Robust Color Classification for Robot Soccer

Ingo Dahm, Sebastian Deutsch, Matthias Hebbel, André Osterhues

Computer Engineering Institute
E-mail: {name.familyname}@uni-dortmund.de
University of Dortmund, D-44227 Dortmund, Germany

Abstract. This paper presents an adaptive approach to improve the reliability and performance of color classification. Therefore, we transform the camera-data from YUV into a novel chrominance space. Thereby, an optimized transformation function is given by an evolutionary algorithm. The novel idea is, not to adapt the thresholds that define a specific color region, but to evolve an optimal chrominance space *transformation*. In the novel chrominance space, the color regions are located in easy-to-separate subspaces which reduces the algorithmic complexity of color segmentation and improves classification accuracy significantly.

1 Introduction

The autonomous four-legged robot teams in the SONY league of RoboCup face very special requirements. One the one hand, the camera data is of low quality. On the other hand, the computing power is so low, that an efficient object classification can be done only by color classification. Nevertheless a robust object classification is essential to play robot soccer [1, 2]. Moreover the robot navigation depends only on the quality of extracted sensory information. Therefore, algorithms are needed that give reliable information about the actual state even in fast changing, dynamical environments. Therefore, each legged robot is equipped with a camera and a hardware-based vision processor that provides a robust eight-color differentiation [3]. As main objects are characterized by color [4], a basic object classification can be done by using that module. Alternatively, objects can be classified by software. Then, additional properties of these objects can be involved for a more exact classification [5]. Furthermore, during image processing more relevant information can be extracted [6, 7]. Hence, the hardware color segmentation is used rarely.

Instead, most teams define color-tables for each match. Those color tables depend on the lighting conditions, the robot camera and its settings. Unfortunately, the quality of the camera-data differs between the robots. According to this, a separate color table for each robot grants best performance. On the other hand, this procedure is a time-consuming, inefficient process. If color tables are used, then the robots need constant (i.e. artificial) light. Thus, it is a challenge to develop an efficient way for color segmentation. Furthermore, the YUV chrominance space [8] as provided by the robots internal camera is very light-sensitive.

Therefore, we suggest to transform the sensor-data space into another signal-space, whereby the relevant information can be extracted with lower processing power and higher accuracy at a fully autonomous level.

2 Chrominance-Space-Transformation

The main idea of color segmentation is to partition the chrominance space into subspaces, where each subspace represents exactly one color[9]. In the past, different publications have shown, that the algorithmic performance of color classification depends on the chosen chrominance space [10, 11].

The reason for that is, that the complexity of color segmentation strongly depends on the shape of the subspaces. If the normal vectors of all bounding hyperplanes are parallel to the unit vectors of the chrominance space coordinate system, then the color assignment can be done at highest efficiency. This is illustrated by Figure 1. A region of a specific color in YUV-space (left) is transformed into another signal space (right). In the first case, color segmentation is done by calculating the distance between a pixel in YUV and the bounding hyperplane. In the second case (right), the color can be determined by two threshold-values (T'^{blue}_{min}, T'^{blue}_{max} resp. S'^{blue}_{min}, S'^{blue}_{max}) on each axis of the coordinate system. For that, we define a new chrominance space where all relevant colors can be extracted by using such sets of thresholds.

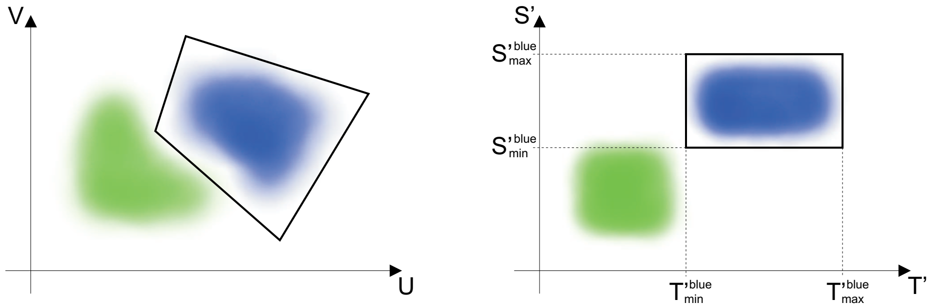

Fig. 1. A two-dimensional space (left) is transformed into another space (right). Thus, the shapes of two clusters (green, blue) change into bars.

The so-called TSL chrominance space is proven to be an efficient choice for fast and robust color segmentation (exemplary in case of skin detection) [12, 11]. Therefore, we use it as starting point for our approach. In case of TSL, the conventional RGB space is transformed as given in [12]. For our purpose, a classification is needed to distinguish between the main nine Robocup colors (green, skyblue, yellow, orange, pink, dark blue, red, black and white) [4]. Unfortunately, the TSL chrominance space mainly depends on the "not blue"-components of the RGB-model. This enhances the quality of color classification for face recognition, as blue components are negligible in skin color. Unfortunately, robot soccer team color as well as landmarks and goals are dark blue and skyblue resp.

Therefore, we suggest to include the "blue"-channel into the calculation of the S-component by changing the original formula. Furthermore, we insert coefficients $(e_0, e_2, ...e_9)$ into the transformation algorithm 1-4 in order to generalize the TSL-approach. To transform YUV into a TSL-like chrominance-space, we initially transform it into an auxiliary space $C_1C_2C_3$. This is expressed by Equation 1.

$$C_n = \frac{e_{(2n+1)} \cdot U + e_{(2n+2)} \cdot V}{e_0 \cdot Y + e_1 U + e_2 V} \; n \in \{1, 2, 3\} \tag{1}$$

$$T' = \frac{255}{\pi} \cdot \begin{cases} atan2(C_1, C_2) + \pi/4 & C_2 > 0 \\ atan2(-C_1, -C_2) + \pi/4 & C_2 < 0 \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

$$S' = 255 \cdot \sqrt{e_9 \cdot (C_1^2 + C_2^2 + C_3^2)} \tag{3}$$

$$L' = Y \tag{4}$$

For the following, we call the resulting chrominance space TSL*, with the components T', S' and L'. In the next section, we show how the coefficients e_i can be adjusted in order to tune color classification quality.

3 Chrominance-Space and Threshold Evolution

A standard solution to optimize parameters in order to solve a specific problem is to use evolutionary algorithms (EA) [13, 14]. Therefore, we use an EA to find an optimal chrominance-space for an efficient color classification. Moreover, we estimate the corresponding thresholds simultaneously. For that purpose, we need an evolution strategy and a fitness function.

3.1 Evolution Strategy

An easy and promising algorithm to optimize a parameter set is the so called "1+1 Evolution Strategy" (OOES):

Assume, $\boldsymbol{p}^k = (p_0, p_1 \ldots, p_N)^T$ is an N-dimensional parameter vector that should be optimized. Using OOES, we initialize $\boldsymbol{p^k}$ with random starting values and add a noise vector $\boldsymbol{r} = (r_0, r_1 \ldots, r_N)^T$, where r_i is a random value given by Gaussian Noise with a variance σ_η^2. A new parameter set – the offspring $\boldsymbol{p}^{(k+1)}$ – is created by adding parent $\boldsymbol{p}^k$ and noise vector $\boldsymbol{r}$:

$$\boldsymbol{p^{(k+1)}} := \boldsymbol{p^k} + \boldsymbol{r} \tag{5}$$

After each addition, we measure the fitness $f(p^{(k+1)}$ of the offspring(see Sec.3.2). If the fitness of the offspring is higher than the parent's fitness, then $p^{(k+1)}$ is taken as new parent, else it is discarded.

As illustrated by Figure 2, it is difficult to find a mutation rate which promises a fast progress rate to the optimal solution. The curves represent solutions of the same fitness. The local optimum surrounded by the curves is marked black. In the left part of Figure 2, a low mutation rate is used: the progress speed is low. On the other hand, the right part of Figure 2 illustrates a high mutation rate: There, the success probability to reach the optimum is lower then in the first case. Rechenberg has shown that in most cases a success probability of 1/5 leads to the fastest progress speed [15].

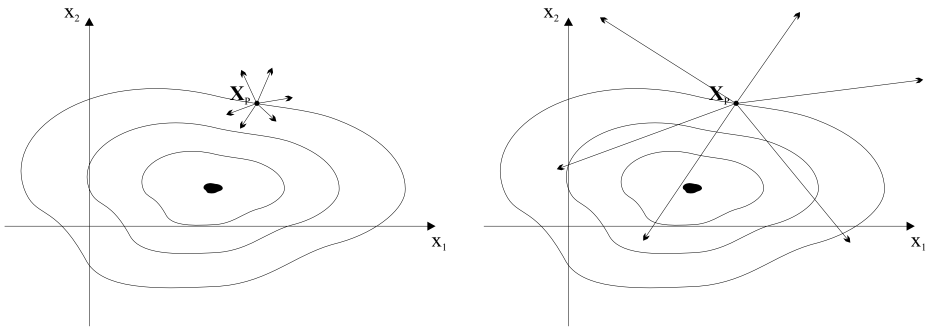

Fig. 2. Possible mutations with low (left) and high mutation strength (right). X_P is an individual in a two-dimensional property-space. The curves represent solutions of the same fitness. The black area is a local optimum.

