In Daniele Nardi, Martin Riedmiller, and Claude Sammut, editors, *RoboCup 2004, Robot Soccer World Cup VIII*, pp. 196-208, Springer Verlag, Berlin, Germany, 2005.

Towards Illumination Invariance in the Legged League

Mohan Sridharan[1] and Peter Stone[2]

[1] Electrical and Computer Engineering, The University of Texas at Austin
smohan@ece.utexas.edu
[2] Department of Computer Sciences, The University of Texas at Austin
pstone@cs.utexas.edu
http://www.cs.utexas.edu/~pstone

Abstract. To date, RoboCup games have all been played under constant, bright lighting conditions. However, in order to meet the overall goal of RoboCup, robots will need to be able to seamlessly handle changing, natural light. One method for doing so is to be able to identify colors regardless of illumination: *color constancy*. Color constancy is a relatively recent, but increasingly important, topic in vision research. Most approaches so far have focussed on stationary cameras. In this paper we propose a methodology for color constancy on mobile robots. We describe a technique that we have used to solve a subset of the problem, in real-time, based on color space distributions and the KL-divergence measure. We fully implement our technique and present detailed empirical results in a robot soccer scenario.
Keywords: Illumination invariance, Color constancy, KL-divergence, mobile robots.

1 Introduction

Color constancy (or illumination invariance), though a major area of focus, continues to be a challenging problem in vision research. It represents the ability of a visual system to recognize an object's true color across a range of variations in factors extrinsic to the object (such as lighting conditions) [3]. In this paper, we consider the problem of color constancy on *mobile robots*.

In the RoboCup Legged League, teams of mobile robots, manufactured by Sony [2], coordinate their activities to play a game of soccer. To date, games have all been played on a small field ($4.4m \times 2.9m$) under constant, bright lighting conditions (Figure 1). However, the overall goal of the RoboCup initiative [1, 11] is, by the year 2050, to develop a team of humanoid robots that can beat the world champion human soccer team on a real outdoor soccer field. Color constancy is one of the important barriers to achieving this goal.

In the past, color constancy has been studied primarily on static cameras with relatively loose computational limitations. On mobile robots, color constancy must be achieved in real time, under constantly changing camera positions, while sharing computational resources with other complex tasks such as localization, movement, decision-making etc. This paper contributes a color constancy method based on the KL-divergence measure that is efficient enough to work on mobile robots that must operate under frequent illumination changes. Our method is fully implemented and tested on a concrete complex robot control task.

The remainder of the paper is organized as follows. In Section 2 we present some basic information on the problem and identify a subset of the overall problem that we address in this paper. Section 3 provides a brief review of related approaches that have been employed to solve the color constancy problem. Section 4 describes the experimental setup, the basic algorithm involved, and the mathematical details of our comparison measure. Details on the experimental results are provided in Section 5 followed by the conclusions in Section 6.

2 Background Information

In this section, we present a brief description of our experimental platform and describe the specific problem addressed in this paper.

On the Sony Aibo ERS-7 robots [2], we perform the visual processing in two stages: color segmentation and object recognition. During the initial off-board training phase, we train a color cube C that maps a space of $128 \times 128 \times 128$ possible pixel values[3] to one of the 10 different colors that appear in its environment (pink, yellow, blue, orange, marker green, red, dark blue, white, field green, black – see Figure 1). C is trained using a Nearest Neighbor (NNr) (weighted average) approach based on a set of hand-labelled input images. Due to computational constraints, C is precomputed and then treated as a lookup table. The robot uses C during task execution to segment an image and then recognize objects of importance. For full details on this process see our technical report [17].

Fig. 1. An Image of the Aibo and the field.

The actual pixel readings associated with a given object can vary significantly with changes in lighting conditions (both environmental and as a result of shadows) and there is significant overlap between some of the colors in the problem space. A color cube trained for one particular lighting condition can therefore be rendered ineffective by a reasonably small change (e.g., the difference between daytime and nighttime on the same playing field within a normal room with windows). In this paper, we propose an approach to solve this problem.

