Color-Based Detection Robust to Varying Illumination Spectrum

Linderoth, Magnus; Robertsson, Anders; Johansson, Rolf

Published in:
Proc. IEEE Workshop on Robot Vision (WoRV2013), January 16-17, 2013, Clearwater, FL, USA

2013

Color-Based Detection Robust to Varying Illumination Spectrum

Magnus Linderoth, Anders Robertsson and Rolf Johansson
Dept. of Automatic Control, LTH, Lund University, Sweden
magnus.linderoth@control.lth.se *†

Abstract

In color-based detection methods, varying illumination often causes problems, since an object may be perceived to have different colors under different lighting conditions. In the field of color constancy *this is usually handled by estimating the illumination spectrum and accounting for its effect on the perceived color.*

In this paper a method for designing a robust classifier is presented, i.e., instead of estimating and adapting to different lighting conditions, the classifier is made wider to detect a colored object for a given range of lighting conditions. This strategy also naturally handles the case where different parts of an object are illuminated by different light sources at the same time. Only one set of training data per light source has to be collected, and then the detector can handle any combination of the light sources for a large range of illumination intensities.

Index Terms–Color-based detection, Color constancy, Robust classifier, Robot vision, Ball catcher

1. Introduction

This paper presents a method for finding objects based on their color. As an example we will use a robot catching green balls that are thrown toward it. As a first step the color of each individual pixel is used to calculate the probability of that pixel being foreground (green ball) or background.

When color is used for object detection, varying illumination conditions may cause problems, since they change the observed color of the object. This is a well known problem, treated in the field of color constancy [4], which tries to estimate the 'true' color of an object irrespective of the illumination. Humans usually do this very well without realizing it, but there are also numerous examples [9] of how the human vision system can be fooled. Land's retinex algorithm [6] attempts to imitate the human vision system, and does so by finding large color gradients and looking at the relative color between nearby areas.

Another way to achieve color constancy is to look at the observed color of a reference object of known color and adjust the color of all observed images, as done in, *e.g.*, [1]. If the illumination changes, a reference object has to be observed again to adjust the color correction. In [11] the position and color of a face are tracked simultaneously to update the color range of the face as the illumination changes.

The methods described in the previous paragraphs depend on the environment (colors in the background or the availability of a reference color) and are adaptive in the sense that they can adjust their color correction if the illumination varies. The classifier presented in this paper, however, is of the robust type in the sense that a single classifier works for a large range of illumination conditions. The method is illustrated by an example that can handle any combination of daylight and fluorescent light, which covers a large set of the conditions encountered in common indoor environments. The cost of having a single classifier for all lighting conditions is that the color range that is classified as foreground gets bigger, and hence there is an increased risk of false positives, but if the range is chosen wisely the amount of false positives can be made small.

A number of different approaches to detecting balls for ball-catching robots have been presented. In [7, 8] color images were used and the balls were detected by defining ranges in the HSV color space [5] and using the centroid of the detected blobs as ball centers, but none of them mentioned any handling of varying illumination. In [2] grayscale images were used and motion was detected by subtracting a background image from all images. The ball was then detected with the Hough transform. In [3] the cameras were not stationary, so no background subtractions could be used. The balls were detected in high-resolution gray-scale images by using a generalization of the Hough transform.

*This work was partly supported by the Swedish Research Council through the LCCC Linnaeus Center and by the ELLIIT Excellence Center.

†The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013 – Challenge 2 – Cognitive Systems, Interaction, Robotics – under grant agreement No 230902 - ROSETTA. This document reflects only the author's views and the European Community is not liable for any use that may be made of the information contained herein.

2. Chromaticity representation

In this paper chromaticity will be represented by the red and green channels in the normalized RGB space [5]. Given a color (R, G, B), where the three components are the intensities of red, green and blue, the chromaticity is given by

$$r = \frac{R}{R+G+B}, \qquad g = \frac{G}{R+G+B} \tag{1}$$

Now assume that you have two colors (R_1, G_1, B_1) and (R_2, G_2, B_2) with the corresponding chromaticities (r_1, g_1) and (r_2, g_2), and you make a linear combination of the two colors to form a third color:

$$(R_3, G_3, B_3) = \lambda_1(R_1, G_1, B_1) + \lambda_2(R_2, G_2, B_2) \tag{2}$$

with $\lambda_1, \lambda_2 \geq 0$. The resulting chromaticity is

$$\begin{aligned}(r_3, g_3) &= \frac{(R_3, G_3)}{R_3 + G_3 + B_3} \\ &= \frac{\lambda_1(R_1, G_1)}{R_3 + G_3 + B_3} + \frac{\lambda_2(R_2, G_2)}{R_3 + G_3 + B_3} \\ &= \underbrace{\frac{\lambda_1(R_1 + G_1 + B_1)}{R_3 + G_3 + B_3}}_{\theta_1} \underbrace{\frac{(R_1, G_1)}{R_1 + G_1 + B_1}}_{(r_1, g_1)} \\ &+ \underbrace{\frac{\lambda_2(R_2 + G_2 + B_2)}{R_3 + G_3 + B_3}}_{\theta_2} \underbrace{\frac{(R_2, G_2)}{R_2 + G_2 + B_2}}_{(r_2, g_2)} \\ &= \theta_1 \cdot (r_1, g_1) + \theta_2 \cdot (r_2, g_2)\end{aligned} \tag{3}$$

It can easily be verified that $\theta_1, \theta_2 \geq 0$ and $\theta_1 + \theta_2 = 1$, *i. e.*, (r_3, g_3) is a convex combination of (r_1, g_1) and (r_2, g_2). This means that for all possible values of λ_1 and λ_2 in (2), (r_3, g_3) is on the straight line between (r_1, g_1) and (r_2, g_2). This property will be used in the design of the classifier proposed in this paper.

3. Method

3.1. Outline of method

3.1.1 Steps to be done off-line

1. Collect training data with a single light source at a time. Collect many images from each scene and calculate the average to reduce the effect of noise.
2. Subtract bias from the images (the intensity recorded by the camera in complete darkness).
3. Make a histogram over the colors. For each pixel, draw a line between the colors at the different illuminations. The resulting histogram will be interpreted as a relative probability density function for the different colors.
4. Make a 3-dimensional look-up table, where the inputs are the intensities of red, green and blue in a pixel, and the output is the probability density from the histogram in the previous bullet. Add back the bias subtracted in the second bullet.
5. Blur the look-up table to account for sensor noise.
6. Choose the probability density of the background.
7. Compare the probability densities for foreground and background and compute a new look-up table with the probability of a color belonging to the foreground.
8. Calculate $(P(foreground) - 0.5)$ to get a score for each element in the look-up table.

3.1.2 Steps to be done on-line

1. Retrieve the score from the look-up table for each pixel in the image.
2. Find circles that locally maximize the sum of the scores of the enclosed pixels in the probability image.

3.2. Training data collection

One set of images should be collected for each type of light source. For the example described in the introduction this means that two sets of images of the green balls should be collected; one with daylight only, and one with fluorescent light only. If possible, several images should be captured for each light source, to form the average image.

The light intensity returned by an image sensor is the sum of a bias and the actual light intensity. When the histogram is created, it is assumed that all intensities will be linear combinations of the illuminations when only one light source is present. For this to be true, the bias has to be subtracted from the images.

The averaging over many images and bias correction attempt to estimate the actual color distributions of the foreground object (green ball) for the different light sources. The variations stem mainly from differences in color on different parts of the training objects and from varying reflectance in different directions. Measurement artifacts, such as bias and noise, will be handled separately.

3.3. Histogram

A histogram over the colors in the training data is created to estimate the probability density (up to a scale) for different colors. For this histogram a grid of bins is created in the chromaticity plane.

For each pixel in the training data, let the bias-corrected intensities be (R_1, G_1, B_1) and (R_2, G_2, B_2) for the two light sources respectively, and the corresponding chromaticities be (r_1, g_1) and (r_2, g_2). According to the last paragraph in Sec. 2, the observed chromaticity will then be on the straight line between (r_1, g_1) and (r_2, g_2) for all combinations of the two light sources. Hence, all bins in the histogram that are intersected by the line between (r_1, g_1) and (r_2, g_2) are increased by 1.

3.4. Look-up table

In order to minimize the amount of calculations at classification time, as much as possible is calculated off-line and stored in a look-up table. The table takes the intensities of red, green and blue in an image as inputs and returns the probability of that pixel belonging to the foreground.

As a first step in creating the look-up table, the bias is subtracted from the (R, G, B)-intensities of each element and the corresponding probability density is retrieved from the (r, g)-histogram. In this process the probabilities are assumed to be independent of the total intensities $(R+G+B)$. The result will be a look-up table where most elements are close to zero, and the elements with a high probability of belonging to the foreground will form a cone with its tip a the point corresponding to the bias, and the cone gets a larger cross-section area as the total intensity increases.

We now have a table with the probability densities for the actual colors of the foreground objects, but the captured images will be corrupted by noise. Hence, the look-up table is convoluted with a Gaussian kernel that corresponds to the noise variance of the image sensor. This smoothing operation in a neat way handles the fact that the chromaticity is affected more by noise in dark areas than in well illuminated areas. In the parts of the look-up table where the intensities are high, the cone of foreground colors is wide compared to the smoothing kernel and the probability densities will not be affected much by the smoothing operation. However, in the darker parts of the table, near the tip of the cone, the smoothing operation will spread out the probability mass over a much larger volume than before and will result in a lower peak probability density. This reflects the reality well and results in more uncertain classifications in dark areas.

We now have a table with scaled probability densities for observing different colors from a foreground object. To determine the probability P_{fg} of the pixel belonging to the foreground, the probability density p_{fg} of the foreground has to be compared to the probability density p_{bg} of the background:

$$P_{fg} = \frac{p_{fg}}{p_{fg} + p_{bg}} \tag{4}$$

For simplicity, the probability density of the background is assumed to be the same for all colors, and its value is the only tuning parameter of the algorithm.

For the elements in the look-up table where any of the colors is saturated, the probability is set to $P_{fg} = 0.5$, indicating that nothing can be said about that pixel.

Finally, all elements in the look-up table are subtracted by 0.5 to generate scores. This results in a table where positive scores indicate that the color is most likely to belong to the foreground and negative scores indicate that the color is most likely to belong to the background.

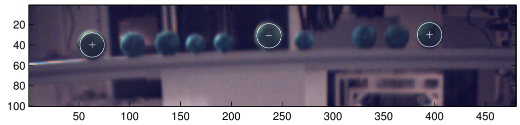

Figure 1. A sample training image with fluorescent light.

3.5. Probability image

The operations described so far can be done off-line during the training phase. The remaining operations have to be done in real-time on the images that should be analyzed. The first operation on the image is to replace the RGB-value of every pixel with the corresponding score from the look-up table.

3.6. Object localization

To find balls in the image, it is searched for circles that locally maximize the sum of the scores for all pixels enclosed by the circle. If the circle is made larger than the optimum, negative scores will be included and reduce the sum. If the circle is made smaller than the optimum, fewer pixels with positive score will be included, and the sum will be reduced.

4. Experimental results

4.1. Cameras

The algorithm described in this paper was experimentally verified on images captured by a Basler A602fc camera. All images used for the experiments were captured with integer intensities in the range $[0, 255]$.

4.2. Training

The bias of the camera was measured to $(5.5, 5.7, 41.4)$ and the noise standard deviation was $(2.1, 1.5, 3.8)$, for the red, green and blue channels respectively. Three green balls were illuminated by fluorescent light only, and 100 images were captured. By calculating the average image the colors could then be calculated with a standard deviation of $(0.21, 0.15, 0.38)$. Similarly 100 images were captured when the balls were illuminated by daylight only. An example image with fluorescent light is shown in Fig. 1. Three green balls, marked by white circles in the figure, were manually picked out and the 1323 pixels enclosed by those circles were used for training. Note that the remaining green balls in the image had a slightly more bluish green color.

Figure 2 shows a scatter plot of the chromaticity for all the pixels in the training data. The sets of points from the different light sources are almost disjoint, which means that if a classifier was trained for only one of the light sources, it

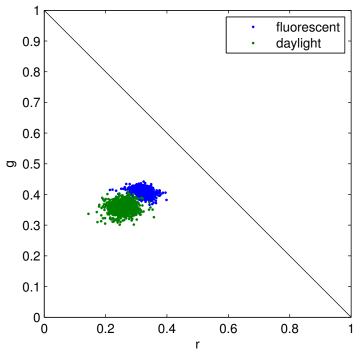

Figure 2. Scatter plot of the chromaticity for the pixels in the training data.

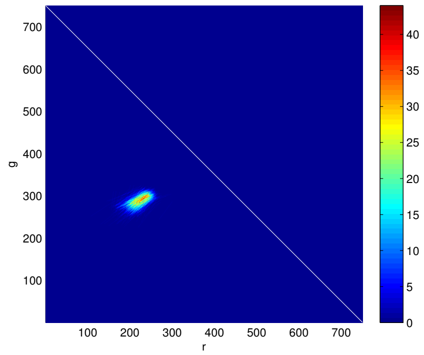

Figure 3. Histogram of foreground colors in the chromaticity space.

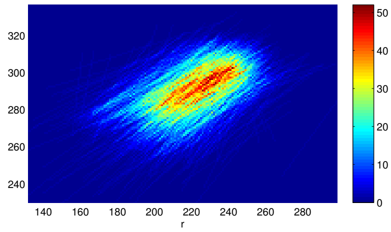

Figure 4. The same histogram as in Fig. 3, zoomed in on the area with the observed colors.

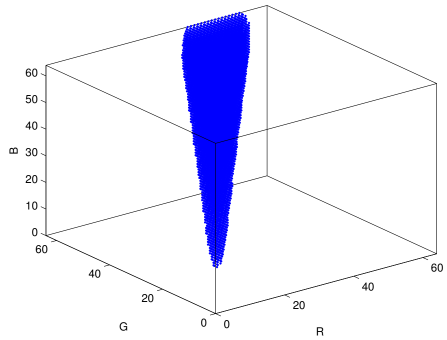

Figure 5. Scatter plot of the elements in the look-up table that had higher probability density than the background, after smoothing.

would work very poorly for the other light source. For the histogram 750×750 bins were used and the result is shown in Figs. 3 and 4. The probability density of the background was chosen so the total probability of the background was 6 times larger than that of the foreground.

In the look-up table 64 intensity levels were used for each color channel, giving a table size of $64^3 = 262144$ elements. Figure 5 shows a scatter plot of the elements that had probability densities higher than the background. The probability densities were transformed to probabilities and subtracted by 0.5 to get the scores. The scores were scaled from the range $[-0.5, 0.5]$ to $[-127, 127]$ and stored in 8-bit signed integers in the look-up table. The resulting size of the look-up table was hence 256 kB, which easily fits inside the L2 cache of modern processors, allowing fast access when the pixels of an image are classified.

4.3. Classification

Images were collected under different illumination conditions to test the performance of the classifier. In Fig. 6 you can see three images captured with fluorescent light, daylight or a combination of both. In the upper part of each image you can see a number of green balls placed on a bar. Three of the balls (those that were used for training) have a slightly more yellowish color than the others. In every image you can also see a ball (of the kind that was used for training) thrown toward the area below the camera. In all images the yellowish green balls have a probability (of belonging to the foreground) that is greater than 0.5 and most other objects have a probability less than 0.5, as intended.

If you look at different areas in the images, you can see that dark areas tend to have probabilities close to 0.5, since the sensor noise makes it hard to determine the chromaticity of the object, while bright areas tend to have probabilities close to 1 if they are green balls and close to 0 otherwise. The pixels where any of the color channels was saturated have the probability 0.5.

