

Improving Percept Reliability in the Sony Four-Legged Robot League^{*}

Walter Nisticò¹ and Thomas Röfer²

¹ Institute for Robot Research (IRF), Universität Dortmund
walter.nisticò@udo.edu

² Center for Computing Technology (TZI), FB 3, Universität Bremen
roefter@tzi.de

Abstract. This paper presents selected methods used by the vision system of the GermanTeam, the World Champion in the Sony Four-Legged League in 2004. Color table generalization is introduced as a means to achieve a larger independence of the lighting situation. Camera calibration is necessary to deal with the weaknesses of the new platform used in the league, the Sony Aibo ERS-7. Since the robot camera uses a rolling shutter, motion compensation is required to improve the information extracted from the camera images.

1 Introduction

The perceptive layer in a robotic architecture is the primary source of information concerning the surrounding environment. In case of the RoboCup 4-legged league, due to the lack of range sensors (e.g. laser scanners, sonars), the robot only has the camera to rely upon for navigation and object detection, and in order to be able to use it to measure distances, the camera position in a robot-centric reference system has to be dynamically estimated from leg and neck joint angle measurements, having to deal with noise in these measures. The robot interacts with a dynamic and competitive environment, in which it has to quickly react to changing situations, facing real-time constraints; image processing tasks are generally computationally expensive, as several operations have to be performed on a pixel level, thus with an order of magnitude of $10^5 - 10^6$ per camera frame. The need to visually track fast moving objects (i.e. the ball) in the observed domain, further complicated by the limited camera field of view, makes it necessary for the vision system to be able to keep up with the highest possible frame rate that the camera can sustain: in the case of the robot Sony Aibo ERS-7, 30 fps. As a result, image segmentation is still mainly achieved through static color classification (see [3]).

2 Color Table Generalization

Robustness of a vision system to lighting variations is a key problem in RoboCup, both as a long term goal to mimic the adaptive capabilities of organic systems,

* The Deutsche Forschungsgemeinschaft supports this work through the priority program “Cooperating teams of mobile robots in dynamic environments”.

as well as a short term need to deal with unforeseen situations which can arise at the competitions, such as additional shadows on the field as a result of the participation of a packed audience. While several remarkable attempts have been made in order to achieve an image processor that doesn't require manual calibration (see [5], [12]), at the moment traditional systems are more efficient for competitions such as the RoboCup. Our goal was to improve a manually created color table, to extend its use to lighting situations which weren't present in the samples used during the calibration process, or to resolve ambiguities along the boundaries among close color regions. Thereto, we have developed a color table generalization technique which uses an exponential influence model similar to the approach described in [7], but in contrast to it, is not used to perform a semi-automated calibration from a set of samples. This new approach is based on the assumption of spatial locality of the color classes in the color space, so that instead of the frequency of a set of color samples, it's the spatial frequency of neighbors in the source color map to determine the final color class of a given point, following the idea that otherwise, a small variation in lighting conditions, producing a spatial shift in the mapping, would easily result in classification errors. Thus, a color table is processed in the following way:

- Each point assigned to a color class irradiates its influence to the whole color space, with an influence factor exponentially decreasing with the distance:

$$I_i(p_1, p_2) = \begin{cases} \lambda^{|p_1 - p_2|} & i = c(p_2) \\ 0 & \forall i \neq c(p_2) \end{cases} \quad (1)$$

where p_1, p_2 are two arbitrary points in the color map, $\lambda < 1$ is the exponential base, $I_i(p_1, p_2)$ is the influence of p_2 on the (new) color class $i \in \{\text{red, orange, yellow, } \dots\}$ of p_1 , and $c(p_2)$ is the color class of p_2 ,

- Manhattan distance is used (instead of Euclidean) to speed up the influence calculation ($O(n^2)$, where n is the number of elements of the color table):

$$|p_1 - p_2|_{\text{manhattan}} = |p_{1y} - p_{2y}| + |p_{1u} - p_{2u}| + |p_{1v} - p_{2v}| \quad (2)$$

- For each point in the new color table, the total influence for each color class is computed:

$$I_i(p_0) = B_i \cdot \sum_{p \neq p_0} I_i(p_0, p) \quad (3)$$

where $B_i \in (0..1]$ is a bias factor which can be used to favor the expansion of one color class over another

- The color class that has the highest influence for a point is chosen, if:

$$\frac{\max(I_i)}{I_{bk} + \sum_i I_i} > \tau \quad (4)$$

where τ is a confidence threshold, I_{bk} is a constant value assigned to the influence of the background (noColor) to prevent an unbounded growth of the colored regions to the empty areas, and i again represents the color class.

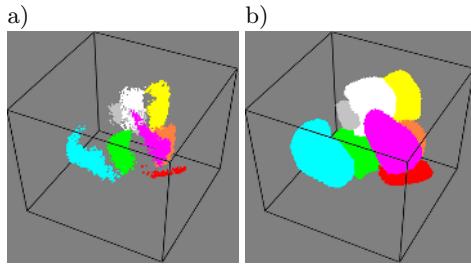


Fig. 1. Effects of the exponential generalization on a color table: (a) original, (b) optimized

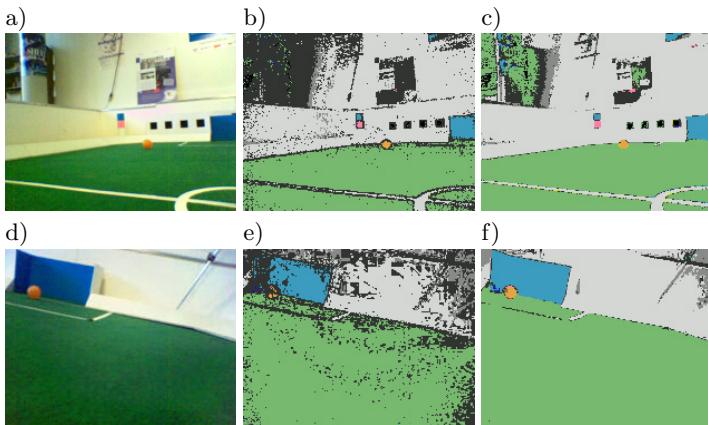


Fig. 2. Exponential generalization. (a) and (d) represent images taken from the same field, but in (d) the amount of sunlight has increased: notice the white walls appearing bluish. (b) and (e) are the result of the classification from the original color table, calibrated for the conditions found in (a); notice that (e) is not satisfactory, as the ball is hard to detect and the goal appears largely incomplete. (c) and (f) are classified using the generalized table, showing that it can gracefully accommodate to the new lighting conditions (f).

The parameters λ , τ , B_i , I_{bk} control the effects of the generalization process, and we have implemented 3 different settings: one for conservative generalization, one for aggressive expansion, one for increasing the minimum distance among neighboring regions. The time required to apply this algorithm, on a 2^{18} elements table, is $\approx 4 - 7$ minutes on a 2.66GHz Pentium4 processor, while for a table of 2^{16} elements, this figure goes down to only 20-30 seconds.

3 Camera Calibration

With the introduction of the ERS-7 as a platform for the 4-Legged League, an analysis of the camera of the new robot was required to adapt to the new

specifications. While the resolution of the new camera is $\approx 31\%$ higher compared to the previous one, preliminary tests revealed some specific issues which weren't present in the old model. First, the light sensitivity is lower, making necessary the use of the highest gain setting, at the expense of amplifying the noise as well; such a problem has been addressed in [10]. Second, images are affected by a vignetting effect (radiometric distortion), which makes the peripheral regions appear darker and dyed in blue.

3.1 Geometric Camera Model

In the previous vision system, the horizontal and vertical opening angles of the camera were used as the basis for all the measurement calculations, following the classical "pinhole" model; however for the new system we decided to use a more complete model taking into account the geometrical distortions of the images due to lens effects, called the *DLT model* (see [1].) This model includes the lack of orthogonality between the image axes s_θ , the difference in their scale (s_x, s_y), and the shift of the projection of the real optical center (principal point) (u_0, v_0) from the center of the image (together called "intrinsic parameters") and the rotation and translation matrices of the camera reference system relative to the robot (R, T , "extrinsic parameters"). In addition to this, we have also decided to evaluate an augmented model including radial and tangential non-linear distortions with polynomial approximations, according to [4] and [8]. In order to estimate the parameters of the aforementioned models for our cameras, we used a Matlab toolbox from Jean-Yves Bouguet (see [2]). The results showed that the coefficients (s_x, s_y, s_θ), are not needed, as the difference in the axis scales is below the measurement error, and so is the axis skew coefficient; the shift between the principal point (u_0, v_0) and the center of the image is moderate and dependent from robot to robot, so we have used an average computed from images taken from 5 different robots. As far as the non-linear distortion is concerned, the results calculated with Bouguet's toolbox showed that on the ERS-7 this kind of error has a moderate entity (maximum displacement ≈ 3 pixel), and since in our preliminary tests, look-up table based correction had an impact of ≈ 3 ms on the running time of the image processor, we decided not to correct it.

3.2 Radiometric Camera Model

As the object recognition is still mostly based on color classification, the blue cast on the corners of the images captured by the ERS-7's camera is a serious hindrance in these areas. Vignetting is a radial drop of image brightness caused by partial obstruction of light from the object space to image space, and is usually dependent on the lens aperture size ([9], [6]), however, in this case the strong chromatic alteration seems difficult to explain merely in terms of optics, and we suspect it could be partially due to digital effects. To be able to observe the characteristics of this vignetting effect, we captured images of uniformly colored objects from the robot's camera, lit by a diffuse light source (in order to minimize the effects of shadows and reflections). As can be seen in Figure 3,

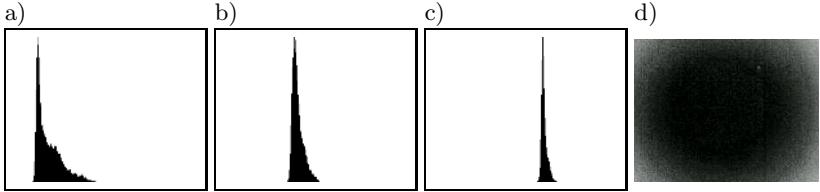


Fig. 3. (a,b,c) Histograms of the U color band for uniformly colored images: yellow (a), white (b) and skyblue (c). In case of little or no vignetting effect, each histogram should exhibit a narrow distribution around the mode, like in (c). (d) Brightness distribution of the U color band for a uniformly colored yellow image.

the radiometric distortion d_i for a given spectrum i of a reference color I is dependent on its actual value (*brightness component*):

$$d_i(I) \propto \lambda_i(I_i) \quad (5)$$

Moreover, the chromatic distortion that applies on a certain pixel (x, y) appears to be also dependent on its distance from a certain point (cf. Fig. 3(d)), center of distortion (u_d, v_d) , which lies approximately close to the optical center of the image, the principal point; so, let $r = \sqrt{(x - u_d)^2 + (y - v_d)^2}$, then (*radial component*):

$$d_i(I(x, y)) \propto \rho_i(r(x, y)) \quad (6)$$

Putting it all together:

$$d_i(I(x, y)) \propto \rho_i(r(x, y)) \cdot \lambda_i(I_i(x, y)) \quad (7)$$

Now, we derive $\rho_i, \lambda_i, \forall i \in \{Y, U, V\}$ from a set of sample pictures; since both sets of functions are non-linear, we decided to use a polynomial approximation, whose coefficients can be estimated using least-square optimization techniques:

$$\begin{aligned} \rho_i(r) &= \sum_{j=0}^n \varrho_{i,j} \cdot r^j \\ \lambda_i(I_i) &= \sum_{j=0}^m l_{i,j} \cdot I_i^j \end{aligned} \quad (8)$$

In order to do so, we have to create a log file containing reference pictures which should represent different points belonging to the functions that we want to estimate, hence we chose to use uniform yellow, blue, white and green images taken under different lighting conditions and intensities. Then, the log file is processed in the following steps:

- For each image, a reference value is estimated for the 3 spectra Y, U, V, as the modal value of the corresponding histogram (*numOfBins = color Levels = 256*).
- The reference values are clustered into classes, such that series of images representing the same object under the same lighting condition have a single

reference value; this is achieved using a first order linear Kalman filter to track the current reference values for the 3 image spectra, and a new class is generated when:

$$\exists j \in \{Y, U, V\} : |r_{j,k}^m - r_{j,k-1}^p| > \vartheta \quad (9)$$

where $r_{j,k}^m$ is the reference (for spectrum j) measured at frame k , $r_{j,k-1}^p$ is the reference predicted by the Kalman filter at frame $k-1$, and $\vartheta = 40$ is a confidence threshold.

- Simulated annealing ([11]) is used to derive the coefficients (u_d, v_d), and $\varrho_{i,j}, l_{i,j} \forall i \in \{Y, U, V\}$ (in a separate process for each color band).
- In each step, the coefficients $\varrho_{i,j}, l_{i,j}$ are mutated by the addition of zero mean gaussian noise, the variance is dependent on the order of the coefficients, such that high order coefficients have increasingly smaller variances than low order ones.
- The mutated coefficients are used to correct the image, as:

$$I'_i(x, y) = I_i(x, y) - \rho_i(r(x, y)) \cdot \lambda_i(I_i(x, y)) \quad (10)$$

- For each image $I_{i,k}$ in the log file (i is the color band, k the frame number), given its reference value previously estimated $r_{i,k}$, the current “energy” E for the annealing process is calculated as:

$$E_i = \sum_{(x,y)} (I'_{i,k}(x, y) - r_{i,k})^2 \quad (11)$$

- The “temperature” T of the annealing is lowered using a linear law, in a number of steps which is given as a parameter to the algorithm to control the amount of time spent in the optimization process; the starting temperature is normalized relative to the initial energy.
- The correction function learned off-line is stored in a look-up table for a fast execution on the robot.

Figure 4 shows some examples of corrections obtained after running the algorithm on a log file composed of 8 image classes (representing different colors at different lighting conditions) of 29 images each, decreasing the temperature to

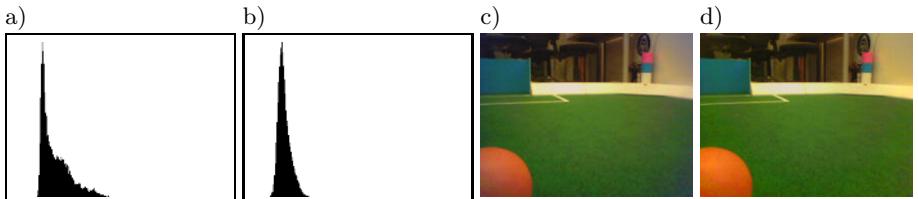


Fig. 4. Color correction in practice: histograms of the U color band of a uniformly yellow colored image before correction (a), and after (b); actual image taken from a game situation, before correction (c) and after (d).

0 in 100 steps, for a total optimization time of 7 minutes (Pentium4 2.66GHz). In case of the image spectra which exhibit the highest distortion (Y, U), the variance after the calibration is reduced by a factor of 10.

4 Motion Compensation

The camera images are read sequentially from a CMOS chip using a *rolling shutter*. This has an impact on the images if the camera is moved while an image is taken, because each scan line is captured at a different time instant. For instance in Figure 5 it can be seen, that the flag is slanted in different directions depending on whether the head is turning left or right. In experiments it was recognized that the timestamp attached to the images by the operating system of the Aibo corresponds to the time when the lowest row of the image was taken. Therefore, features in the upper part of the image were recorded significantly earlier. It is assumed that the first image row is recorded shortly after taking the previous image was finished, i. e. 10% of the interval between two images, so 90% of the overall time is spent to take the images. For the ERS-7, this means that the first row of an image is recorded 30 ms earlier than the last row. If the head, e. g., is rotating with a speed of $180^\circ/\text{s}$, this results in an error of 5.4° for bearings on objects close to the upper image border. Therefore, the bearings have to be corrected. Since this is a quite time-consuming operation, it is not performed as a preprocessing step for image processing. Instead, the compensation is performed on the level of percepts, i. e. recognized flags, goals, edge points, and the ball. The compensation is done by interpolating between the current and the previous camera positions depending on the y image coordinate of the percepts.

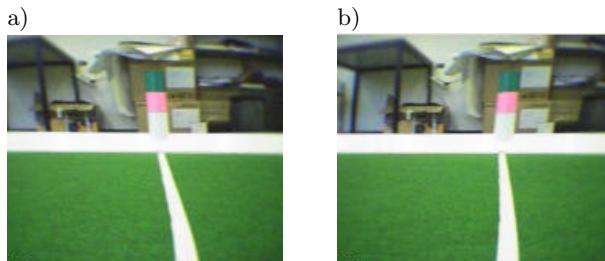


Fig. 5. Images taken while the head is quickly turning. a) Left. b) Right.

5 Results

The algorithms described here have been tested and used as part of the vision system of the GermanTeam which became World Champion in the Four-Legged League at the RoboCup 2004 competitions. The average running time of the whole image processor was 9 ± 2 ms, and the robot was able to keep up with the maximum camera frame rate under all circumstances, i.e. 30 fps. Throughout the competitions, our vision system proved to be robust and accurate, and

our robots' localization was widely acclaimed as the best of the league; these techniques have also been used on the vision system of Microsoft Hellhounds, a member of the GermanTeam, achieving the second place in the Variable Lighting Technical Challenge at the Japan Open 2004 competitions.

Acknowledgments

The program code used was developed by the GermanTeam, a joint effort of the Humboldt-Universität zu Berlin, