A Ball Is Not Just Orange: Using Color and Luminance to Classify Regions of Interest

Hannes Schulz, Hauke Strasdat, and Sven Behnke
Computer Science Institute
University of Freiburg, Germany
{ schulzha | strasdat | behnke }@informatik.uni-freiburg.de

Abstract— **In soccer, the ball is the one object on the field that a player must attend to at all times. If a player looses track of the ball position, it cannot continue play but must search for it. For this reason, the ball in RoboCupSoccer is colored in orange, which makes it possible to localize it in camera images by classifying individual pixels to color classes. This simple approach has several problems. For example, other orange objects next to the field might be confused with the ball. Another problem is motion blur, in particular for humanoid robots capturing images while walking fast. The presence of orange and green at the same pixel during exposure leads to a mixture of the two colors: brown, a common color in the images. Ball detection based solely on color must fail in such a case.**

To overcome these problems, we propose a two-stage system for ball detection and tracking. First, an extended color class is used to find ball candidates. Both, color and luminance from small windows around the candidate locations are classified by a neural network, which has been trained on a large set of balls and distractors. Thus, in addition to color, the network can analyze the shape of the object of interest as well as its shading, including typical highlights and shadows. A detected ball is tracked in a small window in order to achieve a high frame rate on a Pocket PC. This also focuses the attention of the system onto the tracked ball. We evaluated the proposed approach on our NimbRo KidSize 2006 and 2007 robots. The experiments indicate that the ball can be reliably detected and confusion with orange non-ball objects can be avoided.

I. Introduction

In RoboCupSoccer, reliable visual perception is essential for successful play. Because reliable computer vision for real-world situations is beyond the current state of the art, the objects on the RoboCupSoccer fields are color-coded. In the Humanoid League, for example, the goals are yellow and blue, the robots are black with magenta or cyan team markers, the field is green with white lines, and the ball is orange. Color-classification [1]–[7] is commonly used as a first step in the computer-vision systems. This simple approach makes it possible to run computer vision in real-time onboard the small computers the robots are equipped with. It discards, however, detailed information about hue, saturation, and luminance of pixels. While for larger objects, such as the goals, color classification and simple geometric analysis suffice for reliable detection, the perception of the ball is more difficult. The ball moves quickly on the field and outside the field, it is relatively small, and it lacks verifiable internal structure. Common problems in ball detection include false detections in orange objects, wooden floors, or exposed skin next to the field as well as failed detections due to motion blur. If the ball moves quickly relative to the camera, the presence of orange and green at the same pixel during exposure leads to a mixture of the two colors: brown, a common color in the images. Fig 1(b) shows such a blurred ball captured from a walking humanoid robot [8]. Ball detection based solely on color must fail in such a case.

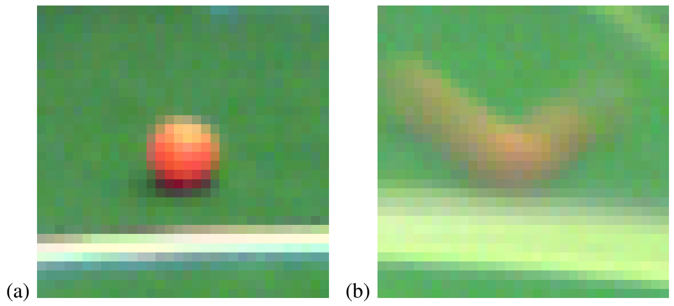

Fig. 1. The orange ball behind a white line on a green field. (a) Clean image captured from a standing robot. Typical highlights and shadows are visible. (b) Same situation with motion blur due to humanoid walking movements. The green blends with the orange to a brownish color.

The ball is not only most difficult to detect, but also the most important object on the field. In fact, the ball is the one object that a player must attend to at all times. If a player looses track of the ball position, it cannot continue play, but must search for it. Both difficulty and importance of ball detection warrant special attention for the ball.

In this paper, we propose a two-stage system for ball detection and tracking. First, in addition to orange a brownish color class is used to find ball candidates. Both, color and luminance from small windows around the candidate locations are classified by a neural network, which has been trained on a large set of balls and distractors. Thus, in addition to color, the network can analyze the shape of the object of interest as well as its shading, including the typical highlights on the upper side of the ball and shadows below the ball, which are visible in Fig. 1(a). A detected ball is tracked in a small window in order to achieve a high frame rate on a Pocket PC. This also focuses the attention of the system onto the tracked ball.

The remainder of the paper is organized as follows. After we review various approaches for ball detection in the next section, we present the proposed framework for ball detection and tracking in Sec. III. We evaluate our system using captured video and online experiments with our NimbRo KidSize robots. The results are presented in Sec. IV. The paper concludes with a discussion and ideas for future work.

II. Related Work

Several approaches to make ball detection less dependent on color classification have been proposed in the literature. In addition to its orange color, the spherical shape of the ball is frequently utilized. Coath and Musumeci [9], for example, proposed a method to detect the ball using arc fitting through three consecutive points along detected edges. The arcs vote for the ball center. This approach is able to detect partially occluded balls, but it relies on reliable edge detection and is applicable only for balls covering many pixels.

Template-based approaches track the ball using a circular template [10], [11]. Regions on both sides of the template border are described by color histograms. While this approach improves upon hard color classification, it ignores important spatial details. Modeling the top side of the ball in the same region as the bottom side makes it impossible to utilize typical highlights and shadows caused by the spherical ball shape that is illuminated from above.

A trainable approach to ball detection was proposed by Mitri et al. [12]. They first apply edge detection to a region of interest. This discards color information as well as shading. A classifier cascade is trained on Haar-like features. This approach detects balls, but the classifier reacts also to other circular objects such as wheels and heads.

Another classifier-based system has been proposed by Mayer et al. [13] for the detection of robots in MiddleSize League. Regions of interest are classified based on simple region descriptors and orientation histograms. This also discards valuable image information. Hence, the applicability to ball detection is unclear.

III. Ball Detection and Tracking

The proposed ball detection system consists of two parts. First, potential *ball candidates* are detected in sub-sampled images which represent color classes. Second, ball candidates are verified by a neural classificator using color and luminance information of small regions around them. A detected ball is tracked in a small window in order to achieve a high frame rate.

A. Detecting Ball Candidates

Our Pocket PC camera captures images with a resolution of 640×480 pixels at a rate of 15fps in YUV 4:2:2 color space. The individual pixels are classified to color classes as follows. First, the Y-component is compared to luminance thresholds for classification of black and white. For pixels with intermediate luminance, the color class is defined by a look-up-table for the U and V values. Color classes are described by ellipses in the UV plane. In addition, each color class is restricted to an interval in the Y dimension. The ball is covered by two color classes, which are illustrated in Fig. 2: Class *Orange* represents pure orange pixels. Class *OrangeCandidate* includes the first one, but it is extended towards green to account for the mixing of the field color and the ball color due to motion blur.

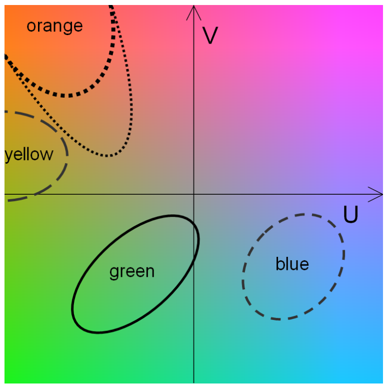

Fig. 2. Color classes as ellipses in the UV plane: The big-dotted ellipse represents the *Orange* class, whereas the small-dotted ellipse defines the *OrangeCandidate* class. The *Green* color class is defined by the unbroken ellipse. The yellow and blue goal are indicated by the dashed ellipses.

For each color class, the pixels belonging to it are counted in an 80×60 grid. Each pixel in this *color class image* represents the number of occurrences of its color in a 8×8 window of the original image. While this subsampling reduces spatial resolution, it allows for quickly assessing the density of the corresponding color in image regions.

In the next processing step, orange-cells containing too few orange pixels are set to zero. Also, orange pixels that have too little green in their neighborhood are removed. Because the Bayer pattern of the camera induces orange and cyan colors at sharp contrasts, *Orange* is also inhibited at the edges of white regions, such as field lines.

