Vision-based Orientation Detection of Humanoid Soccer Robots

Andre Mühlenbrock and Tim Laue

Universität Bremen, Fachbereich 3 – Mathematik und Informatik,
Postfach 330 440, 28334 Bremen, Germany
E-Mail: {muehlenb,tlaue}@informatik.uni-bremen.de

Abstract. Knowing the positions of the other players on a football pitch is a crucial aspect for playing successfully. However, knowing not only the position but also the orientation of another player provides certain tactical advantages. In this paper, we present a vision-based approach for determining the orientation of humanoid NAO robots over short and medium distances. It is based on the idea of analyzing the alignment of other robots' foot sides. In a series of experiments, we demonstrate that the approach is able to perform robust and precise orientation detection in different scenarios.

1 Introduction

Without knowing the positions of other robots, at least in the immediate vicinity, playing soccer in a reasonable manner is barely possible, as collisions and losing possession of the ball will occur frequently. This is why almost all participants across all RoboCup leagues employ some sort of obstacle or robot detection to perform path planning and to support decision making. However, on higher tactical levels, the orientation of opponent robots becomes more and more important. Similar to humans, humanoid robots cannot move in all directions with the same speed. For most robots, moving forward is predominant, requiring an initial rotational movement towards the walk target. Thus, the time to reach a certain position on the field, for instance to intercept a pass, always depends on the initial orientation. Furthermore, humanoid soccer robots have a limited field of view and cannot perceive what is happening behind them, often not even by turning the head. These limitations in motion and perception imply that considering the opponent's orientation in decision making, for instance when choosing the dribble direction or the next pass target, provides a tactical advantage as it contributes to reducing the risk of interference by opponents.

In this paper, we present an approach to solve the first step towards such kind of tactics: detecting the orientation of robots in sensor data. As we use the NAO robot as platform, the approach is vision-based, processing data from the two NAO cameras. Furthermore, as we base our work on the B-Human framework [9], several already existing preprocessing steps are facilitated, including basic robot detection. Our new approach uses the alignment of robot feet to compute

orientations, as this feature turns out to be quite robust, in contrast to a robot's upper body, which might have a complex appearance due to moving arms and a jersey of almost arbitrary style. The major benefits of our approach are its ability to determine orientations even over distances of more than two meters, which is necessary as robots might walk quite fast, as well as its computational efficiency, which is important as the available computing time has always to be shared with many other components.

The remainder of this paper is organized as follows: First, Section 2 discusses related approaches to determine the position and orientation of soccer robots. Afterwards, necessary preliminary works for the orientation detection are presented in Section 3, followed by a description of our approach in Section 4. An evalution of the accuracy and the performance of our approach is presented in Section 5. Finally, the paper concludes in Section 6.

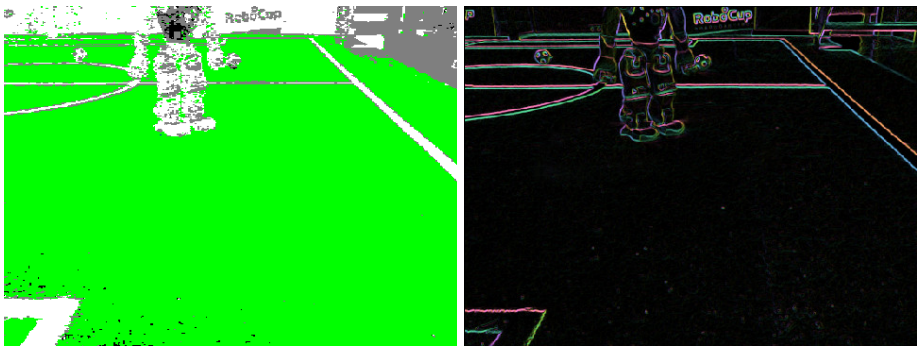
2 Related Work

Early works in the RoboCup Standard Platform did not only ignore any orientation of other robots but also the robot appearance at all. Using AIBO robots, [4] as well as [6] only considered gaps in the perceived field's surface as areas that should be avoided. Algorithms for explicitly detecting other NAO robots have been presented by [8] and [2]. The latter approach also included the determination of the team the detected robot is playing for. This information was used to build a team-wide model of the opponents, as described in [5].