In our case, we use a high mutation rate at the beginning ($\sigma_\eta^2 = 2.0$). After every 64 evolution steps, we decrease the σ_η^2 downto 85%. As shown in Section 4, this takes to a fast progress speed at a high success probability.

3.2 Fitness Function

To measure the quality of each parameter set, a fitness function is needed. We want to distinguish between the nine most significant colors for robot soccer. For implementation reasons, we differentiate between 10 colors, because we add a class for unclassified color (e.g. gray, purple, or brown).

As the first step, we take reference images for each color. Each image contains mainly the color of a specific class. All of these images are stored in YUV format which is directly taken from the robot's head-camera. Every image i is of width w_i and has a height h_i. Thus, the area of the image is given by $A_i = w_i \cdot h_i$. Object of our EA is to maximize the number of pixels in each image that are classified correctly. For that, let $target(x, y)$ be the target color, that we like to get in the optimal case. Furthermore, let $img(x, y)$ be a function that assigns the color of a pixel at (x, y) after transforming and quantizing the picture. This is illustrated by the following example:

Fig. 3. Reference images for each color (left). Color classification by clustering to the 16 most popular image colors (middle). Handmade target images (right).

Example. *Assume, the YUV-picture holds a value (y,u,v) at position (x,y). That YUV-tuple values are transformed into a TSL*-color tuple c=(t',s',l') after Eq. 1-4. Afterwards, we check the position of c in the TSL*-space (comp. Fig. 5) to assign c to a color class img(x,y).*

According to this, we define the fitness function as the ratio of correctly classified pixels to misclassified ones. To reduce the misclassification ratio, we introduce a color-class "unknown" which is chosen for each pixel that cannot assigned to one of the basic colors[1].

The fitness estimation can be customized by using larger images for more important color classes (e.g. colors of ball, opponents, goal) and smaller images for less important colors. For the following, we decided to handle all colors equally. Therefore, the fitness function f must be adapted as given by Equation 6 to reflect the different areas of the reference images.

$$f := \sum_{i=0}^{\forall i} \left(\frac{1}{h_i \cdot w_i} \right) \sum_{x=0}^{x<w_i} \sum_{y=0}^{y<h_i} \begin{cases} 1, \text{if } \texttt{img}(x,y) = \texttt{target}(x,y) \\ 0, \text{otherwise} \end{cases} \tag{6}$$

4 Simulation Results

The creation and selection of reference images is a time-consuming process: Images must be taken by the robots' cameras, the parts of similar color must be selected, cropped and saved. This takes five to ten minutes for a practiced user. On the other hand, the chosen EA provides an excellent performance. Typically a fitness of more than 0.95 is reached after less than 10 evolution steps, whereby

[1] Note, that no reference image for "unknown color" is depicted in Fig. 3 nor 5.

each steps takes only some milliseconds. Thus, a "just-in-time" adaption of the chrominance space is generally possible.

YUV-to-TSL*-Transformation							
$e_0 =$	4.253156	$e_3 =$	-0.051191	$e_5 =$	0.004342	$e_7 =$	0.133934
$e_1 =$	3.721591	$e_4 =$	0.186359	$e_6 =$	0.040280	$e_8 =$	0.162277
$e_2 =$	4.260.283					$e_9 =$	2.603124

Table 1. Parameter set to transform YUV efficiently into TSL* after Equation 1-4.

After approximately 25.000 steps (approx. 5 to 10 minutes - depending on the used CPU), the mutation ratio reaches the lower bound of accuracy of the cpu floating point calculation unit. Exemplary, using our reference images, a fitness of 0.9895 can be reached by applying a signal-space transformation after Equations 1-4 using the parameter set given by Table 1.

As illustrated by Figure 4, the fitness of the color classification remains at a high level after a few evolution steps. Nevertheless, it raises slowly and monotonously with the duration of the evolution. Simultaneously, the mutation ratio is decreased by 15% after every 64 steps.

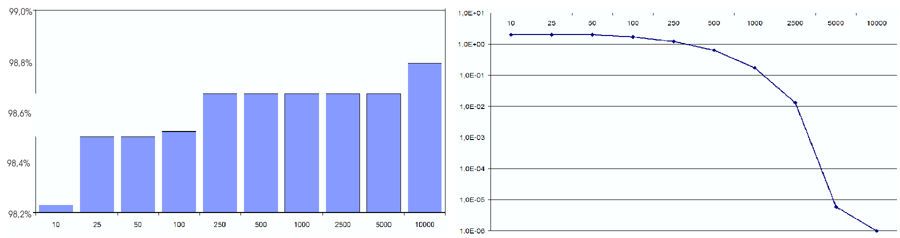

Fig. 4. Individual fitness vs. duration of evolution (left). Mutation ratio σ_η^2 vs. duration of evolution (right).

The power of the suggested algorithm is visualized by Figure 5. There, the reference images are compared with the transformed and quantized pictures and the handmade target images.

After porting the chrominance-space transformation to the robots, we observed a significantly improved color segmentation compared to our old YUV-space segmentation tool [16]. This is illustrated by Figure 6. There, three images taken from the robots' internal head-camera are color-classified using a conventional YUV-approach and the threshold approach after transforming the picture into an optimized TSL*-chrominance space.

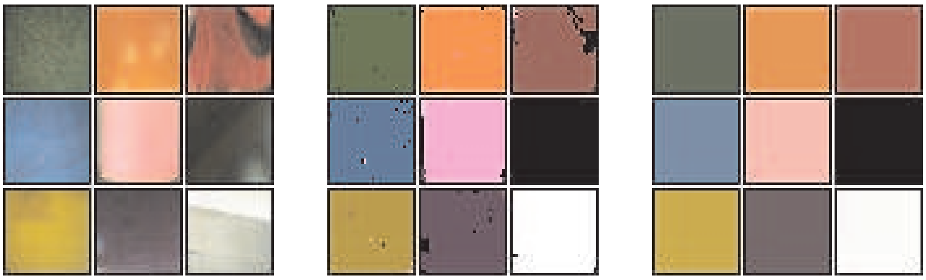

Fig. 5. Reference images for each color (left). Color classification after transforming into TSL*-space (middle). Handmade target images (right).

The TSL* chroma-space based color-segmentation is more robust against luminance variation. This is illustrated by the right part of Fig. 6, where a white wall is shown in the back: Due to suboptimal lighting conditions, the luminance varies and objects drop shadows on the wall. In the TSL*-space, about 50% more pixels are correctly classified as "white" than in the YUV-case. The playing field is illuminated more homogenously than the wall. Thus the robustness against luminance variation is not so clearly observable objects on the playing field. Anyhow, there are some green pixels in the carpet area, that are misclassified in the YUV-space (middle) and blue pixels that are misclassified, too (left).

Another important detail of color-segmentation is the performance of code. On the one hand, the camera picture must be transformed into TSL* which takes additional time. On the other hand, the threshold-based color segmentation is faster than the conventional approach. Thus, the overall consumed time depends mainly on the CPU speed and its provided features. We observed that TSL*-performs faster than YUV-based color-segmentation on a SONY super-core and vice versa on the standard core.

5 Conclusion

In this paper, we presented a method to increase the quality and performance of color classification by transforming the YUV chrominance space into a novel chrominance space, called TSL*. Compared to a conventional color classification in YUV-space, the color segmentation in TSL* is of significantly higher accuracy at comparable computation complexity. This is achieved by a simple EA that optimizes the TSL* chrominance space parameters automatically to consider the current lighting condition. This generic approach can be adapted easily to improve color classification in all Robocup leagues. Further investigations will show, if and how manual reference image preparation can be replaced by a fast automatic reference image generation.