To find regions of interest that could contain balls, modes are searched for in the *Orange* class image. The regions of interest are detected in the following order. As an initial guess, the maximal cell in the color class image is identified. Starting from this point x_0, *Mean Shift* [14] – a fast, nonparametric estimator of density gradient – is used to find the closest mode. Iteratively, the center of color mass within a small search window W around x_k is computed. In the next iteration, x_{k+1} is shifted to it.

$$x_{k+1} \leftarrow \frac{\sum_{(i,j)\in W(x_k)} : \begin{pmatrix} i \\ j \end{pmatrix} \cdot \text{cell}(i,j)}{\sum_{(i,j)\in W(x_k)} : \text{cell}(i,j)} \tag{1}$$

This assignment is repeated until convergence which typically needs only a few iterations. Fortunately, *Mean Shift* provides a the position of the peak with sub-cell accuracy, so the corresponding pixel in the original image can easily be estimated. The mode-finding procedure is repeated four times, so that up to four ball candidates are detected. In doing so, cells are skipped which were visited previously. If less than four modes are found in the *Orange* color class image, we continue our search in the upper half of the *OrangeCandidate* class image. The lower half of this image is not searched because it

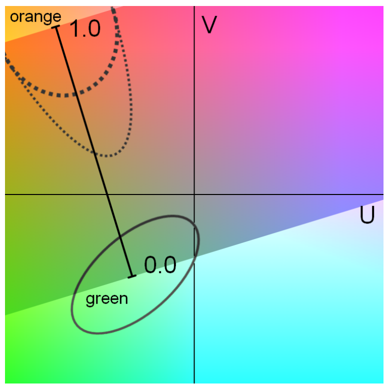

Fig. 3. Representation of the orange-greenness. UV color values are projected orthogonally onto the line between *Orange* (1.0) and *Green* (0.0). Projections outside this interval are saturated.

contains balls close to the robot that are large enough to still be classified as *Orange* under motion blur.

The detected ball candidates are verified as follows. In case a ball candidate is found in the lower half of the image, which is close to the robot, the verification process is straightforward. If it represents a mode in the *Orange* class image, it is deemed to be a ball. Other glaring orange objects should not be on the green field next to the robot.

B. Neural Ball Classification

Ball candidates found in the upper half of the image are more challenging because they consist of fewer pixels, are more strongly affected by motion blur and are more likely to correspond to distractors outside the field. They are classified by a neural network which has been trained to distinguish between balls and non-balls.

The following preprocessing steps are performed in order to represent the region of interest in a format suitable for neural classification. The region of interest is centered at the ball candidate. It is represented as a constant-size (48×48 pixels), two-channel image.

The first image channel carries luminance information. Normalization for average intensity or contrast is not necessary because our camera already adjusts the brightness automatically. This makes the YUV image invariant to global illumination changes.

The second image channel represents color information along a orange-green axis. We will refer to it as the *orange-greenness*. The orange-greenness is computed as illustrated in Fig. 3. Each color value in the UV plane is projected orthogonally onto the axis between the center of the *Orange* color class and the *Green* color class. The *orange-greenness* values are normalized such that *Green* corresponds to zero and *Orange* corresponds to one. Projections outside the interval between the centers of the *Green* and *Orange* colors are saturated to zero or one, respectively. We choose this particular color representation because it is specific for the task at hand, and because it can be quickly adapted to changes in object color and changes in illumination. To adapt to new conditions,

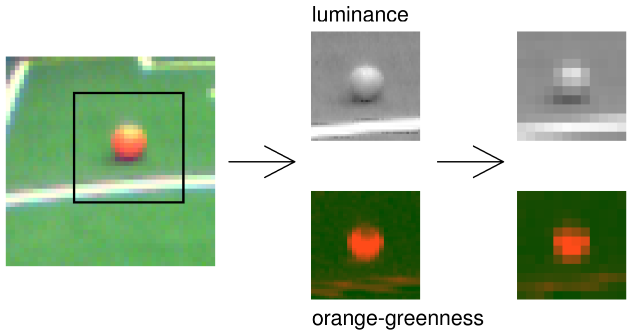

Fig. 4. Preprocessing of regions of interest. The YUV image is represented using two channels (luminance and orange-greenness) and sub-sampled to 12×12 pixels.

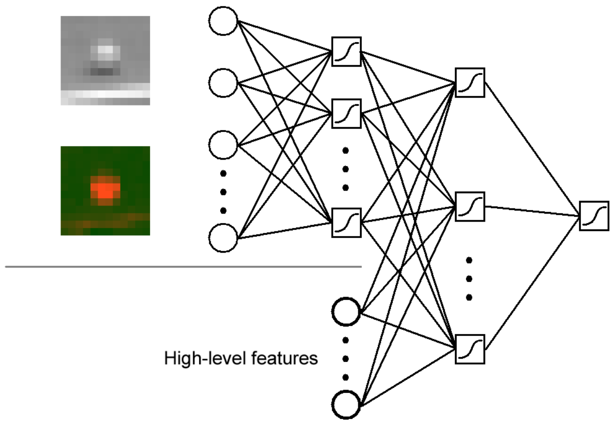

Fig. 5. Network architecture: High-level features are given to the neural network *in addition to* the image-like representation.

only a calibration of the color centers is necessary. The orange-greenness is invariant to such changes. Hence, the neural network needs no retraining.

To reduce the number of parameters for the neural network, the two-channel images are subsampled to 12×12 pixels. This yields a 288-dimensional feature vector that is presented as input to the neural network. The preprocessing steps are illustrated in Fig. 4.

Further preprocessing steps are deliberately omitted to give the neural net the opportunity to identify characteristic ball features itself. For instance, rotation invariant representations would decrease the dimensionality of the feature vector significantly but would also discard important spatial detail. This is undesirable since the spherical balls always show dark shadows in the lower half and bright highlights on the top. Due to the motion blur, reliable segmentation of the ball and the background would be extremely difficult. Image-like representations preserve the relevant spatial relations and avoid the need for segmentation.

In addition to the image-like feature vectors, we also feed higher-level features into the first hidden layer of the neural network. This is illustrated in Fig. 5. In brief, our neural network has a input dimension of 288. The first hidden layer consists of 36 nodes plus 4 high-level features. The network has a second hidden layer with 12 nodes. The output is a scalar between 0 and 1. All nodes have sigmoidal activation functions.

Four high-level features are added, two calculated from the orange-greenness image, and two from the luminance one. In the orange-greenness image the features are sums over circular regions. The first one is a circular region in the center of the image, which is supposed to be orange if a ball is observed. The second feature is the sum over a ring-shaped region outside the center, which is typical non-orange for ball images. In the luminance image, both features are vertical gradients located in the center with different kernel sizes. This reflects the fact that balls have often a high-light on the top and are always dark at the bottom. Finally, a sigmoid transfer function is applied to all four features.

By adding such high-level features in the first hidden layer, domain knowledge can be expressed and offered to the neural network. The network is free to use these features for classification. Because it has access to the image-like representations, the network is not restricted to the features chosen by the designers but it can also construct other features useful for classification in its hidden layer.

The network is trained to output one for balls and zero for non-balls. From the ball candidates presented to the network, the ball candidate with highest output value is selected as the ball. If all output values remain below 0.5, no ball is detected and the ball is deemed non-visible.

C. *Ball Tracking*

If the ball has been detected in the last frame, not the whole image but only a small window of 96×96 pixels around the last ball occurrence is processed. We use the last ball position as predictor for the current ball, because the walking movements of our humanoid robots cause back-and-forth camera motion that is hard to model in more detail. For a wheeled robot, a Kalman predictor would be appropriate here. The focusing to a small window does not only enhance the frame rate significantly (from ca. 5fps to ca. 9fps). It also focuses the search for potential ball candidates to the relevant image parts. Thus, the ratio of true balls among the four ball candidates is raised. In principle, tracking a wrong ball hypothesis could lead to non-detection of better ball-candidates appearing outside the tracking window. To prevent this undesired effect, we fully process every fourth frame. In these frames, the other objects on the field are also perceived.

IV. Experiments

A. *Neural network training*

A set of 600 examples, 160 balls and 440 distractors, all extracted ball candidates as described above, is produced. The set is divided randomly into a training set (480) and a test set (120). A neural network is trained on the training set using RPROP [15]. Using much more distractors than balls for the training process reduces the probability of an output close to 1.0. Thus, it reduces the number of false positive detections.

