Adaptive Vision for Playing Table Soccer

Thilo Weigel, Dapeng Zhang, Klaus Rechert and Bernhard Nebel

Institut für Informatik
Universität Freiburg
79110 Freiburg, Germany
weigel,zhangd,rechert,nebel@informatik.uni-freiburg.de

Abstract. For real time object recognition and tracking often color-based methods are used. While these methods are very efficient, they usually dependent heavily on lighting conditions. In this paper we present a robust and efficient vision system for the table soccer robot *KiRo*. By exploiting knowledge about invariant characteristics of the table soccer game, the system is able to adapt to changing lighting conditions dynamically and to detect relevant objects on the table within a few milliseconds. We give experimental evidence for the robustness and efficiency of our approach.

1 Introduction

For real-time object recognition and tracking often color-based methods are used. While these methods are very efficient, they are also very sensitive to differing lighting conditions. Adjusting the color classes by hand is usually a tedious task and sometimes unfeasible when the color and the brighntess of the environmental light change over time. For this reason, *robust* vision systems, which either automatically adapt the relevant color classes or which don't rely on color labeling at all, are desirable

In this paper we consider the problem of designing an efficient and robust vision system for the autonomous table soccer robot *KiRo* [1]. KiRo allows humans to play table soccer on a regular table against a machine. It observes the playing field with an overhead camera and controls the four rods of one team according to its observations and pre-defined tactics. In many test games KiRo showed to be a competitive challenge for average human players [1]. The original version of the system used a simple, efficient and straightforward color segmentation method, however, it was very sensitive to changing lighting conditions.

One possible solution for dealing with such conditions is to first extract illumination independent descriptions of an image's surface colors [2]. For the analysis of sports videos, using parallel color spaces and detecting players based on histogram similarities has been proposed [3]. In the context of robotic soccer, many approaches for robust color region finding have been developed. For instance, the work by Jüngel et al. [4] proposes a method for an auto-adjusting vision system using color classes defined by their relative position to a reference class. Dahm et al. [5] describe a color space transformation that is guided by an evolutionary algorithm, and Wyeth et al. [6] describe how to remove the image background based on color histograms.

Inspired by these approaches, we designed two robust methods for detecting the field lines, the players, and the ball. Both methods exploit a number of invariant characteristics of the table soccer setup for guiding the vision process. In particular, for each rod, the location of its axis, the number of corresponding playing figures and the distance between the figures are known. Since we use a yellow ball, it is also known that there is at most one yellow object on the playing surface.

The first method is based on color classifiers which are automatically adapted using a heuristic search in the space of color classifiers. In contrast, the second method doesn't rely on explicit color classes, but rather exploits the contrast beween the objects of interest as an illumination independent descriptor. As the two methods are based on fundamentally different principles, we were particularly interested in a direct comparison between them.

The rest of this paper is structured as follows. In Section 2 we give an overview of KiRo's vision system and describe the invariants that can be expoited. In Section 3 we describe the adaptive color classification method and in Section 4 we present the method based on illumination independent color descriptors. These two approaches are evaluated in Section 5 and we conclude with Section 6.

2 The Vision System

KiRo detects the postions of the ball and the playing figures from color images delivered by a camera overlooking the table. The images are in YUV-format which has the advantage that a pixel's luminance and chrominance values are encoded in separate channels. By processing each half-frame individually, we achieve a frame rate of 50 Hz with an image resolution of 384x288 pixels. Figure 1(a) shows an example of a camera image.

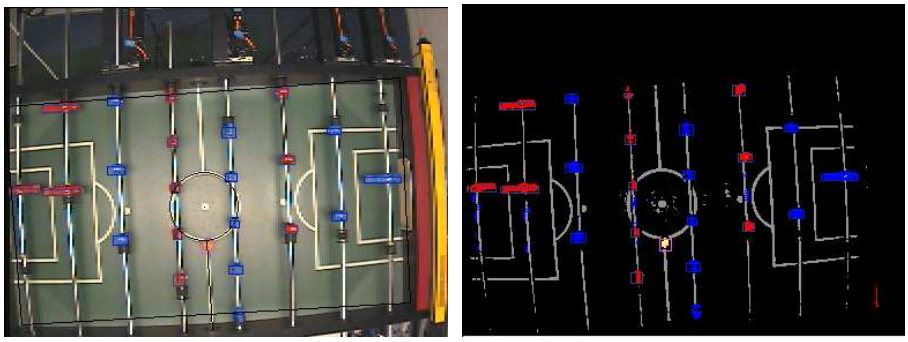
Since the colors of the different players and the yellow ball are easy to distinguish, a straightforward way for detecting the objects is to segment the image regarding predefined color classes and to form regions of pixels which belong to the same class. Each region can then be associated with an object of interest on the table. Using for example the *CMVision* library [7] this can be done efficiently within a few milliseconds.

In order to estimate the real world coordinates of a detected object, the image coordinates have to be transformed to real world coordinates. A suitable transformation matrix can be determined by examining the field lines and finding both the center circle and the center line using a simple template matching method. The circle's center yields the position of the table, the angle of the straight line its orientation and the circle radius the zoom factor [1].