In our lab, the lighting on the soccer field varies significantly between the *bright* condition ($\approx$ 1500 lux with all lights on) and the *dark* condition ($\approx$ 350 lux with only the fluorescent ceiling lights on). One of the primary requirements

[3] We use half the normal resolution of 0-255 along each dimension to reduce storage space requirements.

to playing soccer is that of finding the ball and scoring a goal, which we define as the *find-ball-and-score-goal* task. If trained under the bright illumination condition, the robot is able to perform this task proficiently. But if the same robot is now made to function under the dark illumination condition (or any other illumination condition significantly different from the bright illumination condition), it is totally lost and cannot even recognize the ball. On the other hand, if the robot is equipped with a color cube trained under the dark illumination condition, it scores a goal in the same amount of time as in the bright illumination. Again, it is effectively blind when the lights are all turned on.

Our long-term goal is to enable the robot to perform this task in lighting conditions that may continuously vary between the two extremes (bright and dark). In this paper, we consider the subtask of enabling a robot to work in three illumination conditions: *bright*, *intermediate* and *dark* where *intermediate* refers to an illumination condition almost midway between the other two illumination conditions. Preliminary results indicate that solving this subtask may be sufficient for solving, or nearly solving, the long-term goal itself. In order to work in all three illumination conditions, the robot must be able to:

1. Correctly classify its input images into one of the three illumination conditions;
2. Transition to an appropriate color cube based on the classification and use that for subsequent vision processing;
3. Perform all the necessary computation in real-time without having an adverse effect on its task performance.

We present an algorithm that meets all of these requirements in Section 4. First, we take a brief look at some of the previous techniques developed to achieve illumination invariance.

3 Related approaches

Several approaches have been attempted to solve the problem of color constancy. Though they differ in the algorithmic details, most of them, to date, have focussed on static camera images. The Retinex Theory of Land [12] and the "Gray World" algorithm by Buchsbaum [5] are based on global or local image color averages, though these have later been shown to correlate poorly with the actual scene illuminant [4]. The *gamut mapping* algorithm, first proposed by Forsyth [9] and later modified by Finlayson [6, 7], using *median selection*, is based on a set of mappings that transform image colors (sensor values) under an unknown illuminant to the gamut of colors under a standard (canonical) illuminant. The probabilistic correlation framework, developed by Finlayson [8], operates by determining the likelihood that each of a possible set of illuminants is the scene illuminant.

The Bayesian decision theoretic approach, proposed by Brainard [3], combines all available image statistics and uses a maximum local mass (MLM) estimator to compute the posterior distributions for surfaces and illuminants in the scene for a given set of photosensor responses. Tsin [19] presents a Bayesian MAP (*maximum a posteriori*) approach to achieve color constancy for the task of outdoor object recognition with a static surveillance camera while Rosenberg [15] describes a method that develops models for sensor noise, canonical color and illumination, and determines the global scene illuminant parameters through an exhaustive search that uses KL-divergence as a metric. More recently, Lenser and Veloso [13] presented a tree-based state description/identification technique. They incorporate a time-series of average screen illuminance to distinguish between illumination conditions using the absolute value distance metric to determine the similarity between distributions. In this paper we explore an alternative similarity measure based on color space distributions.

In the domain of mobile robots, the problem of color constancy has often been avoided by using non-vision-based sensors such as laser range finders and sonar sensors [18]. Even when visual input is considered, the focus has been on recognizing just a couple of well-separated colors [10, 14]. There has been relatively little work on illumination invariance with a moving camera in the presence of shadows and artifacts caused by the rapid movement in complex problem spaces. Further, with few exceptions (e.g. [13]), the approaches that do exist for this problem cannot function in real-time with the limited processing power that we have at our disposal.

4 Approach

In this section, we introduce our experimental setup as well as our algorithmic framework.

4.1 Experimental Setup

We set out to see if it would be possible to distinguish between and adapt to the three different lighting conditions in our lab. Similar to our earlier work [16], we trained three different color cubes, one each for the *bright*, *intermediate*, and the *dark* illumination conditions.

We hypothesized that images from the same lighting conditions would have measurably similar distributions of pixels in color space. The original image is available in the YCbCr format, quantized into 256 bins: [0-255] along each dimension. In an attempt to reduce processing, but still retain the useful information, we transformed the image to the normalized RGB space, i.e. (r, g, b). By definition,

$$r = \frac{R+1}{R+G+B+3}, \quad g = \frac{G+1}{R+G+B+3}, \quad b = \frac{B+1}{R+G+B+3}$$

and $r + g + b = 1$. Thus any two of the three features are a sufficient statistic for the pixel values. For a set of training images captured at different positions on the field for each of the three illumination conditions, we then stored the distributions in the (r, g) space, quantized into 64 bins along each dimension. Once the distributions are obtained (one corresponding to each training image), the next question to address is the measure/metric to be used to compare any two given distributions.