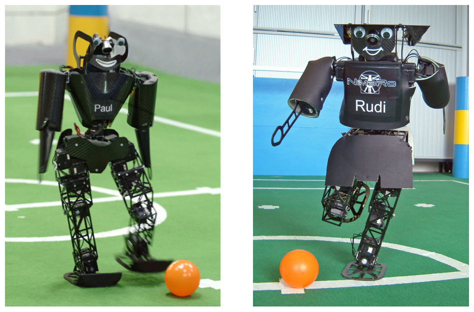

Fig. 6. Robots used for the experiments: NimbRo KidSize 2006 (left) with Pocket PC and two cameras and NimbRo KidSize 2007 (right) with a tiny PC and three cameras.

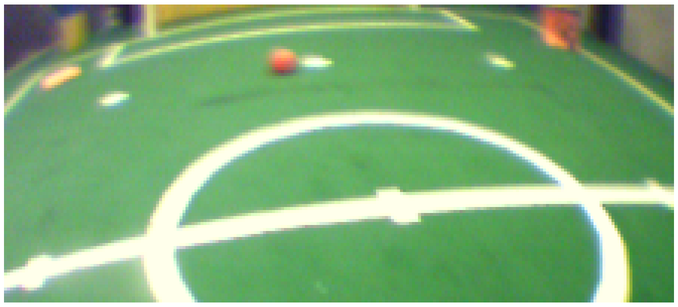

Fig. 7. Real-time experiment: The robot is supposed to approach the ball (middle). Other orange objects – a felt cloth (left) and a book (right) – are placed on the field in order to distract the robot.

In order to avoid overfitting, the training is stopped as soon the total sum of the squared error drops below 1.0.

B. *Recognition performance*

The intended generalization property is evaluated on the test set, consisting of 32 balls and 88 distractors. The mean squared error for tested balls is 0.0466 and the maximal error is 0.8538. Only one of the balls has not been detected (1 false negative out of 32).

The mean squared error for tested distractors is 0.0008, with a maximal error of 0.1851. This means that all distractors have been classified as non-balls (0 false positive out of 88).

C. *Real-time task*

In the second experiment, the trained neural network is evaluated in closed-loop running on the Pocket PC onboard a NimbRo KidSize 2006 robot. The robot is shown in the left part of Fig. 6. A ball as well as other orange objects - like a book or an felt cloth - are placed on the field, as shown in Fig. 7. In order to ensure a fair experiment, these orange distractors are not trained as negative examples beforehand. The humanoid is supposed to approach the ball without being distracted by the other orange objects. This experiment is performed twice: once on a robot using simple orange-center-green-surround ball template, another time on a robot using the presented two-stage framework. For the purpose of evaluation, a video is recorded on the robot.

In the real-time experiment using the simple ball template, the humanoid approaches the ball only by chance. As expected, it is distracted by the other orange objects. When

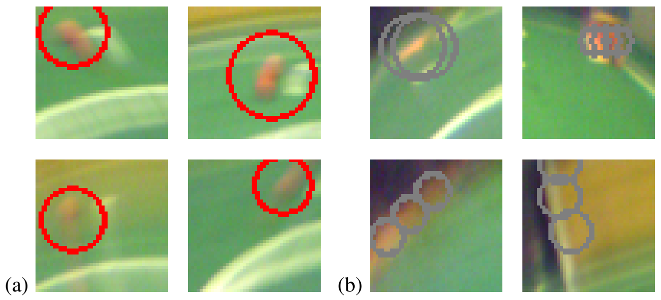
Fig. 8. Ball candidates classified by a neural network. (a) True positive classifications of the ball are shown in red circles. (b) Distractors – a felt cloth, a book, wooden floor and a yellow goal post (top left to down right) – are classified correctly as non-balls.