Knowing the table's posture in the camera image allows to mask out the background and to exclusively focus on objects inside the table when detecting the ball and playing figures. Figure 1(b) shows the camera image of Figure 1(a) after calibrating, segmenting and removing the background.

Since the table soccer setup is well defined, a lot of invariant facts can be taken into account for improving the performance of the vision proceess. In particular, the following observations hold throughout a game:

1. There is a total of eleven playing figures of the same color for each team.



(a) (b)

Fig. 1. (a) An original camera image with the black circle and line indicating the result of the calibration step. (b) The same image after segmenting and removing the background.

2. The field lines are fixed and bright white.
3. The ball is the only yellow object inside the playing field.
4. For each rod, the location of its axis, the number of corresponding playing figures and the distance between the figures is known.

By exploiting the first three very general observations, it is possible to automatically find appropriate classifiers for the field lines, the players and the ball. In Section 3 we describe how the classifiers can be obtained by a heuristic search in the space of color bounds.

By utilzing also the fourth, more specialized observation, it is possible to obtain the position of the playing figures without relying on explicit color classes. In Section 4 we describe, how the field lines, the playing figures and the ball can be found based on the contrast to the green playing field.

3 Determining Color Classifiers By Heuristic Search

A color class is a particular subset of the color space. If one is lucky, the class can be approximated by a cube in the color space and can be described by the upper and lower bounds for each channel of the color space. Working on images in the YUV-format we define a color class as

$$\mathcal{C} = \{y_{min}, y_{max}, u_{min}, u_{max}, v_{min}, v_{max}\},$$

and say that a pixel belongs to a color class if its values for y, u and v lie between the respective minima and maxima. As it turns out, for our purposes it is enough to specify just one of these six limits in order to describe the colors relevant for table soccer.

As the y-channel encodes a pixel's luminance, the color class "white" can be defined with all bounds set to their maximum and minimum values, except the lower y-bound:

$$\mathcal{C}_{white} = \{y_{min}, 255, 0, 255, 0, 255\}.$$

The typical setup of the table soccer game features blue and red playing figures. Since high values for u correlate with the blue color and high values for v correlate with the red color, we are able to classify the players sufficiently by also only adjusting one relevant bound while leaving the others fixed:

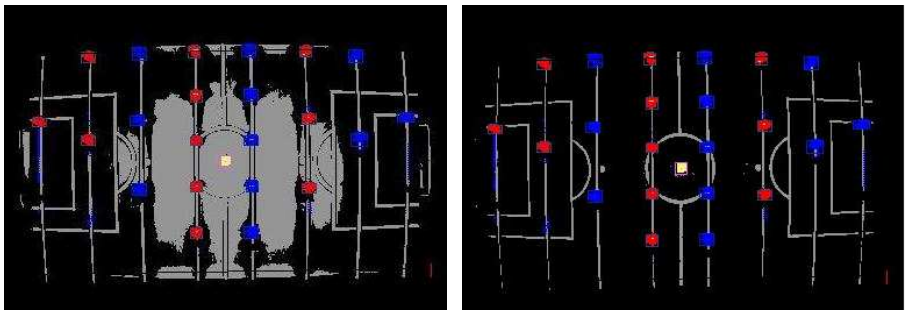
$$\mathcal{C}_{blue} = \{0, 255, u_{min}, 255, 0, 255\}$$
$$\mathcal{C}_{red} = \{0, 255, 0, 255, v_{min}, 255\}.$$

Also the yellow ball can be sufficiently classified by only adjusting one bound, namely the upper u-bound:

$$\mathcal{C}_{yellow} = \{0, 255, 0, u_{max}, 0, 255\}.$$

3.1 Field-line Identification

Figure 2 shows, how changing the lower y-bound affects the classification of "white" pixels.



Y_w = 170 Y_w = 241

Fig. 2. $\mathcal{C}_{white}$ for two different values of y_{min}.

Let N denote the total number of pixels in a camera image and let t_w denote the total number of pixels which are classified as "white" with respect to a particular classifier. Then, the percentage of "white" pixels in the image is given by $p_w = \frac{t_w}{N}$. Now, the important observation is, that if the classifier is adjusted to capture just the field lines and nothing else, p_w is a constant for a fixed N. Therefore, the process of calibrating the field lines can be expressed as the search for the lower y-bound which yields the p_w closest to the known fixed percentage. In other words, as p_w is a function of y_{min}, the appropriate y_{min} for classifying the field-line can be obtained by minimizing

$$f_w(y_{min}) = |p_w(y_{min}) - P_w|, \tag{1}$$

where P_w denotes the constant percentage which can be easily determined empirically.

A simple binary search can be used for minimizing f. Since the field-lines can be covered more or less by the playing figures, the actual percentage of "white" pixels

in the camera image may vary. However, classifying the field lines is usually tolerant against deviations in the exact value for p_w.

As the table's posture in the camera image is not known prior to assessing the transformation matrix (which in turn requires an appropriate classifier), white objects outside the table may render the described method completely unusable. However, such "noise" can usually be filtered out reliably by discarding all "white" regions whose width and height makes them very unlikely to correspond to a field-line.

In order to cope with such conditions, the previously found value for y_{min} is revised by adapting P_w according to the identified noise: With n_w denoting the number of pixels which are considered to be noise, a new P'_w is calculated as $P'_w(y_{min}) = \frac{n_w}{N} + P_w$. Now, we seek to minimize an enhanced version of f_w:

$$f'_w(y'_{min}) = |p_w(y'_{min}) - P'_w(y_{min})|. \tag{2}$$

Again, this can be done by a simple binary search. However, as P'_w is a function of y_{min} we need to repeat the search with P'_w re-calculated according to the newly found y'_{min}. This process is iterated until $P'_w(y_{min})$ converges such that the absolute difference between two consecutive P'_w's is smaller than some ϵ.

The described method can either be run on one camera frame until the calibration is completed or be used in an *anytime-fashion*. While the first is usually appropriate for an initial calibration step, the latter is advantageous for adapting the system to changing lighting conditions during a game where only a limited amount of time is available for the vision process in each control cycle.

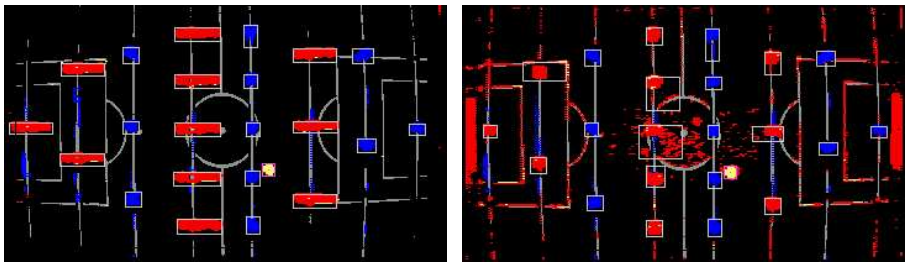
When the calibration is updated in an *anytime-fashion* we usually allow about 5 msec in each cycle for the vision process. Then, the currently considered values for y'_{min} and $P'_w(y_{min})$ are saved for continuing the search with a new frame in the next cycle. As sudden brightness changes between consecutive frames can change the wanted value for y'_{min} considerably, a linear search is used in these cases in order to assure completeness of the search.

3.2 Identifying Blue and Red Players

As the sizes of the player-regions change considerably when a rod is rotated, a fixed percentage value can't be used to guide the search for the appropriate classifiers of the blue and red players. Figure 3 illustrates the effect of applying the same method as for the field lines by using a fixed target percentage. If all the playing figures are positioned horizontally, an appropriate classifier is found. But if the figures are facing downwards, the resulting classifier produces too much noise which will hamper the correct detection of the rod's posture.

However, we know that there is a fixed total of eleven playing figures of the same color for each team and we also know the possible minimum and maximum size of the regions corresponding to a playing figures. Furthermore, it is now possible to examine only image pixels which correspond to the inside of the table since its posture in the camera image has been previously determined using the field lines.

Exploiting this knowledge, we are able to find the optimal blue and red classifier by searching for the lower u- and v-bound which yield the best *noise ratios* $r_b(u_{min})$ and



(a) (b)

Fig. 3. The same target percentage resulting in an appropriate classifier (a) or a too noisy one (b). The bounding boxes indicate the estimated player dimensions.

$r_r(v_{min})$. With N_{table} being the total amount of pixels belonging to the table, the ratios are calculated as

$$r_b(u_{min}) = \frac{n_b(u_{min})}{N_{table}} \quad \text{and} \quad r_r(v_{min}) = \frac{n_r(v_{min})}{N_{table}},$$

where n_b and n_r are the number of "blue" and "red" pixels which are considered as noise. With empirically determined target percentages P_b and P_r, the search for the optimum classifiers can now be described as minimizing the following equations:

$$f_b(u_{min}) = |r_b(u_{min}) - P_b| \tag{3}$$
$$f_r(v_{min}) = |r_r(v_{min}) - P_r|. \tag{4}$$

Of course, the distinction between noise and correctly classified pixels is crucial for this approach. However, with the aid of our domain knowledge, this can be accomplished in a rule-based manner:

- If there are less than eleven regions and no region has dimensions larger than the possible maximum, then there is no noise.
- If there is a region larger than the possible maximum size, then all its pixels are considered to be noise.
- if there are more than eleven regions, all but the eleven biggest ones are considered to be noise.

Considering these "rules", the search can be guided to increase or decrease the lower u- and v-bounds such that f_b and f_r converge to a minimum value.

As for the field lines, a binary search can be used if the calibration is done on one frame only. In general, though, a linear search is applied for being able to interrupt and resume the search arbitrarily.

The current values for u_{min} and v_{min} can now be used for classifying the image. Subsequently, the position of a playing figure is calculated from the center of gravity of its corresponding bounding box and the player's angle is estimated from its size [1].

3.3 Identifying the Ball

Since the ball is the only yellow object on the table, estimating the ball color is very similar to estimating the color of the players. We now seek to minimize the following:

$$f_y(u_{max}) = |r_y(u_{max}) - P_y|, \tag{5}$$

where $r_y(u_{max}) = \frac{n_y(u_{max})}{N_{table}}$ and $n_y(u_{max})$ is the number of pixels which are considered as noise with respect to the yellow classifier.

The decision if a pixel has to be considered as noise is made in a similar way as for the players – with the only difference that there is only one target region now. Interestingly, this approach also copes with situations where the ball is not visible because it is either outside the table or hidden by a playing figure or a rod: As "yellow" noise arises during the search at many places simultaneously, f_y usually converges in such a case to an absolute value greater than the required ϵ. As a consequence, the ball detection can be suspended until an appropriate classifier is found once the ball is back in play.

In order to cope with situations where the ball is partially covered by a rod or a playing figure a more elaborate approach than for the players is used to estimate it's position on the table. At all possible ball locations squares of the expected blob size are examined and the pixels which correspond to the ball, the players or the rods are counted. By selecting the square with the "best" ratio between the number of pixels of the different color classes, the ball's center can usually be estimated reliably even if it is only partially visible [1].

4 Detecting Objects By Contrast

An alternative to region finding using explicit color classes is to detect objects of interest based on the contrast between them. By searching along scan-lines we are able to identify the object positions directly. However, for being able to determine the coordinate transform for both approaches in the same way, we explicitly calculated the lower y-bound of the field lines.

4.1 Field-Line Identification

Currently, the method for determining the coordinate transformation requires the lower y-bound y_{fl}^- of the field lines. As this bound needs to separate the field lines from the background, it can also be considered as the upper y-bound y_r^+ of all the *remaining* color-classes: $y_{fl}^- = y_r^+$.

In the following, a *scan-line* is defined as a horizontal scan along the image which only considers the y-channel of the YUV image data. Figure 4(a) shows a raw scan-line. The peaks usually mark the locations of the field lines and the rods but may also be caused by image noise. Therefore, y_r^+ can be obtained by filtering out these peaks.

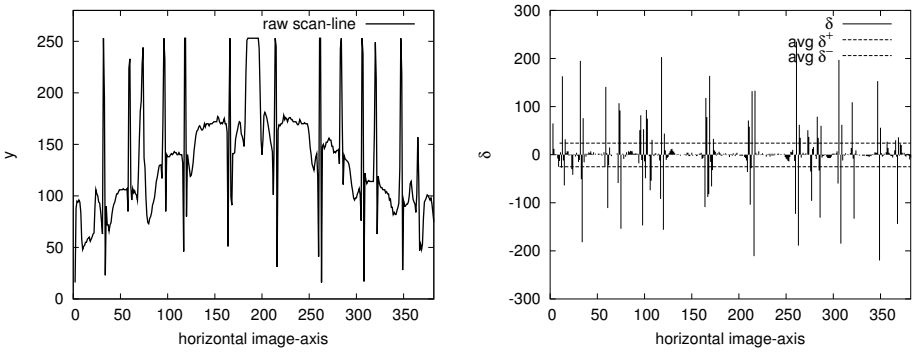
For this, along the scan-line the positive and negative difference between the actual y-value and its predecessor is computed as

$$\delta_i^+ = \begin{cases} \delta_i, \text{if } \delta_i > 0 \\ 0, \text{else} \end{cases} \qquad \delta_i^- = \begin{cases} \delta_i, \text{if } \delta_i < 0 \\ 0, \text{else,} \end{cases} \tag{6}$$

where $\delta_i = y_i - y_{i-1}$ and $i \in [1, n-1]$. By averaging δ_i^+ and δ_i^-, an upper ($\bar{\delta}^+$) and lower ($\bar{\delta}^-$) threshold can be calculated as

$$\bar{\delta}^- = \frac{1}{n^-} \sum_{i=1}^{n-1} \delta_i^- \qquad\qquad \bar{\delta}^+ = \frac{1}{n^+} \sum_{i=1}^{n-1} \delta_i^+, \tag{7}$$

where $n^- = |\{\delta_i | \delta_i < 0\}|$ and $n^+ = |\{\delta_i | \delta_i > 0\}|$. Figure 4(b) shows the calculated δ-values and the two averages.



(a) (b)

Fig. 4. (a) A horizontal scan-line and (b) its corresponding δ-distribution.

After removing y-values with a δ-value above $\bar{\delta}^+$ or below $\bar{\delta}^-$, the scan-line represents the y-color distribution of the *remaining* color-classes and could in principle be used for defining y_{fl}^-. However, the scan-line may still contain plateaus at positions of larger white regions, e. g. at the field's center point. Figure 5(a) shows the filtered scan-line with one remaining plateau.