Figure 7 illustrates how the range of colors classified as foreground varies with the lighting conditions. In the yellowish fluorescent light the small bluish green ball and the cyan color in the color scale are classified as foreground, since their observed colors are close to that of the yellowish green ball in daylight. In the bluish daylight, however, they are too blue to be classified as foreground.

Balls in the images were detected by finding the maximum of the function described in Sec. 3.6. The circles were here approximated by squares. Only the global maximum is shown in each image, but multiple balls can be detected by looking for several local maxima.

4.4. Ball-catching robot

The algorithm described in this paper was successfully used to catch balls with the robotic ball-catcher described in [7, 10]. The execution time for the entire image analysis process was approximately 5 ms per 656×480-pixel image on a desktop computer with 2.4 GHz processor.

4.5. Validation of assumptions

The method described in this paper assumes that all illuminations are linear combinations of two colors. To investigate whether this well describes the illumination conditions commonly encountered in real life, images of the

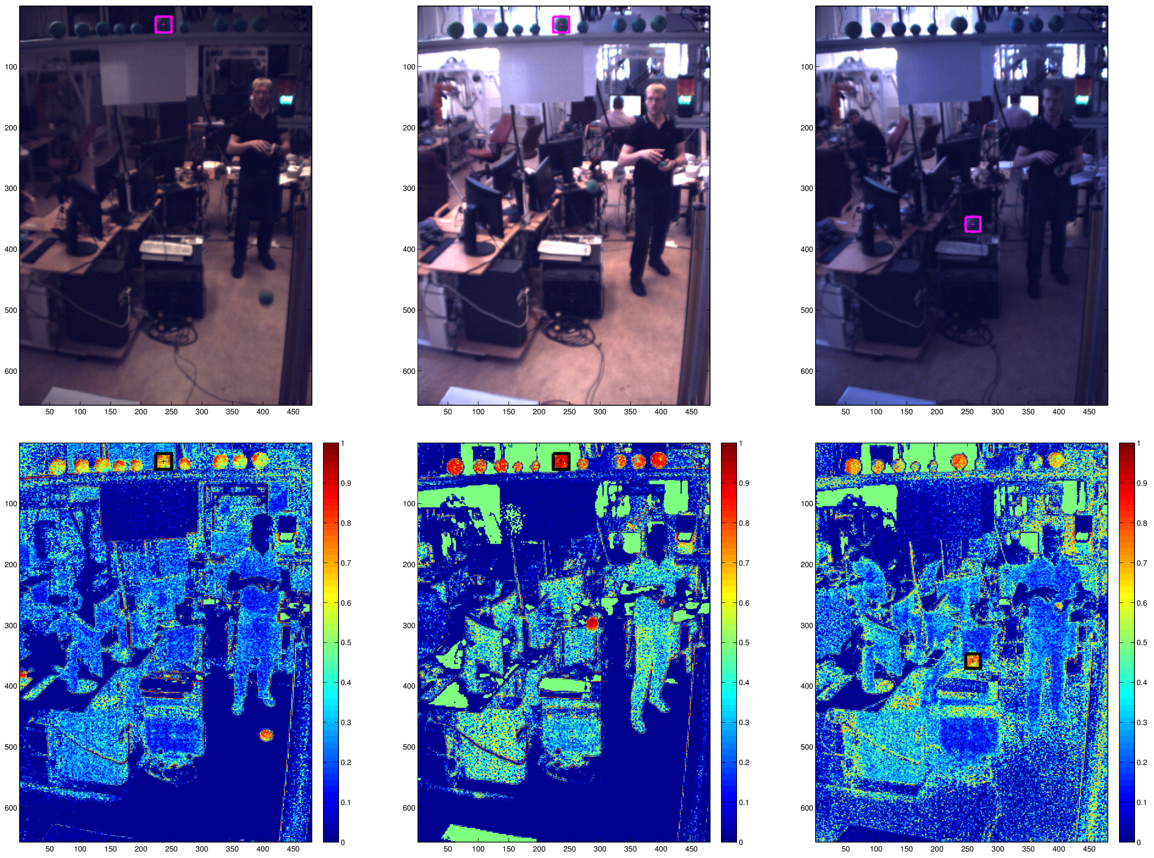

Figure 6. *Top row:* Images that were used to test the classification performance. *Bottom row:* Images showing the probability of a pixel belonging to the foreground. *Left column:* Fluorescent light only. *Middle column:* Fluorescent light and daylight. *Right column:* Daylight only. The ball that maximizes the criterion described in Sec. 3.6 is marked by a magenta/black square in each image.

