Robust Color Segmentation Through Adaptive Color Distribution Transformation

Luca Iocchi

Dipartimento di Informatica e Sistemistica
Università di Roma "La Sapienza"
Via Salaria 113 00198 Rome Italy
iocchi@dis.uniroma1.it

Abstract. Color segmentation is typically the first step of vision processing for a robot operating in a color-coded environment, such as RoboCup soccer, and many object recognition modules rely on that.

Although many approaches to color segmentation have been proposed, in the official games of the RoboCup Four Legged League manual calibration is still preferred by most of the teams. In this paper we present a method for color segmentation that is based on an adaptive transformation of the color distribution of the image: the transformation is dynamically computed depending on the current image (i.e., it adapts to condition changes) and then it is used for color segmentation with static thresholds. The method requires the setting of only a few parameters and has been proved to be very robust to noise and light variations, allowing for setting parameters only once when arriving at a competition site.

The approach has been implemented on AIBO robots, extensively tested in our laboratory, and successfully experimented in the some of the games of the Four Legged League in RoboCup 2005.

1 Introduction

RoboCup soccer is a color-coded environment, where colors are used to define principal objects needed for the robots to perform their robot tasks. Recognition and positioning of colored beacons and goals in the field are used for self-localization and reactive behaviors, while the recognition of the orange ball feeds behaviors and coordination tasks.

Consequently, color segmentation is typically the first step of the vision system of a robot playing RoboCup soccer. Since good color segmentation allows for easy implementation of object recognition and localization, most of the robot vision systems are based on fast and accurate implementation of such process. Conversely, it is also possible to recognize and locate objects from a rough segmentation (e.g., [3]), applying more sophisticated recognition techniques (e.g., region growing) at a later stage. However, this second approach may be less reliable or require more computational resources.

Many approaches to color segmentation have been proposed in the RoboCup soccer scenario. Some of these approaches can be implemented in real-time only on robots with adequate computational resources (e.g., Middle-size robots [1]).

G. Lakemeyer et al. (Eds.): RoboCup 2006, LNAI 4434, pp. 287–295, 2007.

An effective implementation of on-line color segmentation on AIBO robots has been reported in [2]. The paper presents an adaptive non-parametric method that computes color classes by representing them as cuboids in the YUV color space with different precision layers. Experimental results of this approach show good computational performance (about 5 fps). However, manual calibration is still preferred during the games [4], since it provides for fast and accurate results, accepting the drawback of time consuming manual setting that is often repeated several time (e.g., just before each game) and for each robot (since color cameras have different response on different robots).

In this paper we present an approach to color segmentation that has been implemented on AIBO robots and used for RoboCup soccer in the Four Legged League. The method performs an adaptive transformation of the color distribution of the image, that is dynamically computed during robot task and used for segmentation. The approach integrates the following advantages: 1) it computes a dynamic transformation providing for robustness to noise and adaptivity to variable light conditions; 2) it uses static thresholds for fast segmentation; 3) it does not require time consuming manual calibration (only a few parameters must be set when arriving in a new location).

An effective implementation can be obtained by computing the transformation function periodically, for example every 25 frames (about 1 second), and when the robot is not in a critical phase of the game (e.g., about to approach the ball). In such steps results of color classification are stored in a lookup table (or color table), that is used for classification of the subsequent frames. In this way we reach the maximum performance of the image processing module (20-24 fps) for most of the frames (when such transformation is not computed), and periodically a lower performance (currently, slightly less than 100 ms) when this transformation is computed.

Experimental results show the effectiveness of the proposed method: a large data set of labeled images has been used to evaluate the approach in different locations and conditions. Furthermore, we have experimented the method during some of the games in RoboCup 2005. In that case we have set parameters once when we arrived at the competition site and then we used these parameters for some of the games and for the variable light challenge (with the very same code and settings), noticing no difference in the overall behavior of the team with respect to matches in which we have used manual calibration.

2 Color Distribution Transformation

The approach to color segmentation proposed in this paper is based on an adaptive transformation of the color distribution of an image and subsequent static thresholding. The aim of this approach is to integrate the robustness to noise and light changes typical of dynamic methods with efficiency of static ones.

