULKs team during RoboCup 2021. Inside the fields of view of the two cameras, the projected field lines are shown in red, the projected field border in orange.

ing as possible – we replaced most algorithmic robot vision solutions by Deep Learning based approaches for several years now, as they have been proven to provide the necessary robustness to cope with this level of complexity.

Since 2017, the ball detection is based on the classification of previously determined image patches. The most recent version is described in our 2019 winner paper [11]. It uses an encoder-decoder architecture to determine the pixels belonging to a ball and includes an additional position estimation network to infer the exact ball center within a given patch.

In 2019, we presented and started using JET-Net [4], which is able to perform real-time robot detection. Given a full image and no external preprocessing steps, it is able to determine robot bounding boxes along with an estimate of the distance to a detected robot.

One important step in our vision pipeline is the computation of the field border to exclude most parts outside the field, which are not specified and thus contain random things, from further image processing steps. In the past, the detection of the field border required the calibration of the field's green within the color space. Since this year, we do not require this calibration anymore, as points of the field border are computed by another neural network. Along with their position, an uncertainty of their position is inferred too. This allows

a more reliable matching of the field's edges. A full description of this approach is published by Hasselbring and Baude [1].

For fast inference of neural networks on our NAO robots, we use our own implementation, which is available as open source and is described in detail by Thielke and Hasselbring [14].

5 Lighting-independent Field Line Detection

As described in the previous section, multiple important elements of a robot soccer game can already be detected by neural networks. However, for the field lines as well as for the computation of image patches that are candidates for containing the ball, this is not possible yet. These objects can be quite small or thin in an image and – in case of lines – still span over the whole width of an image. This would require to enter a high resolution image into a neural network and to apply more flexible approaches such as semantic segmentation. As the current computing power of the NAO robot is probably not sufficient for such a task, we developed a lighting-independent algorithmic solution.

The field line detection relies on a grid of horizontal and vertical scan lines along which the image is segmented into the classes *green*, *white*, and a generic class for everything else. In the computation of this segmentation, many thresholding parameters are needed. In past years, these had to be calibrated by hand before the start of the game. Instead, we now calculate all needed parameters per image, allowing the perception to adapt to lighting changes during the game as well as reducing calibration overhead.

5.1 Color-Segmented Scan Lines

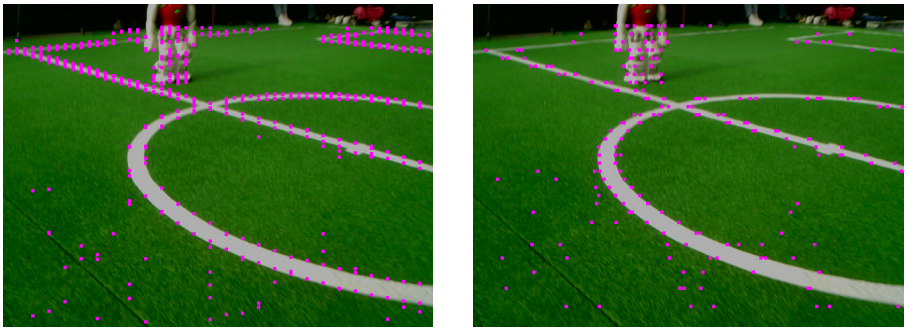
Scan lines are a means to analyze only a part of the image while retaining as much useful information as possible. This is especially important now that our color segmentation approach takes significantly more execution time. That makes it impossible to apply the color segmentation to the whole image, so instead we use it only on the scan lines.

A scan line is simply a vertical or horizontal line of pixels in the image, so each scan line position is signified by a single x- or y-coordinate in the image plane. An adaptive grid defines the scan line positions depending on the robot's view angle. We also cut off the scan lines at the field boundary that was computed beforehand.

The color segmentation of the scan lines is done in four steps:

1. Split each scan line into homogeneous regions
2. Classify regions as field
3. Classify regions as white
4. Perform cleanup operations

Currently, this is done for the vertical and horizontal scan lines independently of each other.



(a) Region transition points of vertical scan lines

(b) Region transition points of horizontal scan lines

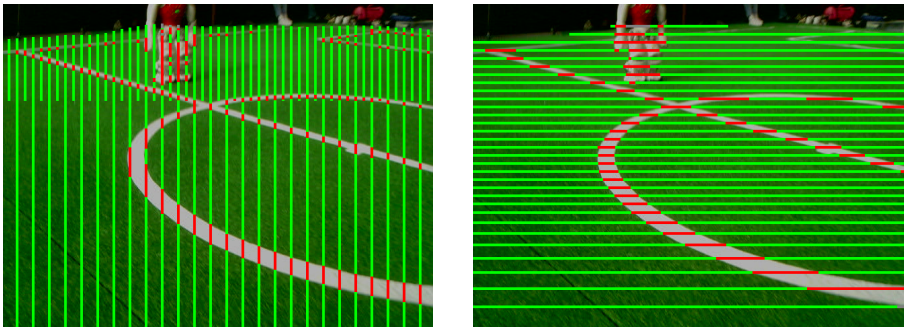
Fig. 3: The pink dots indicate the positions where two scan line regions are separated.

For the first step, we draw samples in regular intervals on each scan line. For each sample point, we apply a Gaussian blur filter on the luminance to smooth out noise. When two consecutive sample points' smoothed luminances differ significantly, a Sobel filter is applied to the interval in between. The highest Sobel value then signifies the exact position that separates two regions of the scan line. In Figure 3, the resulting region transition points are marked with pink dots. Each region gets a representative YHS2 (see [13, p. 60]) color value by averaging over its pixels.

The field often appears in the image in form of rather big homogeneous patches that are within a certain color range. In order to use this characteristic, we unite neighboring regions with similar YHS2 color values. This is done efficiently using a union-find disjointed tree data structure. All united regions that span enough pixels and are in the expected color range of the field become classified as field and allow to determine the exact color range the field has in this image. Afterwards, all remaining regions are also classified as field, if they are within the ascertained field color range.

The classification of white regions utilizes simple thresholding techniques. The thresholds depend on the previously established field color and the approximated average luminance and saturation of the image. The results can be seen in Figure 4.

Finally, neighboring regions on the same scan line, which have the same classification, become united into one region. We also noticed that this classification procedure sometimes leaves small gaps in between a field region and a white region. Because this impairs the line detection, we fill small gaps between a field and a white region by dividing them up equally into the neighboring regions, as can be seen in Figure 5.

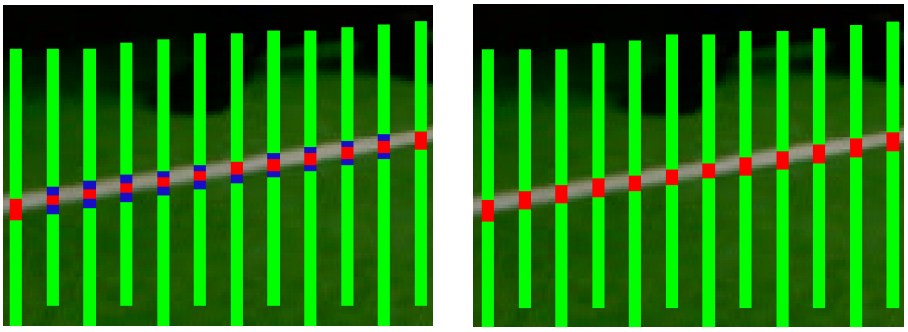


(a) Vertical color segmented scan lines (b) Horizontal color segmented scan lines

Fig. 4: The color segmented scan lines. For better contrast, the regions classified as white are drawn in red.

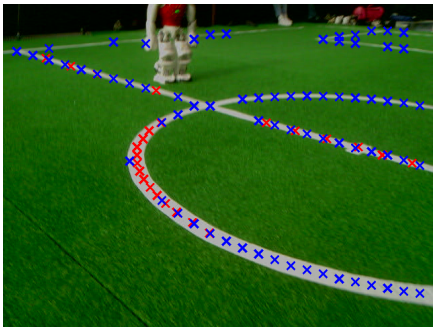
5.2 Line Detection

The perception of field lines relies mostly on the scan line regions. In order to find horizontal lines in the image, adjacent white vertical regions that are not within a perceived obstacle [13, pp. 71-74] are combined to line segments. Correspondingly, vertical line segments are constructed from white horizontal regions. These line segments and the center points of their regions, called *line spots*, are then projected onto the field. Using linear regression of the line spots, the line segments are then merged together and extended to larger line percepts. During this step, line segments are only merged together if at least a given ratio of the resulting line consists of white pixels in the image. Figure 6 shows the process of finding lines and the center circle in the camera image.

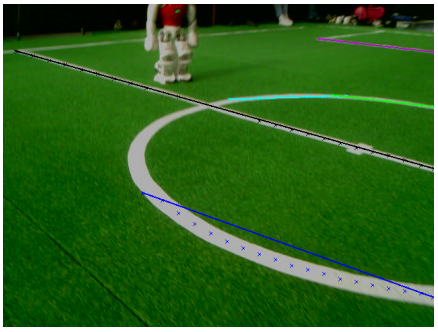


(a) Small wrongly classified regions at the transitions between field and line (b) Transition between field and line with filled in gaps

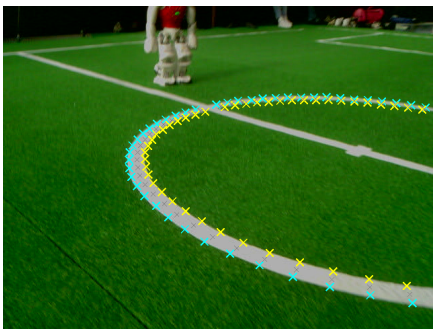
Fig. 5: Filling in of small gaps between a field and a white region. Regions classified as white are shown in red.



(a) Center points of the line spots. The ones found on vertical scan lines are marked in blue and finds on the horizontal scan lines are marked in red.



(b) The line segments built up from the line spots



(c) Circle candidate with points on inner and outer edge marked in yellow and cyan

(d) The final perception of field lines and the center circle

Fig. 6: The process of finding field lines

We don't compute the color segmentation for the whole image any more. Instead, the pixels on the presumed line are compared to their surroundings in order to find out if the line is actually white. In regular intervals, we draw samples from a presumed line, compute two positions above and below the line for each sample and then use these for individual white tests of each pair of sample and comparison position.

To test the hypothesis that the sample point on a supposed line segment is white on green, we expect it to have a considerably higher luminance paired with lower saturation in relation to the comparison points. Additionally, we then test if the sample point satisfies generic thresholds for luminosity and saturation and if the comparison points are in the field color hue range. That effectively results in testing whether the line is white and embedded in green surroundings. Figure 7 shows the positions of the samples and their comparison positions.

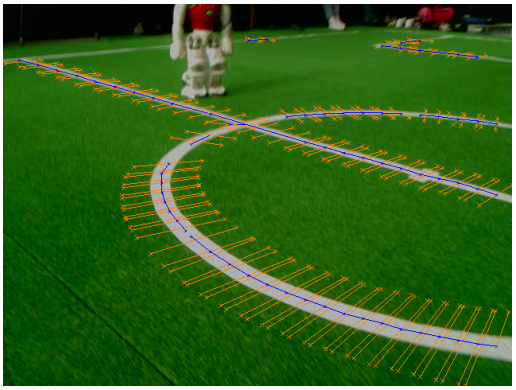


Fig. 7: Sample points (blue) and their corresponding comparison points (orange) for the white test of line segments.

6 Results

At the GORE, we played three games against the other strongest teams in the competition. Since we provided one of the two competition sites used, all our games took place in the other one, i. e. in Dortmund. We won our games 10:0, 8:0, and 10:0. Thereby, we scored 37 % of all goals in this competition of seven teams and achieved the first place. Detailed results of the GORE are provided at the GORE website [3].

At the RoboCup, we won three of the four challenges and became the runner-up in the fourth challenge. In the Obstacle Avoidance Challenge, which required a robot to dribble around obstacles and to score a goal as quickly as possible, two of our three attempts were the fastest ones of all attempts of the fourteen teams that participated in that challenge. In the Passing Challenge, our robots played more passes in all of their three attempts than any other team. In particular, our best attempt with 27 passes in five minutes achieved as many passes as all the other seven teams in all of their attempts combined.

In the Automatic Calibration Challenge, different categories were judged, two time-based and five precision-based. 14 out of 15 times, we achieved the first place in precision-based categories. However, we often only reached the second place in time-based categories, because some teams skipped the calibration phase entirely or finished the main phase very quickly (usually with very low precision results). Overall, we came first place in all remote locations we played at and thereby won the challenge.

In the 1vs1 Challenge, we also used the automatic calibration to prepare for the games. During the games, in contrast to most teams, our robots not only kicked the balls into the opponent half, which was enough to score a point according to the rules. Instead, they actually kicked into the goal, thereby getting the ball back and limiting the opponent's access to balls. Up to the semifinal, this resulted in a point ratio of 57:20 in four games. However, in the final, which

was held in the Hamburg arena, our robot always missed the goal, because its support foot had a different grip than usual during a kick, resulting in a slightly different kick direction. The robot then changed its strategy and just kicked the ball into the opponent half, but that way, it could not compensate for the points that were already lost in the beginning of the game. In the end, the robot of the team HTWK Robots scored one point more than ours, i. e. we lost the final with 12:13 points. Detailed results of all four challenges can found at the SPL website [6].

7 Conclusion

In this paper, we described our research that significantly contributed to our success in the 2021 competitions: autonomous camera calibration as well as a set of robust vision approaches. Although future RoboCup events might be held on site and thus allow more manual tuning again, the new capabilities that allow playing soccer out of the box probably have a high impact. It is now possible to play more friendly games against other teams without the need to travel long distances and without any major loss in performance. Furthermore, the required setup time at a RoboCup event is reduced significantly. However, there are still calibrations left that require automation, such as for precisely kicking over long distances. All algorithms presented in this paper are available as open source and are part of our most recent code release [12], giving other teams similar capabilities.

References

1. Hasselbring, A., Baude, A.: Soccer field boundary using convolutional neural networks. In: Alami, R., Biswas, J., Cakmak, M., Obst, O. (eds.) RoboCup 2021: Robot World Cup XXIV. Lecture Notes in Artificial Intelligence, vol. n/a, p. n/a. Springer (to appear)
2. Laue, T., Moos, A., Göttsch, P.: Let your robot go – challenges of a decentralized remote robot competition. In: Proceedings of the 10th European Conference on Mobile Robots (2021)
3. Moos, A., Göttsch, P.: GORE 2021. https://gore2021.netlify.app (2021), [Online; accessed 15-May-2021]
4. Poppinga, B., Laue, T.: JET-net: Real-time object detection for mobile robots. In: Chalup, S., Niemueller, T., Suthakorn, J., Williams, M.A. (eds.) RoboCup 2019: Robot World Cup XXIII. Lecture Notes in Artificial Intelligence, vol. 11531, pp. 227 – 240. Springer (2019)
5. Reichenberg, P., Röfer, T.: Step adjustment for a robust humanoid walk. In: Alami, R., Biswas, J., Cakmak, M., Obst, O. (eds.) RoboCup 2021: Robot World Cup XXIV. Lecture Notes in Artificial Intelligence, vol. n/a, p. n/a. Springer (to appear)
6. RoboCup Technical Committee: Standard Platform League history, website: https://spl.robocup.org/results-2021/
7. RoboCup Technical Committee: GORE Standard Platform League (NAO) Rule Book (2021), only available online: https://collaborating.tuhh.de/HULKs/gore/-/raw/master/GORe-Rules.pdf

8. RoboCup Technical Committee: RoboCup 2021 virtual tournament (2021), website: https://ssl.robocup.org/robocup-2021-virtual-tournament/
9. RoboCup Technical Committee: RoboCup Standard Platform League (NAO) Challenges & Rule Book (2021), only available online: https://cdn.robocup.org/spl/wp/2021/06/SPL-Rules-2021.pdf
10. RoboCup Technical Committee: Virtual RoboCup Soccer Humanoid League laws of the game 2020/2021 (2021), only available online: https://cdn.robocup.org/hl/wp/2021/06/V-HL21_Rules_v4.pdf
11. Röfer, T., Laue, T., Felsch, G., Hasselbring, A., Haß, T., Oppermann, J., Reichenberg, P., Schrader, N.: B-Human 2019 – complex team play under natural lighting conditions. Lecture Notes in Artificial Intelligence, vol. 11531, pp. 646 – 657. Springer (2019)
12. Röfer, T., Laue, T., Bahr, N., Jaeger, J., Knychalla, J., Lorenzen, T., Matschull, N., Meinken, Y., Monnerjahn, L.M., Plecher, L., Reichenberg, P.: B-Human team report and code release 2021 (2021), https://github.com/bhuman/BHumanCodeRelease/raw/coderelease2021/CodeRelease2021.pdf
13. Röfer, T., Laue, T., Baude, A., Blumenkamp, J., Felsch, G., Fiedler, J., Hasselbring, A., Haß, T., Oppermann, J., Reichenberg, P., Schrader, N., Weiß, D.: B-Human team report and code release 2019 (2019), https://github.com/bhuman/BHumanCodeRelease/raw/coderelease2019/CodeRelease2019.pdf
14. Thielke, F., Hasselbring, A.: A JIT compiler for neural network inference. In: Chalup, S., Niemueller, T., Suthakorn, J., Williams, M.A. (eds.) RoboCup 2019: Robot World Cup XXIII. Lecture Notes in Artificial Intelligence, vol. 11531, pp. 448–456. Springer (2019)