4.2 Comparison Measure

In order to compare image distributions, we need a well-defined measure capable of detecting the correlation between color space image distributions under similar illumination conditions. We examined several such measures on sample images [16], we decided to use the KL-divergence measure [4].

KL-divergence is a popular measure for comparing distributions (especially discrete ones). Consider the case where we have a set of distributions in the 2D (r, g) space. Given two such distributions A and B (with $N = 64$, the number of bins along each dimension),

$$KL(A, B) = -\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}(A_{i,j} \cdot ln\frac{B_{i,j}}{A_{i,j}})$$

The more similar two distributions are, the smaller is the KL-divergence between them. Since the KL-divergence measure contains a term that is a function of the log of the observed color distributions, it is reasonably robust to large peaks in the observed color distributions and is hence less affected by images with large amounts of a single color.

4.3 Algorithmic Framework

Once we had decided on the measure to be used for comparing image distributions, we came up with a concrete algorithm to enable the robot to recognize and adapt to the three different illumination conditions (*bright*, *intermediate*, and *dark*).

The robot starts off assuming that it is in the bright illumination condition. It is equipped with one color cube for each illumination condition and a set of (training) sample distributions generated from images captured at various positions on the field. In our experiments, we used 24 distributions for each illumination condition, though, as we shall show, we do not need so many to perform satisfactorily.

As new images are processed, one is periodically tested for membership in one of the three illumination class, using a fixed number of training samples from each class for comparison. The distribution obtained from a test image is compared with the training samples using the KL-divergence measure, to determine the training sample that is *most similar* to it. The test image is then assigned the same illumination class label as this training sample. If sufficient number of consecutive images are classified as being from an illumination condition, the robot transitions to another color cube (representing the new illumination condition) and uses that for subsequent operations. Parameters included in this process were as follows:

- num_{train}: the number of training samples from each class per comparison.
- t: the time interval between two successive tests.

[4] Strictly speaking, KL-divergence is not a metric as it does not satisfy triangle inequality but it has been used successfully for comparing distributions.

– num_d, num_i, num_b: the number of consecutive dark, intermediate or bright classifications needed to switch to the corresponding color cube. We allow these parameters to differ.
– ang: a threshold camera tilt angle below which images are not considered for the purposes of changing color cubes. This parameter is introduced to account for the fact that the image appears dark when the robot is looking straight down into the ground (or the ball), which in most cases means that it is looking at a region enveloped in shadow.

5 Experimental Results

In this section, we describe the experiments that we used to determine the optimum values for the parameters. We then present the experiments that we ran to estimate the performance of the algorithm on the robot with respect to the goal-scoring task.

5.1 Estimation of parameters

On the ERS-7 robot, with our current code base [17], generating a test distribution takes $\approx 25msec$ and the comparison with all 72 training distributions takes $\approx 130msec$, i.e., comparing the test sample with each training sample takes $\frac{130}{72} \approx 2msec$. Further, the normal vision processing takes $\approx$ 30–35$msec$ per image, leading to a frame rate of 30 frames per second without any check for change in illumination. Thus, if we tested for illumination change on each input image, we would take $\approx 190msec$ per image giving us a very low frame rate of 5–6 frames per second, not including the processing required for action selection and execution. Doing so would lead to a significant loss of useful sensory information. As a result, we cannot afford to test each input image. Instead, we need to identify parameter values that do not significantly affect the normal operation of the robot while also ensuring that the illumination changes are recognized as soon as possible. With the parameter values that the robot ends up using we are still able to operate at around 25 frames per second.

Exp*1*: Classification accuracy The goal of this experiment was to determine how much the classification accuracy, using KL-divergence, depended on the parameters t and num_{train}. In order to do that the robot was placed at various positions on the field and only allowed to pan its head. The robot periodically sampled the images that it received from its camera and tested them to identify the illumination condition. The goal was to measure the real-time performance and see if there was any significant correlation between the performance of the robot and the associated position on the field. We chose six *test positions*, at each of which we measured the classification accuracy over a set of one thousand images. We first performed this experiment with $t = 1$ second, and with three different training sample sizes:

1. $Case1 : num_{train} = 24$ (all training samples).
2. $Case2 : num_{train} = 12$.