Let us denote an image as $\mathcal{I}$, and each pixel of an image as $i \in \mathcal{I}$. A color space $\mathcal{C}$ is a color representation of an image as captured by a color camera (e.g., RGB, YUV, HSV). We will use the notation $c(i)$ to represent the color of pixel i in a given color space.

Color classes can be seen as a partition of the color space $\mathcal{C}$, $\mathcal{CC} = \{\mathcal{C}_1, ..., \mathcal{C}_n\}$, such that $\mathcal{C}_k \subset \mathcal{C}$ and $k_1 \neq k_2 \Rightarrow \mathcal{C}_{k_1} \cap \mathcal{C}_{k_2} = \emptyset$. It is convenient to consider a complete coverage for the color classes, i.e. $\mathcal{C} = \cup_{k=1..n} \mathcal{C}_k$, possibly defining a special color class including undefined (or uninteresting) colors. We also define the color distribution of an image $\mathcal{I}$ in the color space $\mathcal{C}$ by $D_{\mathcal{I}}(\gamma)$ as the number of pixels with color $\gamma \in \mathcal{C}$.

For a formal definition of our method we define a transformation function $\tau : \mathcal{C} \rightarrow \mathcal{C}$, and we call τ-*distribution* the new color distribution obtained by applying the function τ to the color distribution of an image, and τ-*transformation* the operation of computing the τ-*distribution* of an image. Having τ, color classification is obtained by assigning to each pixel in the image one class within the predefined set of color classes

$$f_{\tau,\mathcal{CC}}(i) = k \quad \texttt{such that} \quad \tau(c(i)) \in \mathcal{C}_k \tag{1}$$

The objective of the transformation $\tau(\cdot)$ is to modify the color distribution of an image in such a way that the subsequent use of fixed thresholds can be effective for color segmentation, i.e., both robust to noise and light variations and efficient. In particular, we would like that the new color distribution do not vary too much in presence of light variations, specially in proximity of the thresholds. To obtain an adaptive method, τ is periodically computed from the current images the robot acquires.

In this paper we refer to the color component H of the color space HSV, because it is a mono-dimensional space that allows for distinguishing most of the colors of interest in RoboCup. The transformation $\tau(\cdot)$ is computed by the following algorithm, where the color space H if discretized to 360 integer values (i.e., $H = [0, 360)$).

Algorithm 1. Color distribution transformation

Input: Image $\mathcal{I}$, window size δ
Output: Function $\tau(h)$, $h \in H$ (explicit representation though a vector $\tau[0:359]$)

for each $\gamma \in [0:359]$ set $\tau(\gamma) = \gamma$
compute color distribution $\mathcal{D}_{\mathcal{I}}(\gamma)$
for $\gamma = \delta/2$ to $360 - \delta/2$ **do**
  $\tau[\gamma] = \mathbf{argmax}_{\lambda \in [\gamma - \delta/2, \gamma + \delta/2)} \{\mathcal{D}_{\mathcal{I}}(\lambda)\}$
end for
while $\tau[\gamma] \neq \gamma \wedge \tau[\gamma] \neq \tau[\tau[\gamma]]$ **do**
  $\tau[\gamma] = \tau[\tau[\gamma]]$
end while

The function τ is first initialized to the identity function, i.e. $\tau(\gamma) = \gamma$, then two steps are performed: the first step assigns to $\tau[\gamma]$ the value $\gamma^* \in [\gamma - \delta/2, \gamma + \delta/2]$ for which $\mathcal{D}_{\mathcal{I}}(\gamma^*)$ is maximum in such interval; the second step performs a transitive closure of the τ function. The value δ is used to limit the interval in which the maximum value is searched. This value is set in such a way to process

a significant interval of values in the color space: smaller values of δ may fail to accumulate colors over a single component (thus producing more peaks), larger values may collapse different colors in a single peak. By empirical tests, we have determined that a value around 10 provides good results in our implementation.

An example is shown in Figure 1. Figure 1b) shows the original distribution (lighter color), the τ-distribution (darker color) in the H color space and three values for the orange-yellow threshold (vertical lines), respectively 51, 57, and 64. The transformation τ is instead given in Figure 1c). The images in Figure 1d) are the results of segmentation with such different values for the orange-yellow threshold, by using τ-distribution (left side) and original distribution (right side). It is possible to see that the segmentation obtained with τ-distribution is more robust since it essentially returns the same correct segmentation for all the values of the threshold, while the segmentation obtained with the original distribution produces some bad classification (orange is classified as yellow when the threshold is lower, yellow is classified as orange when the threshold is higher). This example shows that segmentation on the τ-distribution is more robust. In fact, any value for the orange-yellow threshold between 51 and 64 are good.