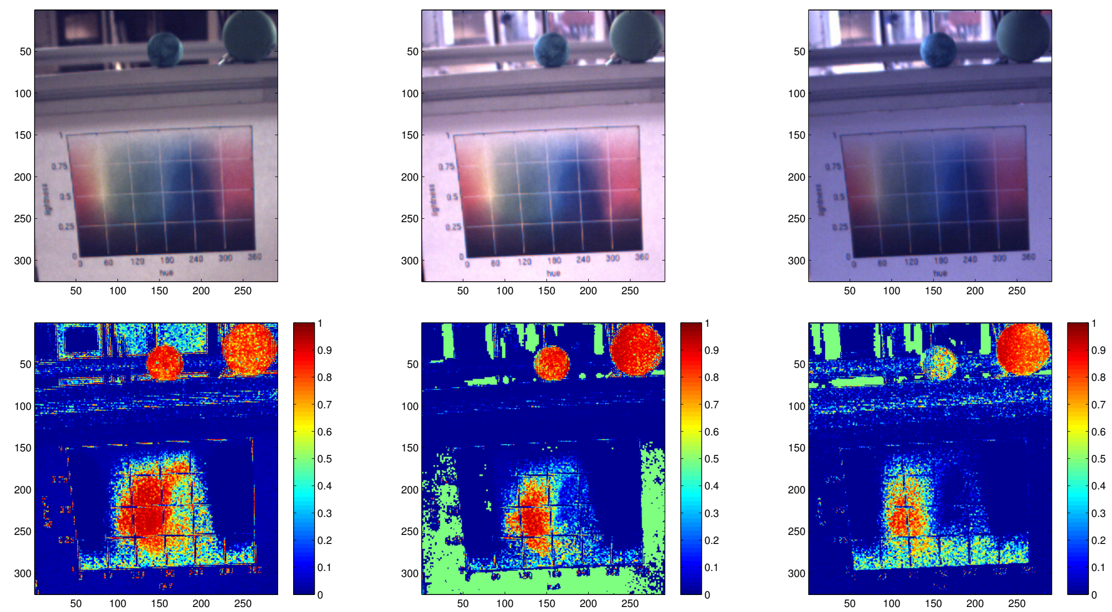

Figure 7. Images of color scales, illustrating the width of the classifier in different illumination conditions. *Left column:* Fluorescent light only. *Middle column:* Fluorescent light and daylight. *Right column:* Daylight only.

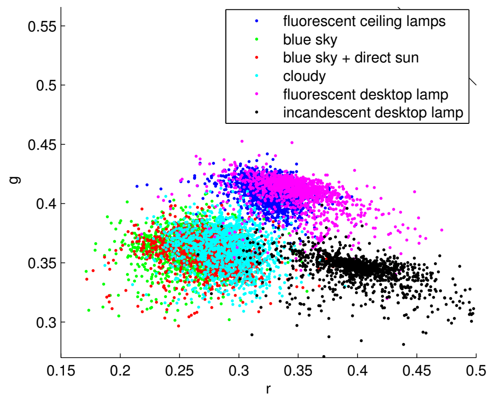

Figure 8. Observed colors of the green balls for different illuminations.

green balls were captured under a number of different illumination conditions. The resulting distributions of the chromaticities are shown in Fig. 8. The images were captured in a lab with windows, but no direct sunlight could hit the balls. The daylight data used in Sec. 4.2 (green in Fig. 8) was captured when the sky was blue but the sun itself was behind clouds. The fluorescent light data used in Sec. 4.2 (blue in Fig. 8) was captured at night with fluorescent ceiling lamps pointing upwards, so most of the light hitting the balls was reflected off the white ceiling.