3. $Case3 : num_{train} = 6$.

Figure 2 shows the results of the experiment when performed under each illumination, corresponding to cases 1, 2 and 3. We also measured the number of times the robot transitioned between color cubes for each given test condition. Since each test was conducted completely under one of the three illumination conditions, ideally there would be no transitions. The values in parentheses therefore represent the number of (incorrect) color cube transitions that would occur if we chose to use the corresponding values of the parameters during normal operation. Observe that in the dark lighting conditions there were practically no misclassifications at any of the *test positions*.

	Bright		Interm		Dark	
num_{train}	Max	Min	Max	Min	Max	Min
24	$\frac{950}{1000}$(0)	$\frac{904}{1000}$(0)	$\frac{997}{1000}$(0)	$\frac{934}{1000}$(0)	$\frac{1000}{1000}$(0)	$\frac{1000}{1000}$(0)
12	$\frac{913}{1000}$(0)	$\frac{857}{1000}$(1)	$\frac{979}{1000}$(0)	$\frac{903}{1000}$(0)	$\frac{1000}{1000}$(0)	$\frac{999}{1000}$(0)
6	$\frac{874}{1000}$(0)	$\frac{712}{1000}$(3)	$\frac{964}{1000}$(0)	$\frac{850}{1000}$(0)	$\frac{1000}{1000}$(0)	$\frac{944}{1000}$(0)

Fig. 2. Classification accuracy and color cube transitions (all three illuminations)

The transition to the *bright* illumination takes place after two consecutive test images are classified as being under the bright illumination condition while for the other two illumination conditions (intermediate, dark), this threshold is set at four and six respectively (i.e., $num_d = 6$, $num_i = 4$ and $num_b = 2$). The transition parameters were weighted in this manner because it was noted during experimentation that lighting inconsistencies (such as shadows) during both training and testing led to significantly more noise in the bright conditions.

From a close examination of the raw data, we report several observations:

- In the dark illumination, the robot did not err even in the case where $num_{train} = 6$. This makes sense considering the fact that the dark illumination condition is much different from the bright illumination condition (350lux vs 1500lux) and the presence of shadows or obstructions only makes the image *darker*.
- In the bright and intermediate illumination conditions, the number of training samples made a difference in the performance of the robot; at each position, the performance worsens as num_{train} is decreased.
- The transitions (shifts) that occurred in the bright illumination condition were due to shadows or obstructions and this caused the robot to move from the bright to the intermediate illumination condition.
- In the rare case that the robot changed to the incorrect color cube, the error was sustained only for a few test frames before the robot recovered.

Next we varied t and repeated the experiments performed previously. Here, no significant change was noticed in the classification accuracy. With $t = 0.5$ seconds or 0.25 seconds instead of 1 second there was no significant change in the classification accuracy. However it did increase the processing performed. We quantify this effect in subsequent experiments.

Exp2: Task Execution The *find-ball-and-score-goal* task was incorporated in this experiment to estimate the effect of the color constancy algorithm on the robot's task performance under constant illumination. The robot was placed at the center of the field facing one of the goals (say g_1) while the ball was placed at the center of the penalty box around the other goal (g_2) (see Figure 1). The robot had to find the orange ball and then score on g_2. We performed this experiment with the robot trying to score on either goal (blue/yellow) over one half of the total number of trials. In this process, it used the colored markers around the field to localize itself [17].

The robot performed the check for change in the illumination condition with $t = 1$ and $ang = -10$ (i.e. consider cases where the tilt angle of the camera is greater than 10^o) and its accuracy was tested under all three sampling conditions used in experiment 1, i.e., we tested for $num_{train} = 24$, 12, and 6. We set the tilt angle threshold to ensure that when the robot is staring down at the ground or the ball, the shadows do not cause the robot to make a wrong transition. Figure 3 displays the classification accuracy and the number of transitions that occurred during task execution under all three cases. Since testing was done under each illumination condition separately, the *shifts* column represents the number of incorrect transitions that occurred.

num_{train}	Bright	Shifts	Intermediate	Shifts	Dark	Shifts
24	$\frac{432}{500}$	0	$\frac{456}{500}$	0	$\frac{480}{500}$	0
12	$\frac{398}{500}$	1	$\frac{421}{500}$	1	$\frac{482}{500}$	0
6	$\frac{360}{500}$	2	$\frac{395}{500}$	1	$\frac{490}{500}$	0

Fig. 3. Accuracy and transitions under each illumination under all three sampling cases

From the results, we deduced that the value of num_{train} does not make a big change in the dark illumination case; the misclassifications that did occur in the dark illumination case happened when the robot fell down and was staring at the white border or field lines. But this was not the case under the bright and intermediate illuminations; with the decrease in num_{train}, the robot ended up making more errors (and wrong transitions between color cubes). The errors that occurred were mostly due to shadows when the robot was running into the goal. But the robot always recovered within a few test frames (fewer than 5).