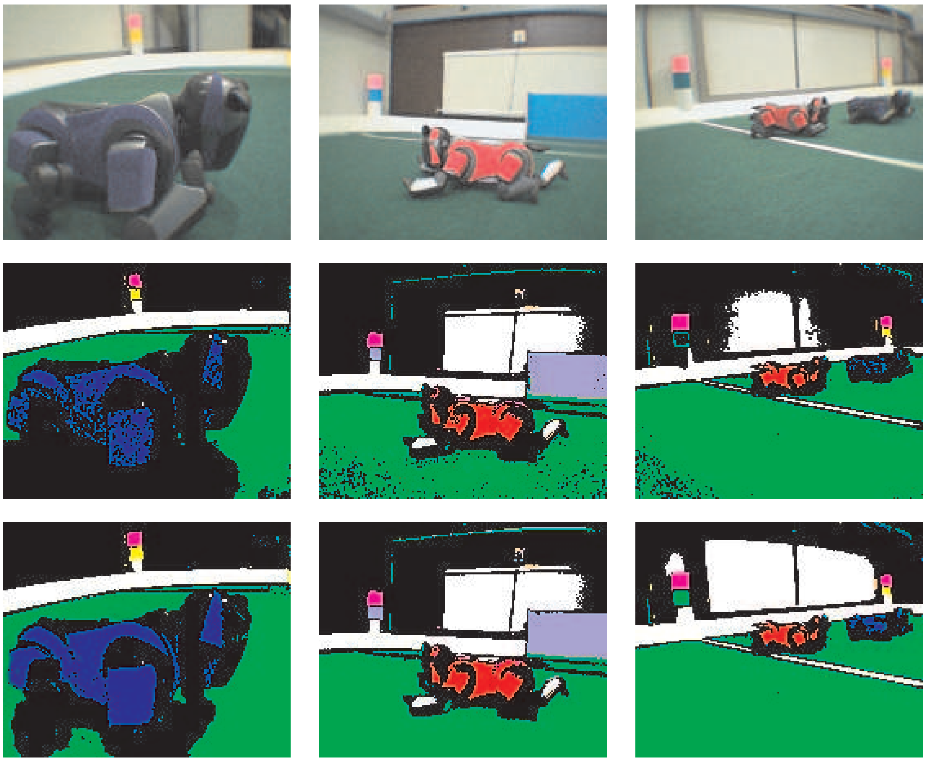

Fig. 6. Color classified images as taken from a robot camera (top), classified using conventional color segmentation in YUV (middle) and the TSL*-classification algorithm (bottom).

Acknowledgments

This work was supported by Deutsche Forschungsgemeinschaft (DFG), Microsoft Academic Alliance, Microsoft Hellhounds and German Team. Thanks to Jens Ziegler for his valuable ideas and support.

References

1. T. Weigel, W. Auerbach, M. Dietl, B. Dumler, J.-S. Gutmann, K. Marko, K. Muller, B. Nebel, B. Szerbakowski, and M. Thiel, "CS freiburg: Doing the right thing in a group," in *RoboCup*, pp. 52–63, 2000.
2. J. Bruce and K. Chaudhuri, "Automatic thresholding for realtime color vision: Proposal," 2000.
3. M. Veloso, W. Uther, M. Fujita, M. Asada, and H. Kitano, "Playing soccer with legged robots," in *Intl. Conference on Intelligent Robots and Systems IEEE/RSJ*, pp. 437–442, October 1998.

4. SONY, *Sony Four Legged Robot Football League Rule Book*. SONY, 2nd ed., 2000.
5. I. Dahm and J. Ziegler, "Adaptive methods to improve self-localization in robot soccer," in *RoboCup Symposium Fukuoka*, 2002.
6. P. Jonker, J. Caarls, and W. Bokhove, "Fast and accurate robot vision for vision based motion," *Lecture Notes in Computer Science*, vol. 2019, pp. 149–??, 2001.
7. W. Skarbek and A. Koschan, "Colour image segmentation – a survey –," tech. rep., Institute for Technical Informatics, Technical University of Berlin, October 1994.
8. C. Poynton, *A technical introduction to digital video*. Toronto: Prentice Hall, 1996.
9. J. Bruce, T. Balch, and M. M. Veloso, "Fast and inexpensive color image segmentation for interactive robots," in *Proceedings of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '00)*, vol. 3, pp. 2061 – 2066, October 2000.
10. M. C. Shin, K. I. Chang, and L. V. Tsap, "Does colorspace transformation make any difference on skin detection?," in *IEEE Workshop on Applications of Computer Vision*, 2002.
11. J.-C. Terrillon, H. Fukamachi, S. Akamatsu, and M. N. Shirazi, "Comparative performance of different skin chrominance models and chrominance spaces for the automatic detection of human faces in color images," in *Fourth IEEE International Conference on Automatic Face and Gesture Recognition*, p. 54ff., 2000.
12. J.-C. Terrillon and S. Akamatsu, "Comparative performance of different chrominance spaces for color segmentation and detection of human faces in complex scene images."
13. J. R. Koza, *Genetic Programming*. Cambridge, MA: MIT Press, 1992.
14. W. Banzhaf, P. Nordin, R. E. Keller, and F. D. Francone, *Genetic Programming – An Introduction; On the Automatic Evolution of Computer Programs and its Applications*. Morgan Kaufmann, dpunkt.verlag, 1998.
15. I. Rechenberg, *Evolutionsstrategie '94 (in German)*. No. ISBN 3-772-81642-8, Frommann–Holzboog Verlag, 1st ed., 1994.
16. U. Dueffert, M. Juengel, M. Loetzsch, R. Brunn, M. Kallnik, N. Kuntze, M. Kunz, S. Petters, M. Risler, N. Koschmieder, T. Laue, T. Roefer, K. Spiess, A. Cesarz, I. Dahm, M. Hebbel, and J. Ziegler, "German Team 2002," in *RoboCup 2002*, Lecture Notes in Artificial Intelligence. Springer, 2002.