The additional data sets were captured when it was sunny (red), cloudy (cyan), with a fluorescent desktop lamp shining directly onto the balls (magenta), and with a light bulb shining directly onto the balls (black).

It can be seen that the different kinds of daylight and fluorescent light lie close to a line, as desired, but the light bulb gave colors that are quite different. Hence, Figure 8 indicates that the illuminations encountered in practice really can be described as combinations of sunlight and fluorescent light, as long as light bulbs are not used.

5. Discussion

In this paper the probability density was assumed to be uniform for all colors in the background, which is quite unrealistic. The uniform background distribution was used to show that the method works with such simple assumptions and should work even better with a properly modeled background.

Only a combination of two light sources was considered in this paper. The method could easily be generalized to more light sources. Instead of drawing a line between the data points from the two light sources, one should then draw the polygon forming the convex hull of the data points.

In this paper it is assumed that nothing is known about a pixel if any color channel is saturated, but if, *e.g.*, only the red and blue channels are saturated, you can say with high confidence that the object is not green.

Many different machine learning strategies could have been considered for training the classifier, but if they are applied without a good understanding of the problem it may be hard to collect data that cover all illumination conditions that can be encountered at classification time. With the method described in this paper it was enough to capture data under two different illumination conditions to be able to classify images from a large range of illumination spectrums and a range of illumination intensities that is only limited by the dynamic range of the camera.

6. Conclusions

A method for designing a color classifier that works over a large range of illumination conditions was presented. Only two different illumination conditions were required during the training phase and there is only one tuning parameter. The method was experimentally verified on real data and used to detect balls for a ball-catching robot.

References

[1] C. Balkenius, A. J. Johansson, and A. Balkenius. Color constancy in visual scene perception. *Lund University Cognitive Science, Lund, Sweden*, 2003. ISSN: 1101-8453.

[2] D. Barteit, H. Frank, and F. Kupzog. Accurate prediction of interception positions for catching thrown objects in production systems. *Proc. IEEE Intl. Conf. Industrial Informatics (INDIN), Daejon, Korea*, pages 893–898, July 2008.

[3] O. Birbach, U. Frese, and B. Bäuml. Realtime perception for catching a flying ball with a mobile humanoid. *Proc. IEEE Intl. Conf. Robotics and Automation (ICRA), Shanghai, China*, pages 5955–5962, May 2011.

[4] M. Ebner. *Color Constancy*. John Wiley & Sons, Chichester, England, 2007.

[5] D. A. Forsyth and J. Ponce. *Computer Vision: A Modern Approach*. Prentice Hall, Upper Saddle River, NJ, 2003.

[6] E. H. Land. Recent advances in retinex theory and some implications for cortical computations: Color vision and the natural image. *Proc. Natl. Acad. Sci. USA*, 80:5163–5169, 1983.

[7] M. Linderoth, A. Robertsson, K. Åström, and R. Johansson. Vision based tracker for dart-catching robot. In *Preprints 9th IFAC Intl. Symp. on Robot Control (SYROCO'09)*, Gifu, Japan, September 9-12, 2009, pp. 883-888, Sept. 2009.

[8] V. Lippiello and F. Ruggiero. 3D monocular robotic ball catching with an iterative trajectory estimation refinement. In *Proc. IEEE Intl. Conf. Robotics and Automation (ICRA)*, pages 3950–3955, Saint Paul, Minnesota, USA, May 2012.

[9] Lotto Lab Studio, Illusions in colour perception. http://www.lottolab.org/articles/illusionsoflight.asp, 2011.

[10] M. Linderoth, Video of robot catching balls and darts. http://www.youtube.com/watch?v=Fxzh3pFr3Gs, 2009.

[11] M. Soriano, B. Martinkauppi, S. Huovinen, and M. Laaksonen. Skin detection in video under changing illumination conditions. In *Proc. 15th Intl. Conf. Pattern Recognition*, pages 839–842, Barcelona, Spain, September 2000.