Under this tilt angle setting, the (incorrect) color cube transitions were only one-off from the actual illumination condition, i.e., there were no incorrect transitions from *bright* to *dark* or vice versa. We could set higher tilt angle thresholds but then the robot is slow to identify changed illumination conditions which has a bad effect on its overall performance.

Exp3: Parameter combinations Next, we wanted to determine the parameter settings that would enable strong real-time performance. Specifically, we considered the parameters num_{train} and t.

To do so, we defined the *find-and-walk-to-ball* task. This task is identical to the *find-ball-and-score-goal* task, except that the robot only needs to find the ball and walk up to it, rather than actually score. This modification makes the measurements less dependent on the performance of other modules, such as kicking.

Under constant lighting conditions with a single color cube, the robot can *find-and-walk-to-ball* in 6.7($\pm$0.6) seconds. The results for the bright illumination case, averaged over 10 trials, are in Figure 4. The values for the other two illuminations were not significantly different (as expected). We considered the cases where the robot did not find the ball after two minutes to be complete misses and omitted them from the results. The numbers in parentheses indicate the number of complete misses that occurred while collecting 10 values.

t (sec)	num_{train}=24	num_{train}=12	num_{train}=6
1.0	6.8±0.3(0)	6.8±0.4(0)	6.8±0.4(0)
0.5	6.9±0.3(0)	7.0±0.6(0)	7.0±0.5(0)
0.25	8.8±0.6(2)	9.1±1.8(0)	8.2±1.9(0)
0.125	51.8±31.7(4)	18.3±2.5(1)	11.7±6.2(0)
0.0	75.0±36.9(6)	52.8±13.1(3)	13.8±3.0(0)

Fig. 4. Time taken (in seconds) to *find-and-walk-to-ball* under bright illumination

From Figure 4, we conclude that:

- The parameter values $t = 1$ second and $t = 0.5$ seconds (and to some extent $t = 0.25$ seconds), with all three sampling schemes, work fine on the robot in real-time without having an adverse effect on the normal game playing performance.
- With all the other testing frequencies there were instances, especially with $num_{train} = 24$, when the robot missed the ball during its scan due to the computation involved; by the time the robot had processed one frame that had the ball, its head was beyond the position were it could recognize the ball (the robot needs to see the ball continuously for around 3 frames before it accepts it as the ball [17]). In addition, when the testing was done on every frame (or even once every 0.125 seconds), the robot's motion towards the ball was extremely jerky.

5.2 Changing Illumination

Once we had determined values for all the parameters, we were ready to test the robot on its task under changing illumination conditions. Based on the experiments described above, we chose the parameter values: $ang = -10^o$, $t = 1$ second, $num_d = 6$, $num_i = 4$, $num_b = 2$, and $num_{train} = 24$.

Real-time Transitions In these experiments, the robot was tested with the lighting conditions changing after a specific interval. The robot starts off in one illumination condition and after 1.5 seconds (the time it takes the robot to turn and see the ball), the illumination is changed by adjusting the intensity of all the lamps. The robot is then timed as it performs the *find-and-walk-to-ball* task. Recall that with a single color cube, the robot is unable to do so: when the illumination condition changes significantly, it is unable to see a ball that is right in front of its camera. Now, when the illumination conditions change, the robot seems lost for a couple of seconds while it recognizes the change and then functions as normal, scoring goals once again. The results are shown in Figure 5[5].

Lighting (start/after 1.5 seconds)	Time (seconds)
bright / intermediate	8.5 ±0.9
bright / dark	11.8 ±1.3
intermediate / bright	8.6 ±1.0
intermediate / dark	9.6 ±3.1
dark / intermediate	11.5 ±1.4
dark / bright	10.7 ±1.1

Fig. 5. Time taken to *find-and-walk-to-ball* under changing illumination

Stress Tests To further explore the robustness of our approach, we report the results of two tests for which the current algorithm was not designed: intermediate lighting conditions (Test 1) and adversarial illumination changes (Test 2).