The RoboCup robot league that currently has the most advanced tactical plays is probably the Small Size League. In this league, orientation information is important as the robots' kick devices are always at the front part. However, through having external cameras over the field and standardized colored patterns (that encode position, identity, and orientation) on top of all robots, the detection process is quite simple and is solved by the standardized vision system [12].

A general approach to determine the orientation of objects is template matching. In a preprocessing step, a specific amount of images, containing objects with different orientations, must be analyzed for their features (like color gradients and normals) and saved as templates. To determine the orientation of an input image, its features must be compared with the templates. The orientation of the most similar template will be returned. However, template matching approaches require a lot of computing time. For instance, the software *Linemod* [3] has an average execution time of 119 ms on a 2,3 Ghz Intel CPU. There is also a related approach which was tested in the Sony Four-Legged League using SIFT descriptors. The tested implementation has an average execution time of 300 ms on a 576 MHz CPU [7]. In [11], a decision tree learning approach that considered the alignment of the distinctive AIBO robot jerseys patches was described.

One recent work in the RoboCup Standard Platform League is based on a deep neural network which gets the brightness values of pixels of a robot region as input and returns a two dimensional direction vector [10]. The average error is declared with approximately 8° and the execution time is approximately 0.23 ms



(a) (b)

Fig. 1. (a) A color classified image and (b) a contrast-normalized Sobel image from the upper camera of a NAO.

per robot region. However, it is not clear if this approach also works for moving robots and over which distances the result can be reproduced.

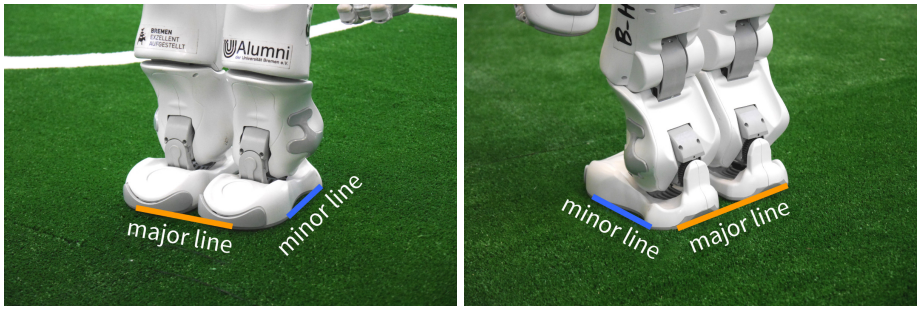
3 Preliminary Works

The approach presented in this paper does not work on plain images but on already preprocessed data. In addition, the basic task of detecting a robot in an image is also carried out by a different component.

In the B-Human framework [9], there are two different preprocessed images that are created from the original camera images. Both can be used for orientation detection. The first one is a color classified image, in which pixels are just classified as black, white, green, or no color (see Fig. 1a). As a first step, classifying a pixel's color is done by applying a threshold to its saturation to classify this pixel as whether colored or non-colored. For the distinction of black and white, there is just another threshold applied to the luminance. Finally, for green, there is a range defined regarding the hue. Due to recent changes in the setup of the Standard Platform League, no more color classes are needed. Robot jersey colors can be chosen arbitrarily and are thus not part of the color classification process.

The second image is the contrast-normalized Sobel image (see Fig. 1b). This is an image in which the pixels represent vectors whose lengths indicate how heavy the pixels' colors change and whose directions show the changing direction. In general, the contrast-normalized Sobel image offers a higher quality in finding silhouettes than the color classified image.

The B-Human framework already contains different implementations for robot detection. The one that has been used for this paper subsamples the color classified image along vertical lines and merges line parts that are not green within union find data structures. Field lines and balls are filtered out by size



(a) (b)

Fig. 2. Illustration of the major line and minor line definition on a (a) forward-facing and a (b) backward-facing robot.

and shape so that only potential robot regions remain. These regions are used for the orientation determination described in this paper. Please note that our approach is in no way limited to this particular player perception software.

4 The Orientation Detection Approach

The approach consists of two steps which both analyze the region around a previously recognized robot's feet. First, we search for two lines – the so-called *major line* and *minor line* – in this region. These two lines already allow calculating an orientation in the range of $[-90°, 90°)$. To obtain the complete orientation in the range of $[-180°, 180°)$, we determine whether the robot is facing forwards or backwards by analyzing two areas in the color classified image over the feet. Both steps turn out to work quite reliably. However, by focussing on the feet, the approach is not able to handle lying robots or robots that have a ball directly at their feet.

4.1 Determining Major Line and Minor Line

To determine the orientation of a robot region, we have defined two lines for which we are searching in the image. This can be done in the contrast-normalized Sobel image as well as in the color classified image. The major line is defined from toe to toe and from heel to heel while the minor line is defined as a side line of a foot. An example is shown in Fig. 2.

To find these lines, we have created the lower silhouette of the feet by scanning the image upwards for each pixel on the lower robot region border. In the contrast-normalized Sobel image, a pixel on the silhouette is found when it exceeds a length threshold. When the color classified image is used, a pixel on the silhouette is found when it is not green and more than two subsequent pixels are also not green. After we created the lower silhouette of the feet, we use *Andrew's*

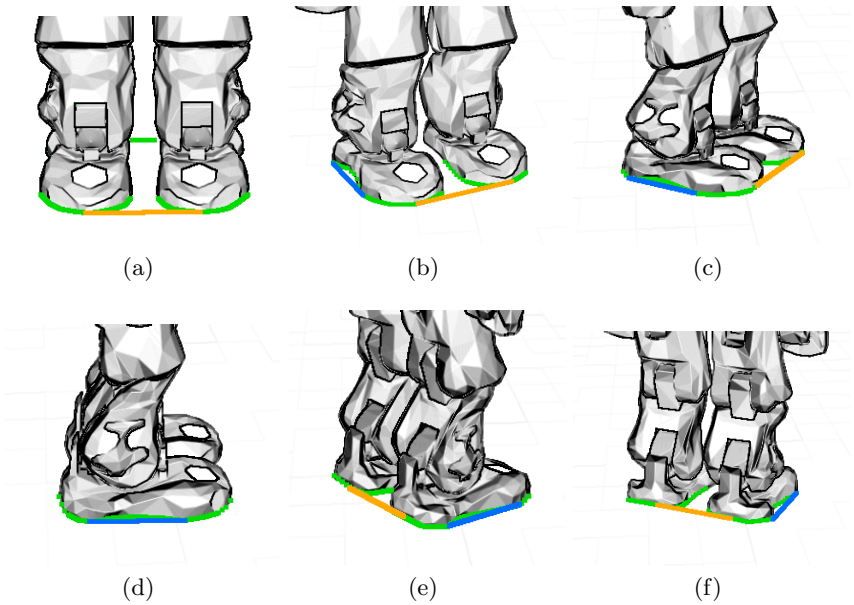
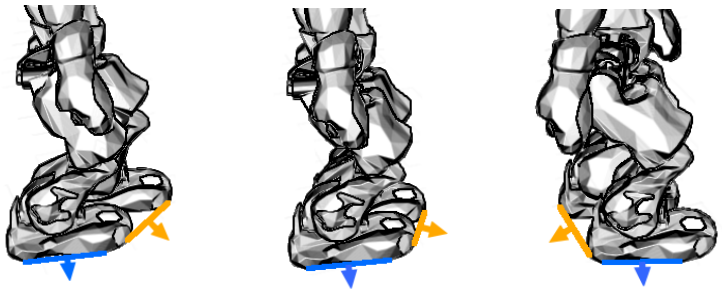


Fig. 3. Illustration of the major line (orange) and minor line (blue) on differently rotated robots. The silhouette is green. It is evident that there is always a greater sum of jumps on the silhouette in the range of the major line than on the silhouette in the range of the minor line.

Monoton Chain Algorithm [1] to determine the lower convex hull. Looking at the longest three edges with a specific minimum length of the lower convex hull, it is evident that two of these are the major line and the minor line.

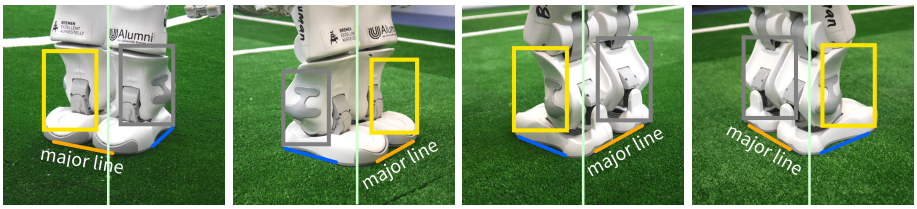
At this point, we have found at maximum three lines and have just to determine which of these lines are the major line and the minor line. For distinction, we calculate the sum of all jumps of neighboring pixels in the silhouette, which have a certain minimum length (the current, experimentally determined length is 2.4 pixels). The idea is that the major line has the greater sum of jumps because there is a gap between both feet (see Fig. 3). More precisely, there are three cases:

- Just one long line: In this case, the robot must have an orientation of approx. $0°$, $90°$, $-90°$ or $-180°$ and the line is parallel to the x-axis, since in all other cases there would be a second long line. When the sum of jumps is zero, we know that the recognized line is the minor line. If the sum of jumps is much larger than zero, we know that the line is the major line (see Fig.3 a/d).
- Two long lines: In this case, the line with the greater sum of jumps is the major line and the line with the smaller sum of jumps is the minor line (see Fig. 3 b/c/e/f).
- Three long lines: This is just a special case which can occur with the NAO's feet and an orientation of nearly $180°$. In this case, the line with the lowest



(a) (b) (c)

Fig. 4. Variation of major line (orange) and minor line (blue) in a walking sequence.



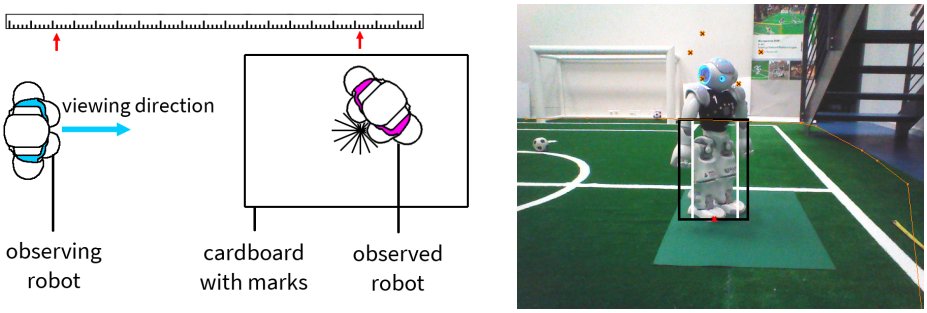
(a) (b) (c) (d)

Fig. 5. Location of the major line (orange) and the area that contains more green pixels (yellow). (a/b) The major line and the yellow marked area are on the same side when the robot is facing forwards and (c/d) they are on a different side when the robot is facing backwards.

sum of jumps is the minor line and the line that has the lowest mid point in the image is the major line.

If both lines have been found, we initially assume that the robot is facing forwards. To calculate the orientation, we transform the major line and minor line from image coordinates in field coordinates and define a vector $\boldsymbol{v}$ parallel to the transformated minor line and calculate the normal vector $\boldsymbol{n}$ to the transformated major line which is directed towards the observing robot. If $\boldsymbol{v} \cdot \boldsymbol{n} \leq 0$, we mirror the vector $\boldsymbol{v}$ so it is directed also towards the observing robot. Now we use the vector $\boldsymbol{v}$ to calculate the orientation $\alpha := atan2(-\boldsymbol{v}_y, -\boldsymbol{v}_x)$.

The reason for not simply using the normal vector $\boldsymbol{n}$ of the major line is that the major line obviously has a higher deviation when the robot is walking (see Fig. 4). If we just found the major line, we simply set $\alpha := 0°$ and add a flag that indicates that there is no information whether the robot is facing forwards or backwards, so the opposite orientation would be also plausible. When we just found the minor line, we set $\alpha := 90°$ and add the same flag, so that we know that the orientation is $90°$ or $-90°$.



(a) (b)

Fig. 6. (a) First experimental setup and (b) an image made by the upper camera of the observing robot. The rectangles indicate the inner and outer robot areas identified by the previously executed robot detection.

4.2 Determing the Facing Direction of a Recognized Robot

Finally, we have to determine whether the recognized robot is facing forwards or facing backwards. To do this, we span two rectangles starting in the most left and the most right point of the silhouette (see Fig. 5 a/b), and determine which of the two areas contains more green pixels.

When the area that contains more green pixels is on the same side as the major line, we know that the robot is facing forwards. When the area with more green pixels is on the opposite side, the robot is facing backwards (see Fig. 5 c/d). If we detect that the robot is facing backwards, we just have to add 180° or −180° to the previously computed orientation α.

5 Evaluation

To evaluate our approach, we performed a number of experiments, differing in the distance between the robots as well as in the robot motions. In all setups, we created two silhouettes in each robot region, the first by the contrast-normalized Sobel image and the second by the color classified image. This allows us to directly compare the results of the contrast-normalized Sobel image and the color classified image. In the following, we will present two of our experimental setups.

5.1 First Experimental Setup: Standing Robots

In the first experimental setup, the observed robot was placed in a distance of 60 cm, 120 cm, 180 cm, and 240 cm to the observing robot and was turned in 22.5°-steps around its axis for each distance. For accuracy, we used a cardboard with barely visible marks every 22.5°. Both robots were just standing still. The setup is depicted in Fig. 6.

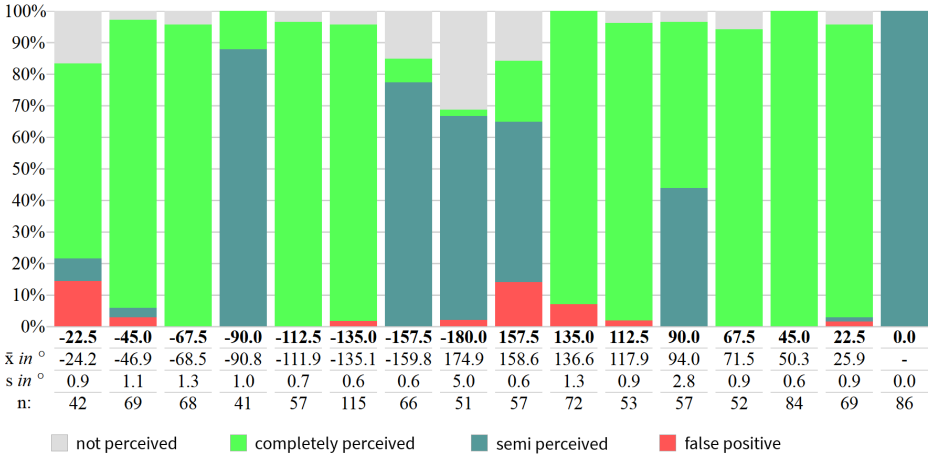


Fig. 7. Results of the first experiment with a distance of 120 cm. The contrast-normalized Sobel image was used here. $\bar{x}$ is the arithmetic mean, s is the standard deviation, and n is the number of evaluated perceptions.

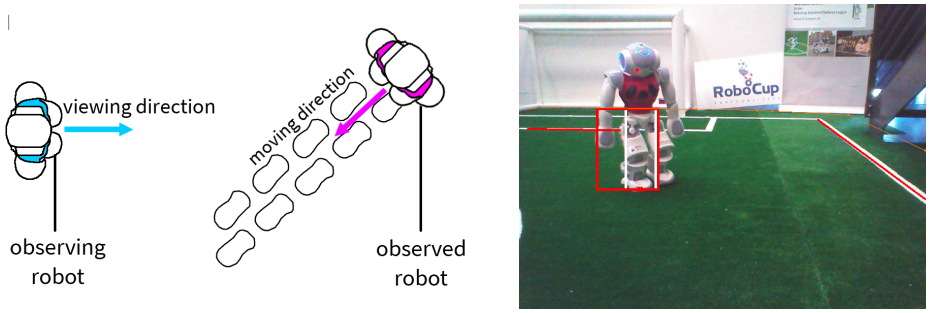
In the evaluation, the returned orientation is classified as *completely perceived* when it does not deviate from the original orientation by more than $\pm$ 11.25° (i. e. more than half of a step). When the algorithm was not able to determine the facing direction but the returned orientation is within the range of tolerance, it is classified as *semi perceived*. The returned orientation is classified as a *false positive* when it deviates by more than $\pm$ 11.25°. The category *not perceived* means that the orientation detection was not successful. The arithmetic mean $\bar{x}$ is calculated over all completely perceived orientations while the standard deviation s is calculated over all completely and semi perceived orientations. The count of evaluated frames n differs from orientation to orientation.

In a nutshell, the results were extremly precise with the contrast-normalized Sobel image. The average false positive rate with 0,3 % at 60cm, 2,5 % at 120cm (see Fig. 7 for details), and 2,3 % at 180cm is extremely low and the standard deviation is in nearly all distances up to 180 cm lower than 4° in nearly all cases. Only at a distance of 240 cm, the results were significantly worse.

5.2 Second Experimental Setup: Walking Observed Robot

In the second experimental setup we made, the observing robot is just standing again while the observed robot is walking in eight different directions with a speed of approximately 12 cm/s in the first execution and with a speed of approximtely 20 cm/s in the second execution (see Fig. 8).

In a preprocessing step of the evaluation, we estimated the ground truth orientation by calculating the rounded average of all returned orientations except for a few runaway values because we did not use the cardbord for exact positioning and had no ground truth tracking system available. The returned orientation



(a) (b)

Fig. 8. (a) Second experimental setup and (b) an image made by the upper camera of the observing robot. The rectangles indicate the inner and outer robot areas identified by the previously executed robot detection.

is classified as completely perceived or semi perceived when it does not deviate from the expected orientation by more than (a) $\pm$ 22.5° in the first execution and (b) $\pm$ 15° in the second execution. We chose a higher tolerance in the first execution as the robot turned slightly while walking.

In some further experiments, the observing robot was also moving (shaking body and turning head) while the observed robot stands or was also moving. In these situations, the standard deviation only increases by approximately 5 degrees while the perceiving rate was just slightly affected.

In summary, the results are good. The ratio of false positives to correctly perceived orientations is in the range of $\frac{3}{50}$ to $\frac{1}{3}$ depending on the orientation. In most cases, the reason for the incorrect perception was the wrong assignment of major and minor lines, so the returned orientation differs about 90°. The standard deviation is in the range of 3.0° to 11.2° depending on the orientation. With the use of a downstream modeling component, it seems to be possible to use the orientation information returned by this algorithm to make decisions. Extended results are shown in Fig. 9.

5.3 Performance

We measured the execution time of single code sections as well as of the total orientation detection code as a function of the distance to the observed robot. We expected that the execution time is shorter at a higher distance, as the analyzed image region that contains the robot feet is smaller. The measurement was done just with standing robots but one can assume that there will not be a great difference to walking robots. The algorithm was processed by a NAO.

At a distance of 60 cm, the approach required the longest execution time with 0.53 ms. The scanning of the silhouette consumes the most time with 0.37 ms in the contrast-normalized Sobel image. It is noticeable that the scanning of

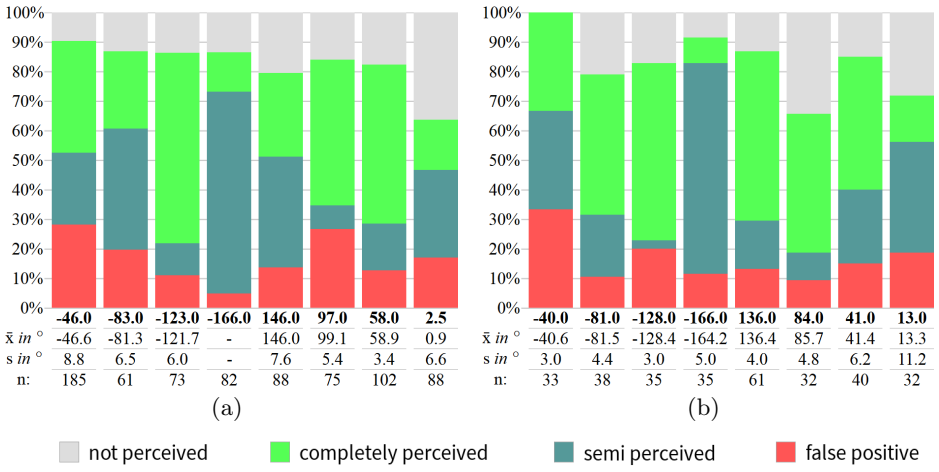


Fig. 9. Results of the second experiment. The walking speed is (a) approximately 12 cm/s in the first execution and (b) approximately 20 cm/s in the second execution. The contrast-normalized Sobel image was used here. $\bar{x}$ is the arithmetic mean, s is the standard deviation, and n is the number of evaluated perceptions.

the silhouette in the color-classified image takes only 0.13 ms. This is caused by some operations which are done for every pixel in the contrast-normalized Sobel image while the pixels can be read directly from the color-classified image.

The function of the total execution time in figure 10 decreases at longer distances. To also reduce the execution time in a short distance a bit, we could reduce the sampling rate for the silhouette. As we have seen in the evaluation, this should not take a great impact on the quality of the result.

6 Conclusion and Future Work

In this paper, we presented an approach that is able to recognize the orientation of humanoid robots that are standing or walking on a soccer field. Our approach is both very fast and robust, as the evaluation results show. Furthermore, it works in combination with color-classified images as well as with the output of an edge detection, as Fig. 11 shows. The approach has been developed for and tested in the RoboCup Standard Platform League only, but we assume that the general idea can be transferred easily to other humanoid robots.

We are currently working on the integration of the orientation data into our Kalman Filter-based robot tracking. This modeling component will estimate the robots' orientation over time and thus probably be able to compensate most ambiguities of semi perceived robot orientations. Furthermore, team behaviors, which will be able to exploit this new kind of information, are about to become developed for the 2017 RoboCup competitions.

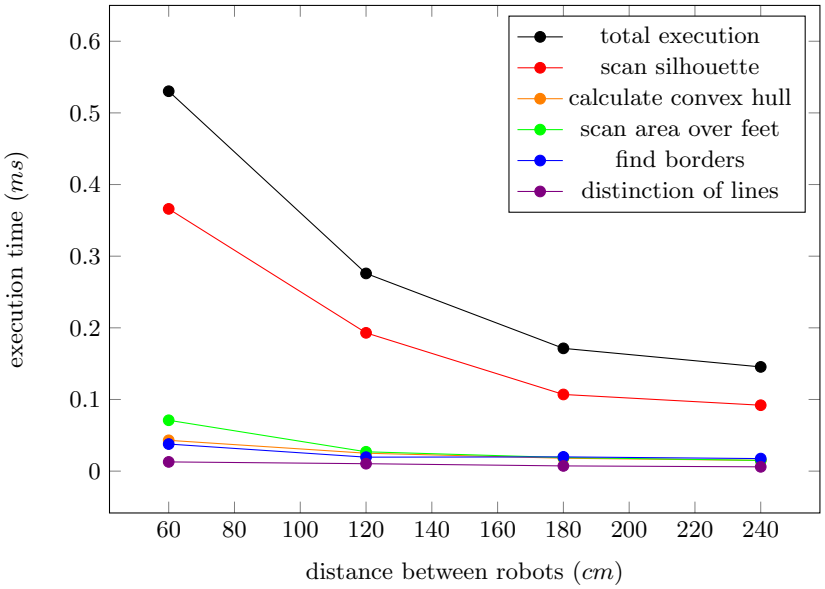


Fig. 10. Average execution time as a function of the distance to the observed robot. The contrast-normalized Sobel image was used for this experiment.

References

1. Andrew, A.M.: Another efficient algorithm for convex hulls in two dimensions. Information Processing Letters 9(5), 216–219 (1979)
2. Fabisch, A., Laue, T., Röfer, T.: Robot recognition and modeling in the robocup standard platform league. In: Pagello, E., Zhou, C., Behnke, S., Menegatti, E., Röfer, T., Stone, P. (eds.) Proceedings of the Fifth Workshop on Humanoid Soccer Robots in conjunction with the 2010 IEEE-RAS International Conference on Humanoid Robots. Nashville, TN, USA (2010)
3. Hinterstoisser, S., Lepetit, V., Ilic, S., Holzer, S., Bradski, G., Konolige, K., Navab, N.: Model Based Training, Detection and Pose Estimation of Texture-Less 3D Objects in Heavily Cluttered Scenes. In: Computer Vision – ACCV 2012: 11th Asian Conference on Computer Vision, Daejeon, Korea, November 5-9, 2012, Revised Selected Papers, Part I. pp. 548–562. Springer (2013)
4. Hoffmann, J., Jüngel, M., Lötzsch, M.: A vision based system for goal-directed obstacle avoidance. In: Nardi, D., Riedmiller, M., Sammut, C., Santos-Victor, J. (eds.) RoboCup 2004: Robot Soccer World Cup VIII. Lecture Notes in Artificial Intelligence, vol. 3276, pp. 418–425. Springer (2005)
5. Laue, T., Röfer, T., Gillmann, K., Wenk, F., Graf, C., Kastner, T.: B-Human 2011 – eliminating game delays. In: Röfer, T., Mayer, N.M., Savage, J., Saranlı, U. (eds.) RoboCup 2011: Robot Soccer World Cup XV. Lecture Notes in Artificial Intelligence, vol. 7416, pp. 25–36. Springer (2012)
6. Lenser, S., Veloso, M.: Visual sonar: Fast obstacle avoidance using monocular vision. In: Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003). vol. 1, pp. 886–891. Las Vegas, USA (2003)

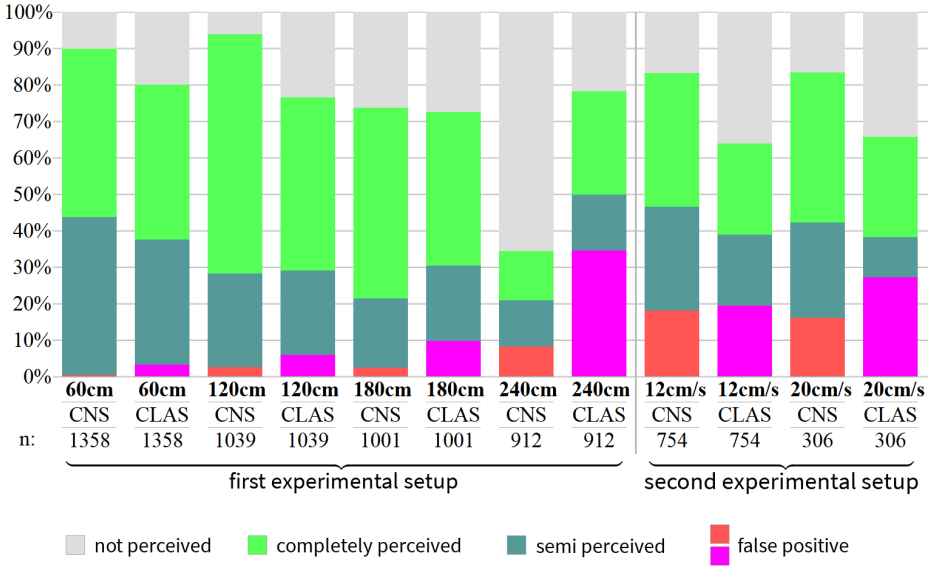


Fig. 11. Combined results of the first and second experimental setup. In addition to the silhouette, which is calculated by using the contrast-normalized Sobel image (CNS), we also calculated a second silhouette with the color classified image (CLAS).

7. Loncomilla, P., Ruiz-del Solar, J.: Gaze Direction Determination of Opponents and Teammates in Robot Soccer. In: RoboCup 2005: Robot Soccer World Cup IX. pp. 230–242. Springer (2006)
8. Metzler, S., Nieuwenhuisen, M., Behnke, S.: Learning visual obstacle detection using color histogram features. In: Röfer, T., Mayer, N.M., Savage, J., Saranli, U. (eds.) RoboCup 2011: Robot Soccer World Cup XV. Lecture Notes in Computer Science, vol. 7416, pp. 149–161. Springer (2011)
9. Röfer, T., Laue, T., Kuball, J., Lübken, A., Maaß, F., Müller, J., Post, L., Richter-Klug, J., Schulz, P., Stolpmann, A., Stöwing, A., Thielke, F.: B-Human team report and code release 2016 (2016), `https://github.com/bhuman/BHumanCodeRelease/blob/master/CodeRelease2016.pdf`
10. Tilgner, R., Reinhardt, T., Kalbitz, T., Seering, S., Fritzsche, R., Eckermann, S., Müller, H., Engel, M., Wünsch, M., Mende, J., Freick, P., Stadler, L., Schließer, J., Hinerasky, H.: Sparse Feature Learning for Visual Robot Angle Estimation (2013), `http://www.tzi.de/spl/pub/Website/Challenges2013/HTWK.pdf`
11. Wilking, D., Röfer, T.: Real-time object recognition using decision tree learning. In: RoboCup 2004: Robot World Cup VIII. Lecture Notes in Artificial Intelligence, vol. 3276, pp. 556–563. Springer (2005)
12. Zickler, S., Laue, T., Birbach, O., Wongphati, M., Veloso, M.: SSL-vision: The shared vision system for the RoboCup Small Size League. In: Baltes, J., Lagoudakis, M.G., Naruse, T., Shiry, S. (eds.) RoboCup 2009: Robot Soccer World Cup XIII. Lecture Notes in Artificial Intelligence, vol. 5949, pp. 425–436. Springer (2010)