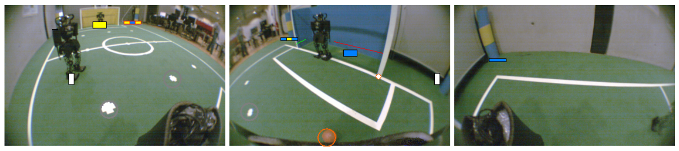
Fig. 9. Images captured by the three cameras of a NimbRo KidSize 2007 robot. The robot can see in all directions above the horizon. Objects close to the robot are also in the field-of-view.

the presented two-stage framework is used, its movements are much more directed towards the ball. We examined the recorded video after the experiment. Although the ball is often blurred and the other orange objects are prominent in the picture, the neural network makes the right classification decision in almost all cases. Fig. 8 shows some of the ball and the non-ball candidate regions.

D. Transfer to NimbRo KidSize 2007 robots

After RoboCup 2007, we ported our two-stage ball detection system, which was developed for our KidSize 2006 robots, to our new NimbRo KidSize 2007 robots [16]. These are equipped with three ultra-wide angle WVGA USB cameras which provide them with an omnidirectional field-of-view, as shown in Fig. 9. In contrast to the Pocket PC, the uEye industrial USB cameras have much less motion blur, because they feature a global shutter for which we configured a short exposure time of 15ms.

We compiled an image data set of 273 balls and 548 non-balls under varying lighting conditions, randomly split in a training set of 521 samples, a validation set of 108 samples, and a test set of 192 samples. To account for the capabilities of the new camera, we chose a difficult set of distractors containing ball-sized orange boxes and orange toys (Fig. 10). The averaged distractor and ball images can be seen in Fig. 11. Please note that balls, as opposed to distractors, show a characteristic top-down gradient in luminance, while in the orange-greenness the average ball looks like a bright disc.

For the NimbRo KidSize 2007 ball classification we use a neural classifier similar to the one presented in previous sections. We compared the performance of the network with a k-Nearest-Neighbour (KNN) classifier with $k = 5$ and a

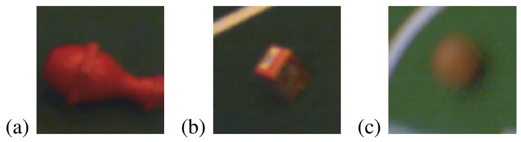
Fig. 10. Sample Stimuli for NimbRo KidSize 2007 robots: (a) orange toy (b) orange box (c) ball

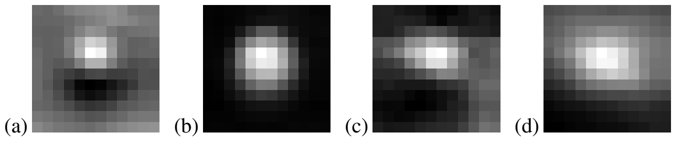
Fig. 11. Averaged neural-network stimuli. (a) Luminance image of balls (b) Orange-greenness of balls (c) Luminance of non-balls (d) Orange-greeness of non-balls.

neural classifier with one hidden linearly activated neuron.

While the KNN classifier achieved an accuracy of 88.5% on the test set, the linear neural net classifies 86.5% correctly. Our neural classifier with 4 hidden units achieved a 91.1% accuracy. Thus, the two-stage system can reliably distinguish balls from other orange objects on the field.

We further analyzed the learned classification by flipping the balls in the test set upside-down and presenting them to the network. Now, the KNN achieves an accuracy of 76.6%, the linear classifier 70.0% and our neural classifier with four hidden neurons 74.0%. The performance drop suggests, that the luminance gradient on the ball is an important characteristic of the ball.

The invariance of orange-greenness to lighting conditions was tested by training a neural classifier on a subset of the balls where lighting conditions differed significantly from lighting conditions in the test set. Even here, a neural classifier with seven hidden units achieved an accuracy of 88.2%.

V. Conclusion

In this paper, we presented a two-stage system for ball detection and tracking. The system finds regions of interest using a ball-candidate color. The ball candidates are classified by a neural network, which can make use of both color and luminance information. Detected balls are tracked in real-time on a Pocket PC and a tiny PC.

The experimental results show that our system is able to ignore orange objects, which typically would lead to false detections. Moreover, the neural classifier is able to detect balls that are severely blurred. Both effects increase the reliability of the ball perception, in particular for balls far-away from the robot. Being able to see the ball at larger distances directly improves performance in the soccer games.

The system is practical for the use in the RoboCupSoccer domain, but it requires some effort for color calibration and the annotation of captured video. In future work, we would like to incorporate semi-automatic color calibration and semi-automatic generation of ball and non-ball examples in order to reduce the work load for setup.

We also plan to apply the neural classifier to verify detections of other small-sized objects on the field, such as goal post footers, corners, junctions and crossings of field lines, as well as other robots.

Acknowledgment

This work was funded by German Research Foundation (DFG) under grants BE 2556/2-2, /4.

References

[1] M. Simon, S. Behnke, and R. Rojas, "Robust real time color tracking," in *RoboCup-2000: Robot Soccer World Cup IV*, ser. LNCS 2019. Springer, pp. 239–248.

[2] Z. Wasik and A. Saffiotti, "Robust color segmentation for the robocup domain," in *Proceedings of 16th International Conference on Pattern Recognition (ICPR'02)*, vol. 2, 2002, pp. 651–654.

[3] I. Dahm, S. Deutsch, M. Hebbel, and A. Osterhues, "Robust color classification for robot soccer," in *RoboCup 2003: Robot Soccer World Cup VII*, ser. LNCS 3020. Springer, 2004, pp. 677–686.

[4] M. Jüngel, "Bayesian color estimation for adaptive vision-based robot localization," in *Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Sendai, Japan*, 2004, pp. 1884–1889.

[5] ——, "Using layered color precision for a self-calibrating vision system," in *RoboCup 2004: Robot Soccer World Cup VIII*, ser. LNCS 3276. Springer, 2005, pp. 209–220.

[6] L. Iocchi, "Robust color segmentation through adaptive color space transformation," in *Proceedings of 10th RoboCup International Symposium, Bremen, Germany*, 2006.

[7] M. Sridharan and P. Stone, "Color learning on a mobile robot: Towards full autonomy under changing illumination," in *The 20th International Joint Conference on Artificial Intelligence (IJCAI'07)*, January 2007, pp. 2212–2217.

[8] S. Behnke, M. Schreiber, J. Stückler, R. Renner, and H. Strasdat, "See, walk, and kick: Humanoid robots start to play soccer," in *Proceedings of 2006 IEEE-RAS International Conference on Humanoid Robots (Humanoids'06), Genoa, Italy*, 2006, pp. 497–503.

[9] G. Coath and P. Musumeci, "Adaptive arc fitting for ball detection in robocup," in *Proceedings of APRS Workshop on Digital Image Computing, Brisbane, Australia*, 2003.

[10] R. Hanek, T. Schmitt, S. Buck, and M. Beetz, "Toward robocup without color labeling," *AI Magazine*, vol. 24, no. 2, pp. 47–50, 2003.

[11] S. Olufs, F. Adolf, R. Hartanto, and P. Plöger, "Towards probabilistic shape vision in RoboCup: A practical approach," in *Proceedings of 2006 RoboCup Symposium, Bremen, Germany*, 2006.

[12] S. Mitri, K. Pervölz, H. Surmann, and A. Nüchter, "Fast color-independent ball detection for mobile robots," in *Proceedings of the IEEE International Conference Mechatronics and Robotics 2004 (MechRob '04), Aachen, Germany*, 2004, pp. 900–905.

[13] G. Mayer, J. Melchert, H. Utz, G. Kraetzschmar, and G. Palm, "Neural robot detection in robocup," in *Biomimetic Neural Learning for Intelligent Robots – Intelligent Systems, Cognitive Robotics, and Neuroscience*, ser. LNCS 3575. Springer, 2005, pp. 349–361.

[14] D. Comaniciu and P. Meer, "A robust approach towards feature space analysis," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, 2002.

[15] M. Riedmiller and H. Braun, "A direct adaptive method for faster backpropagation learning: The RPROP algorithm," in *Proc. of the IEEE Intl. Conf. on Neural Networks*, San Francisco, CA, 1993, pp. 586–591. [Online]. Available: citeseer.ist.psu.edu/riedmiller93direct.html

[16] S. Behnke, J. Stückler, M. Schreiber, H. Schulz, M. Böhnert, and K. Meier, "Hierarchical reactive control for a team of humanoid soccer robots," in *Proceedings of 2007 IEEE-RAS International Conference on Humanoid Robots (Humanoids'07), Pittsburgh, PA*, to appear Dec. 2007.