<u>*Test* 1</u> The first experiment we performed involved reducing the intensity of the lamps in specific patterns. In our lab, we have four lamps mounted on stands along the shorter edges of the field, as shown in Figure 6.

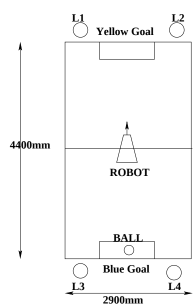

Fig. 6. A Line drawing of the field and the lamp arrangement.

During testing, we reduced the intensity of all the lamps such that the illumination on the field is in between the illumination conditions that the robot was explicitly trained for. To enable comparison of these results, we used the *find-and-walk-to-ball* task as in previous sections and recorded the time taken by the robot to perform the task. In Figure 7 we present the values corresponding to the case wherein the robot starts off in the *bright* illumination condition. About 1.5 seconds later, the lighting is changed such that it is between the *bright* and the *intermediate* illuminations (we also tested with the illumination changed to be midway between the *intermediate* and the *dark* conditions).

<u>*Test* 2</u> Finally, we decided to test the robot under adversarial conditions. That is, we tried our best to confuse the robot completely by varying the illumination in the worst possible way for the robot. Here, we performed the *find-ball-and-score-goal* task.

[5] Video showing the robots performing under varying lighting conditions is available at `http://www.cs.utexas.edu/~AustinVilla/legged/illumination`

As soon as the robot recognized the ball and started walking towards it, we changed the illumination condition. That is, as soon as the robot transitioned to the correct color cube, as indicated by an LED on the robot, we would change the illumination to be one class away from the actual illumination condition. Due to the values of num_d, num_i and num_b, assuming that we start off under the bright illumination, the experiment would involve changing the illumination to correspond to the intermediate illumination after two seconds. We would turn the lamps off (dark illumination) after another four seconds, which would be followed by adjusting the intensity of the lamps to half the maximum value (intermediate illumination) after a further six seconds. Then we would turn the lamps on at full intensity (bright illumination) after around four seconds. This cycle of change in illumination is performed repeatedly and Figure 8 depicts the corresponding average time (and standard deviation) taken to accomplish the *find-ball-and-score-goal* task.

Lighting	Time (seconds)
bet. bright and interm	12.27 $\pm$0.5
bet. interm and dark	13.3 $\pm$2.0

Fig. 7. Time taken (in seconds) to *find-and-walk-to-ball*

The fact that the robot can perform this task at all is due to the fact that it can occasionally recognize the ball even if it is not using the color cube corresponding to the current illumination condition. Even in the case where some of the lamps are selectively turned off (or their intensity is reduced), the robot transitions into an appropriate color cube and is still able to perform the task of scoring on the goal. The only way we could confuse the robot further would be to change between the bright and the dark illumination conditions in a similar manner. In that case, the robot does not make any progress to the ball at all.

Lighting	Time (seconds)
Adversarial	34.3 ($\pm$7.8)

Fig. 8. Time taken (in seconds) to *find-ball-and-score-goal*

An important point to note here is that in previous work [16] (where we had incorporated only two illumination conditions: bright and dark) we had problems when we tested the robot on illumination conditions that it was not trained for. There were problems especially while trying to score on the yellow goal in differentiating between yellow and orange (and between pink and orange). The robot would then walk away in an entirely wrong direction in an attempt to follow a spurious estimate of the ball. We had then hypothesized that adding a few more illumination conditions in between the two extreme ones might help alleviate some of the problems. We now see that with the added *intermediate* illumination condition, the robot does perform much better. In fact, in all the experiments mentioned above, the robot performed equal number of trials scoring/walking towards either goal (blue/yellow).

Also, in our earlier work [16] we used the previous version of the Sony robots: ERS210A. With very little modification in code (only to incorporate one more illumination condition), the strategy works fine to distinguish between three different illumination conditions. We find that with this change, the robot is better able to work in illumination conditions corresponding to which the robot does not have training samples.

6 Conclusions/Future Work

In this paper, we have presented an approach that works in real-time to achieve color constancy on mobile robots in the RoboCup domain. The technique uses color space distributions, easy to train color cubes, and a simple and efficient comparison measure (KL-divergence) to determine and adapt to three discrete illumination conditions. Though we have solved only a subset of the problem, the results obtained seem to indicate that we do not need to consider a continuous spectrum of illuminations. When presented with illumination conditions that the robot is not trained for, there is little degradation of performance.

The problem of color constancy, on mobile robots or otherwise, is extremely challenging and is far from being solved. In the future, we shall first try to extend the approach to enable the robot to perform well under an even wider range of possible illumination conditions. One possible method would be to train a few discrete color cubes to represent significantly different illuminations (as we have done here) and then dynamically update the cubes for minor variations in illumination conditions. We shall also look into alternate stochastic approaches that may enable us to achieve illumination invariance without having to resort to training several color cubes. Ultimately, we aim to solve the daunting problem of developing efficient algorithms that enable a mobile robot to function under completely uncontrolled natural lighting conditions, with all its associated variations.

Acknowledgements

We would like to thank the members of the UT Austin Villa team for their efforts in developing the soccer-playing software mentioned in this paper. This research was supported in part by NSF CAREER award IIS-0237699.

References

1. The International RoboSoccer Competition. `http://www.robocup.org`.
2. The Sony Aibo robots. `http://www.us.aibo.com`.
3. D. H. Brainard and W. T. Freeman. Bayesian color constancy. *Journal of Optical Soceity of America A*, 14(7):1393–1411, 1997.
4. D. H. Brainard and B. A. Wandell. Analysis of the retinex theory of color vision. *Journal of Optical Soceity of America A*, 3(10):1651–1661, 1986.
5. G. Buchsbaum. A spatial processor model for object color perception. *Journal of Franklin Institute*, 310:1–26, 1980.
6. G. Finlayson. Color in perspective. *In IEEE Transactions of Pattern Analysis and Machine Intelligence*, 18(10):1034–1038, July 1996.
7. G. Finlayson and S. Hordley. Improving gamut mapping color constancy. *In IEEE Transactions on Image Processing*, 9(10), October 2000.
8. G. Finlayson, S. Hordley, and P. Hubel. Color by correlation: A simple, unifying framework for color constancy. *In IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11), November 2001.

9. D. Forsyth. A novel algorithm for color constancy. *In International Journal of Computer Vision*, 5(1):5–36, 1990.
10. Jeff Hyams, Mark W. Powell, and Robin R. Murphy. Cooperative navigation of micro-rovers using color segmentation. *In Journal of Autonomous Robots*, 9(1):7–16, 2000.
11. Hiroaki Kitano, Minoru Asada, Yasuo Kuniyoshi, Itsuki Noda, and Eiichi Osawa. Robocup:the robot world cup initiative. proceedings of the first international conference on autonomous agents. In *Proceedings of the International Conference of Robotics and Automation*, pages 340–347, February 1997.
12. E. H. Land. The retinex theory of color constancy. *Scientific American*, pages 108–129, 1977.
13. S. Lenser and M. Veloso. Automatic detection and response to environmental change. In *Proceedings of the International Conference of Robotics and Automation*, May 2003.
14. B. W. Minten, R. R. Murphy, J. Hyams, and M. Micire. Low-order-complexity vision-based docking. *IEEE Transactions on Robotics and Automation*, 17(6):922–930, 2001.
15. C. Rosenberg, M. Hebert, and S. Thrun. Color constancy using kl-divergence. In *IEEE International Conference on Computer Vision*, 2001.
16. M. Sridharan and P. Stone. Towards illumination invariance on mobile robots. In *The First Canadian Conference on Computer and Robot Vision*, 2004.
17. Peter Stone, Kurt Dresner, Selim T. Erdoğan, Peggy Fidelman, Nicholas K. Jong, Nate Kohl, Gregory Kuhlmann, Ellie Lin, Mohan Sridharan, Daniel Stronger, and Gurushyam Hariharan. Ut austin villa 2003: A new robocup four-legged team, ai technical report 03-304. Technical report, Department of Computer Sciences, University of Texas at Austin, October 2003.
18. S. Thrun, D. Fox, W. Burgard, and F. Dellaert. Robust monte carlo localization for mobile robots. *Journal of Artificial Intelligence*, 2001.
19. Y. Tsin, R. T. Collins, V. Ramesh, and T. Kanade. Bayesian color constancy for outdoor object recognition. *In IEEE Pattern Recognition and Computer Vision*, December 2001.