Plateaus are characterized by a a series of y-values which lie between a sharp ascent and descent of the y-values along the scan line. In order to filter them out, the ascent and descent are detected in a similar way as the peaks were detected and the values between are removed. However, more elaborate δ-values are now used, since $\bar{\delta}^+$ and $\bar{\delta}^-$ are not suited for discriminating between the background and white areas in very bright images. An additional threshold which depends on the general image brightness is computed, and the thresholds for filtering the plateaus are computed from both as:

$$\bar{\delta}'^+ = min(2 * \bar{\delta}^+, \frac{y_{max} - y_{r_{max}}}{2}) \tag{8}$$

$$\bar{\delta}'^- = min(2 * \bar{\delta}^-, \frac{y_{max} - y_{r_{max}}}{2}), \tag{9}$$

where $y_{r_{max}}$ denotes the current maximum y-value along the scan-line which doesn't belong to a plateau.

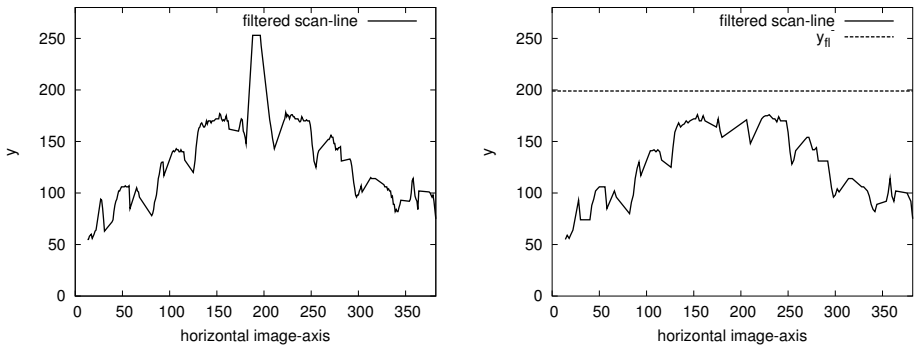


Fig. 5. (a) The scan-line from Figure 4(a) with the peaks removed and and (b) the same scan-line with the plateau removed and resulting y_{fl}^{-}.

After futher removing from the scan-line all y-values with a δ-value above $\bar{\delta}'^{+}$ or below $\bar{\delta}'^{-}$, the upper bound of the non-white color classes can now be obtained as:

$$y_r^+ = \max_{y \in \mathbf{S}} y, \tag{10}$$

where $\mathbf{S}$ is the set of all the y-values along the scan-line which haven't been filtered out in the previous steps. In principle, y_r^+ could be directly used as the lower y-bound for classifying the field-lines. However, in order to achieve a clear and robust separation between the color class for the field-lines and the remaining color classes, we define y_{fl}^{-} by adding an offset as

$$y_{fl}^- = y_r^+ + \epsilon. \tag{11}$$

In relatively dark environments the contrast between white regions and the background (with respect to the lower y-bound) is larger than in brighther environments. Consequently, ϵ should be larger in darker environments. Sine the general brightness level is reflected in y_r^+, we define ϵ depending on y_r^+ as $\epsilon = \frac{y_{max} - y_r^+}{P}$, where $y_{max} = 255$. It turned out that a good value for ϵ is 10% of the distance between y_r^+ and y_{max} ($P = 10$). Figure 5(b) shows the resulting y_{fl}^{-}.

4.2 Identifying Blue and Red Players

For detecting the playing figures we use the knowledge of the fixed axis of their corresponding rods. With the known transformation from world coordinates to image coordinates we compute a scan line for each rod which corresponds to its location in the camera image. The playing figures can then be found by searching for the contrast between them and the background along the scan-lines. However, this search only yields an approximate position and needs to be refined in a second step.

For the search we exploit the fact, that the chrominance values u and v of the playing figures differ significantly while the respective values are usually very similar for the

rods and the green field. In Figure 6 it can be seen how three men cause significant peaks along the scan-line.

As the chrominance values are usually independent from lighting conditions, a linear combination of both chrominance channels is a robust feature to distinguish the playing figures from the rod along the scan-line. Empirically, we determined the linear combinations such that they have a local maximum at the positions of the blue or red player's playing figures:

$$f_{blue}(i) = u_i + |u_i - v_i| \qquad f_{red}(i) = v_i + |u_i - v_i|. \tag{12}$$

Here, i denotes a position along the scan-line. Finding the N playing figures of a rod can now be considered as finding the N maxima of f. In Figure 6 it can be seen, how f_{blue} produces three clearly distinguishable peaks at the playing figure's positons.

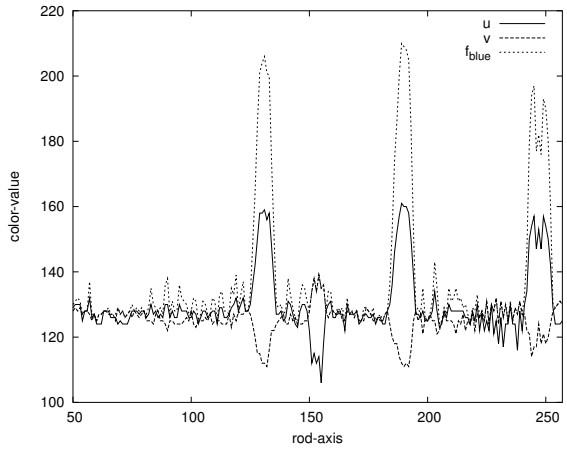


Fig. 6. Scan along a rod with three men

In order to make the search more robust, we exploit the knowledge of the fixed distance D between the rod's playing figures. This allows us to consider in parallel N points on the scan-line being D pixels apart. Thus, we aim at finding the maximum of the following functions:

$$f'_{blue}(i) = \sum_{n=0}^{N-1} f_{blue}(i + n * D) \qquad f'_{red}(i) = \sum_{mn0}^{N-1} f_{red}(i + n * D) \tag{13}$$

The position of the nth playing figure i_n along the scan-line can then be obtained as:

$$i_n = \arg\max_i(f'(i + n * D)). \tag{14}$$

Starting with the estimated man position, the man's exact position can be found by expanding the position to an area equal to the man's width. This is done by an *hill-climbing* algorithm, which finds the man's most likely position: Starting with an initial

interval, which only contains the approximated position, the interval is grown until its length reaches the man width M. The interval is grown by comparing the f-values to the left and the right of the interval and growing the interval towards the direction of the larger value.

For determining the angle of a playing figure, several scans are done parallel to the rod axis around the figure's position. Using the same functions f, these scans check in an interval $[-M, M]$ for the presence of the playing figure. As the scans are moved in both directions away from the rod center, a sudden change in the maximum values of f indicates, that the figure's bounds are reached. Since the playing figure's maximum possible length (when aligned horizontally) is known, the search-scans can be limited to the area around the rod defined by the maximum possible figure's length.

Once a figure's length is known, it's angle can be estimated from this information in a straightforward way [1].

4.3 Identifying the Ball

For detecting the ball, a priori knowledge of its radius and color is utilized. Finding the ball's exact position is done in a similar way as finding the men's positions with a linear combination defined as

$$f_{ball}(i,j) = (255 - u_{ij}) + |u_{ij} - v_{ij}| \tag{15}$$

where i and j now denote coordinates on the table. Since we have to scan the whole field we speed up the search by searching along scan-lines which are the ball's radius apart from each other. This way, first an approximate of the ball's position is found at the global maximum of f_{ball}. Starting at this position, a *hill-climbing* algorithm, similar to the one used for the playing figures, finds the exact area which covers the ball.

5 Results

In order to evaluate the techniques described in the previous sections, we took a log-file consisting of images taken by the camera during a 90 second period. While the log-file was taken, we continuously altered lighting conditions by turning on and off different environmental light sources and changing the camera's shutter speed and gain. We also changed the position and angle of the playing figures and the position of the ball between well known positions. The real world coordinates of these poses were measured by hand and served as the *ground truth* for the evaluation process. While the positions were changed the log file taking was stopped for a moment. Figure 7 shows screen shots of four different stages of the log file.

For each frame of the log file we computed the resulting position and angle estimates of the two dynamic methods and one additional static approach. The static classifier for the players and the ball was adjusted once at the beginning of the log file and tremained fixed during the log file

For a rough comparison of the results, we also calculated color classifiers for the contrast method based on the color distributions at the found object positions. As can

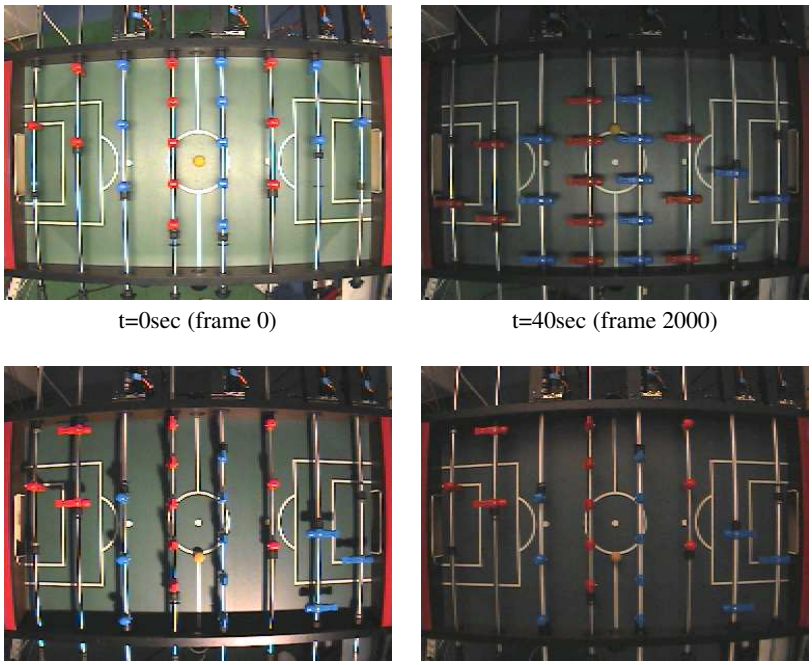


Fig. 7. Camera images from different positions of the log file.

be seen in Figure 8, the two dynamic approaches produced similar color classificators and – unlike the static classifier – reflect the changes in illumination.

More interesting is, of course, how the different features on the field are estimated. The results are plotted in Figures 9 – 11, whereby the deviations are averaged over 20 frames. When a method couldn't perceive a feature, the difference values for the respective object were set to a maximum of 40mm and 50°.

As one can see, all methods produced similar good estimates at the beginning of the log file where lighting conditions were best. Considering, that at the current image resolution one pixel corresponds to 3mm on the table, a position deviation of less than 4mm seems very acceptable. Also the angle estimates are satisfying since usually only the rough orientation of a playing figure needs to be known. Considering, that the angles have to be calculated from the observed length of the playing figures it would certainly be hard to obtain more accurate estimates.

However, when lighting conditions changed and got more difficult, the static classifier lost the object positions frequently while the adaptive methods continued to give accurate position and angle estimates. The heuristic search approach sometimes needed a couple of frames for recovering from drastic brightness changes but turned out to be slightly more accurate than the contrast approach, especially for the ball. The contrast approach, however, showed a clearly better worst case performance.

Figure 12 displays the averaged position and angle devitations with respect to the ground truth. Please note, that the means for the static classifier strongly depend on

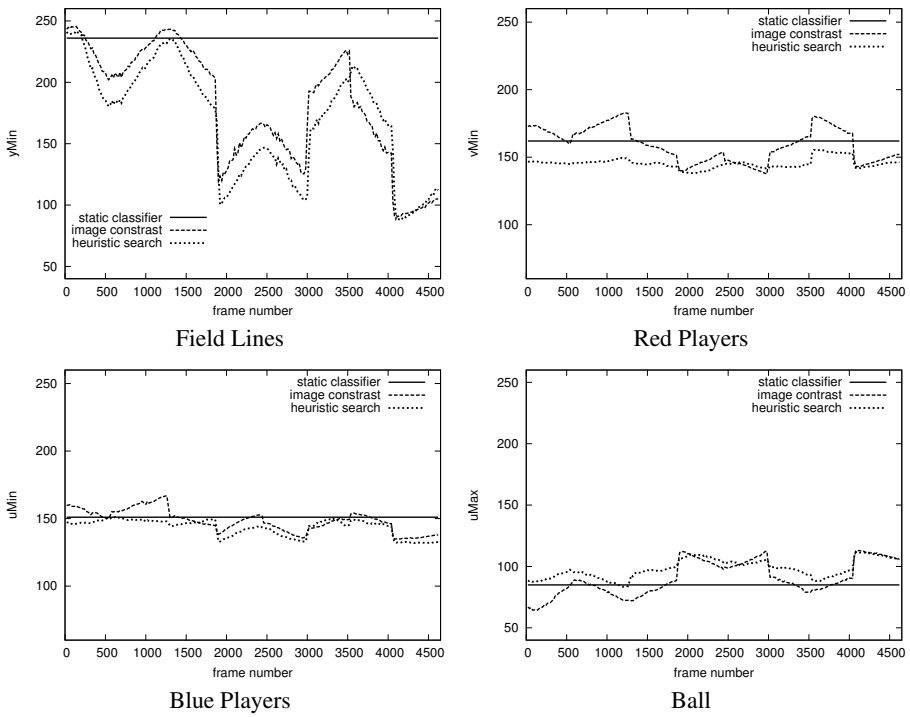


Fig. 8. Change in classifiers for the different methods.

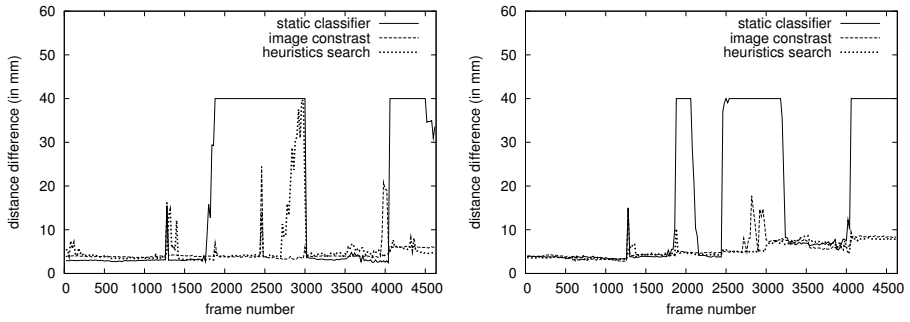
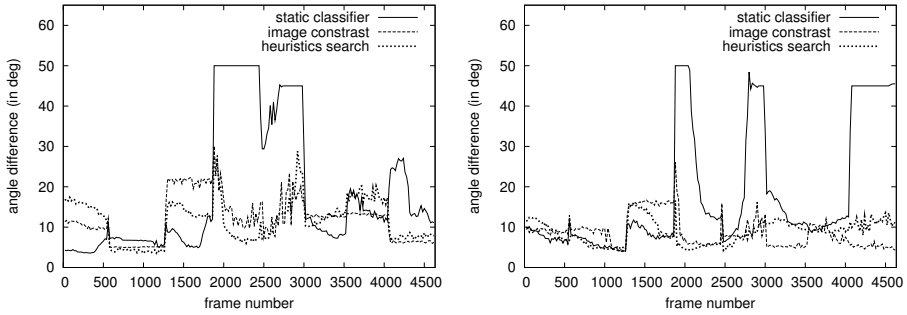


Fig. 9. Deviation of the (a) the red player's positions and (b) the blue player's positions with respect to the ground truth.

the pre-set worst case values for unrecognized objects. Nevertheless, it becomes clear, that the two adaptive approaches are superior to the static variant. Even though the color information in YUV-images is robust against some brightness variations, the two dynamic approaches were far more often capable of adapting to the changed conditions.