The approach presented in this section can also be extended to automatically generate a color table from a set of images, for example by selecting for each value in the color space the color class that has been chosen more often during the sequence. This is useful for off-line automatic calibration or for generation of a first color table to be manually refined.

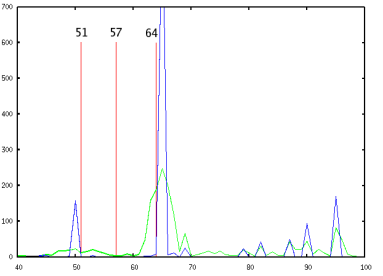
3 Experiments

In order to evaluate the approach presented here we have performed several experiments. In this section we first discuss about performance metrics of color segmentation algorithms, and then present the results of our experiments.

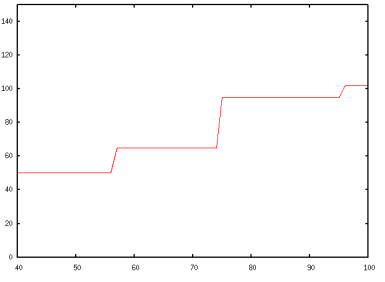
Performance metrics. Defining performance metrics for color segmentation can be relatively easy. For example, we may manually label a set of images by correctly classifying each pixel and then compute classification rate of a segmentation method as the number of correctly classified pixels. Manual labeling can be performed only on a subset of pixels relative to important objects in the scene (see for example [5]). This is a good metric for evaluating color segmentation methods, but it requires a lot of time to generate the ground truth (since several pixels on each image must be manually classified). Moreover, typically object recognition processes that follow the segmentation phase are somewhat robust to small errors in classification, making this measure perhaps too much demanding. In fact, to our knowledge it does not exist a large data set of images labeled in this way, that can be used as a standard benchmark for segmentation algorithms (the one reported in [5] is limited to only a few dozens of images).

In this paper we propose an alternative metric for evaluation of a segmentation algorithm. This metric can be considered as an approximation of the above one, with the advantage of being much easier to be used to produce ground truth,

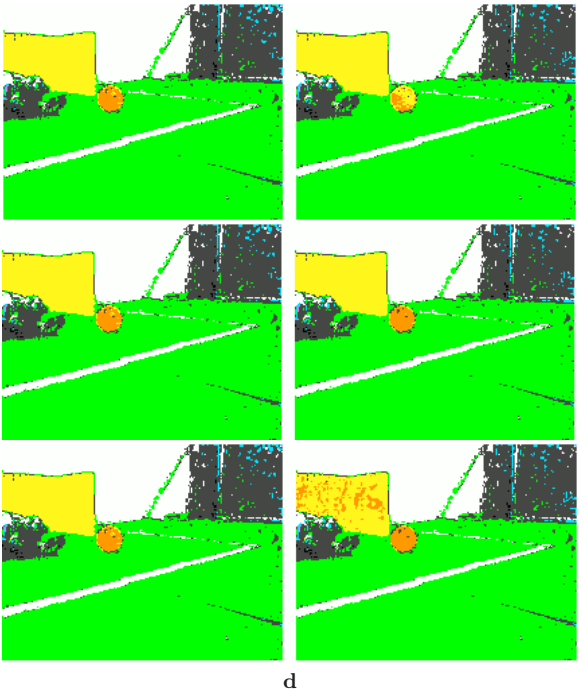
a



b



c



d

Fig. 1. a) Original image; b) Color distributions; c) A portion of $\tau(\cdot)$ function for image a); d) Color segmentation

thus allowing for easily generating larger data sets to be used for comparing different methods.

Instead of labeling each pixel on an image, we define a 4-sided polygon around any object that must be recognized in the environment. For the RoboCup soccer environment we chose to consider ball, goals and beacons. When objects are partially occluded or in case of non-polygonal shapes (e.g., a goal seen closely), we anyway define the best fitting 4-sided polygon around the object. The ball is instead modeled with an ellipse.

From a labeled image and the result of color segmentation on that image, we can measure the percentage of correctly classified pixels within such polygons. Let us denote with $\eta_o(\mathcal{I}_t)$ the percentage of correctly classified pixels of object o at frame $\mathcal{I}_t$. We want to measure the performance of the method over all the frames, i.e. over the entire data set $\{\mathcal{I}_t\}$. A simple choice would be to compute average and standard deviation of this value over t. However, by plotting typical responses of $\eta_o(\mathcal{I}_t)$ over t we see that the outcome is usually multi-modal, thus average and standard deviation seem not to be a good measure for the overall sequence. We found, for example, that in presence of motion blurring the percentage $\eta_o(\mathcal{I}_t)$ for the beacons may be very low. In order to summarize the behavior of the system over time, we propose to compute a distribution over η_o, counting how many times the pixels within the object o are correctly classified with a percentage rate of at least η_o. More specifically, we want to define $V_o(\lambda)$ as the percentage of times in which at least a percentage λ of pixels within the object o have been correctly classified. For example, $V_{ball}(0.5)$ expresses the percentage of frames in which at least half of the pixels of the ball have been correctly classified. $V_o(\lambda)$ $(\lambda \in [0,1])$ is thus defined as

$$V_o(\lambda) = \frac{|\{\mathcal{I}_t | \eta_o(\mathcal{I}_t) \geq \lambda\}|}{|\{\mathcal{I}_t | \mathcal{I}_t \text{ contains object } o\}|}$$

This value can be directly related to the capabilities of an object recognition module. For instance, we have empirically determined that current implementation of our object recognition module allows for correct recognition of objects if at least about 60 % of the pixels are correctly classified.

To evaluate the rate of false positives we have computed the ratio between the number of pixels classified with colors belonging to the objects (i.e., orange, yellow, sky blue and pink) that are outside of the polygons of the objects provided by the groud truth and the total number of pixels in the image.

Experimental evaluation. Evaluation of the proposed method has been performed by computing the functions $V_{ball}(\lambda)$, $V_{goal}(\lambda)$, and $V_{beacon}(\lambda)$ over a number of different data sets, comparing the results of the approach presented in this paper and segmentation obtained by manual definition of a color table. The data sets include at 208x160 pixels resolution and other internal information

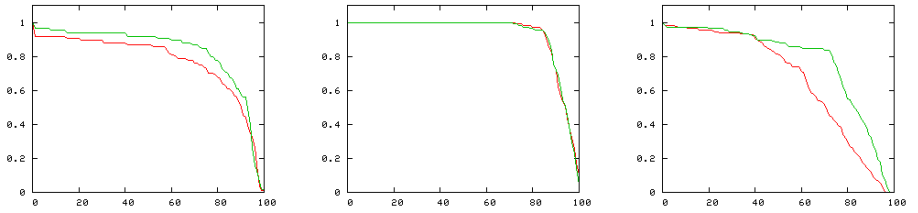


Fig. 2. First data set: $V_{ball}(\lambda)$, $V_{goal}(\lambda)$, $V_{beacon}(\lambda)$

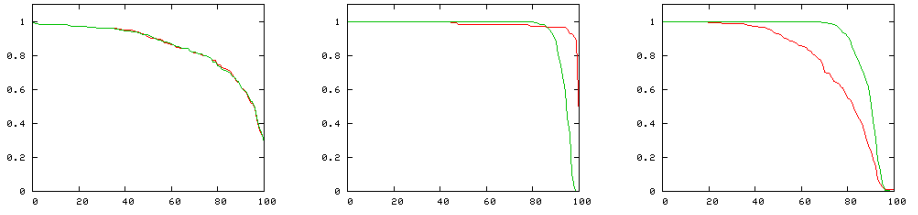


Fig. 3. Second data set: $V_{ball}(\lambda)$, $V_{goal}(\lambda)$, $V_{beacon}(\lambda)$