(a) (b)

Fig. 10. Deviation of the (a) the red player's angle and (b) the blue player's anle with respect to the ground truth.

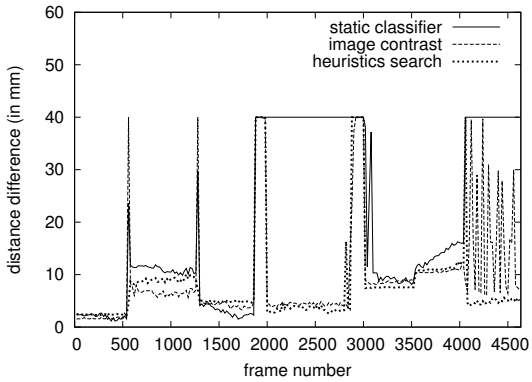


Fig. 11. Deviation of the ball positions with respect to the ground truth.

Interestingly, the performance of two dynamic approaches was very similar and the differences almost not significant.

An important difference between the two dynamic methods are the computational resources required by them. The heuristic search approach may need up to 211 msec (on a 2.6 GHz CPU) to recover the classifiers after illumination changed drastically. However, on average only 12.5 msec are needed for this task. This allows to distribute the adaptive calibration over a series of frames. Limiting the processing time to 5 msec per frame, on average 3 frames are needed for adapting the classifiers. Nevertheless, the approach based on image contrast is far more efficient. It needs less than 2 msec per frame since only a few scan-lines have to be followed and evaluated.

6 Conclusion And Outlook

We presented two methods for efficient and robust object recognition and tracking for the autonomous table soccer robot *KiRo*. Both methods heavily exploit domain knowl-

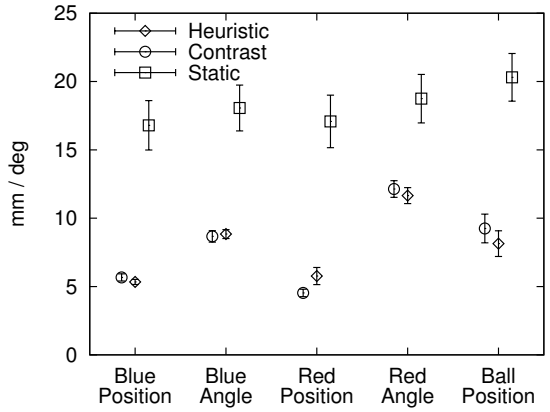


Fig. 12. Averaged position and angle deviations with 95% confidence intervals.

edge. The first method is based on a heuristic search in the space of color classifiers, while the second uses mainly image contrast information and knowledge about the geometry of the table.

Both methods are much better than static color classification and give comparable results. The percentage values required by the heuristic search method are easy to obtain, but the method can be very expensive compuationally. The contrast method is a lot more efficient. However, the functions for discriminating the features may be more difficult to obtain.

In the future, we will evaluate the possibility of learning the percentage values and the color discriminating functions. We will also address the problem of non-uniform lighting conditions.

References

1. Weigel, T., Nebel, B.: KiRo – An Autonomous Table Soccer Player. In: Proc. Int. RoboCup Symposium '02. Springer-Verlag, Fukuoka, Japan (2002) 119 – 127
2. Barnhard, K., Finlayson, G., Funt, B.: Colour Constancy for Scenes with Varying illumination. Computer Vision and Image Understanding (1997) 311–321
3. Ekin, A., Tekalp, A.M., Mehrotra, R.: Automatic Soccer Video Analysis and Summarization. IEEE Transactions on Image Processing (to appear)
4. Jüngel, M., Hoffmann, J., Lötzsch, M.: A Real-Time Auto-Adjusting Vision System for Robotic Soccer. In: Proc. Int. RoboCup Symposium '03. (2004)
5. Dahm, I., Deutsch, S., Hebbel, M., Osterhues, A.: Robust color classification for robot soccer. In: Proc. Int. RoboCup Symposium '03. (2004)
6. Wyeth, G., Brown, B.: Robust Adaptive Vision for Robot Soccer. Mechatronics and Machine Vision in Practice (2000) 41–48
7. Bruce, J., Balch, T., Veloso, M.: Fast and inexpensive color image segmentation for interactive robots. In: Proc. Int. Conf. on Intelligent Robots and Systems (IROS), Takamatsu, Japan (2000) 2061–2066