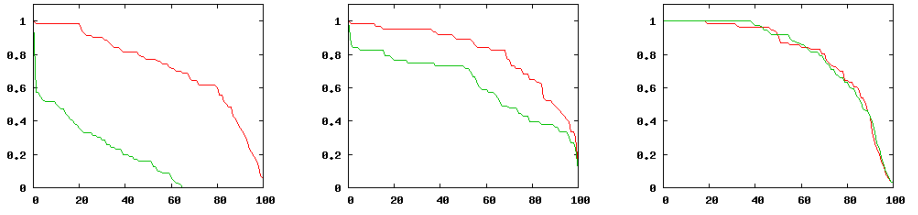


Fig. 4. Osaka data set: $V_{ball}(\lambda)$, $V_{goal}(\lambda)$, $V_{beacon}(\lambda)$

about the robot, they have been been taken from different locations (including Paderborn 2005, Osaka 2005 and our lab) and manually labeled as described above[1]. The first two data sets (containing 395 and 593 frames) used in the experiments reported here are taken from two different locations of our laboratory, using the same elements for the field and the same kind of illumination devices, but in different external light conditions. The third data set (193 frames) has been taken during RoboCup 2005 in Osaka.

For the first data set we have manually created a color table for static segmentation and tuned the parameters of the method proposed in this paper. We then use the same configuration for all the three data sets. For each data set,

[1] The data sets have been used also for evaluating object recognition and localization tasks and are available from `http://www.dis.uniroma1.it/~spqr`.

Table 1. False positives average and maximum rates

Data Set	**Our method** avg/max	**Static segmentation** avg/max
1	0.006 / 0.044	0.010 / 0.057
2	0.013 / 0.062	0.033 / 0.094
3	0.070 / 0.242	0.065 / 0.225

we have thus evaluated the functions $V_o(\lambda)$ for each object (ball, goal, beacon) and the false positive error rates, obtained with the method presented here and with static segmentation.

The results are reported in Figures 2, 3, 4, where red (darker) plot indicates our method, while the green (lighter) plot is the result of static segmentation, and in Table 1. As expected, static segmentation outperforms the dynamic method presented here in the first data sets, where it has been calibrated. However, the graphs also show that our method is very competitive. Also the rate of false positives are similar. In the second data set, that is similar to the first one but in different external conditions, we still have comparable results. However, while there is not a clear distinction in the functions $V_o(\lambda)$ (see Fig. 3), the false positive error rates are larger when using static segmentation. In the third data set, that is quite different from the first one, instead there is a clear difference in the performance (see Fig. 4). Our method provides for higher robustness to different lighting conditions, and to different environments.

These experiments show that the method proposed here is competitive with static segmentation obtained by manual calibration and robust to different light conditions and different environments.

4 Conclusions

In this paper we presented a dynamic color segmentation approach based on adaptive transformation of the color distribution of an image, that is suitable for implementation on robots with low computational resources and effective in presence of noise and illumination changes. The approach has been successfully implemented on AIBOs and extensively experimented in laboratory as well as during official RoboCup games. A new evaluation approach has been proposed and a large data set of labeled images have been created to evaluate and compare different methods.

The present method requires setting only a few parameters usually once arrived at a new location, and then it is robust to different light conditions over a typical competition period. As future work we intend to exploit machine learning techniques for learning the few parameters the method require for realizing a complete non-parametric method.

References

1. Anzani, F., Bosisio, D., Matteucci, M., Sorrenti, D.G.: On-line color calibration in non-stationary environments. In: Bredenfeld, A., Jacoff, A., Noda, I., Takahashi, Y. (eds.) RoboCup 2005. LNCS (LNAI), vol. 4020, Springer, Heidelberg (2006)
2. Jüngel, M.: Using layered color precision for a self-calibrating vision system. In: Nardi, D., Riedmiller, M., Sammut, C., Santos-Victor, J. (eds.) RoboCup 2004. LNCS (LNAI), vol. 3276, Springer, Heidelberg (2005)
3. Lovell, N.: Illumination independent object recognition. In: Bredenfeld, A., Jacoff, A., Noda, I., Takahashi, Y. (eds.) RoboCup 2005. LNCS (LNAI), vol. 4020, Springer, Heidelberg (2006)
4. Nisticò, W., Röfer, T.: Improving percept reliability in the sony four legged league. In: Bredenfeld, A., Jacoff, A., Noda, I., Takahashi, Y. (eds.) RoboCup 2005. LNCS (LNAI), vol. 4020, Springer, Heidelberg (2006)
5. Sridharan, M., Stone, P.: Autonomous color learning on a mobile robot. In: Proc. of AAAI (2005)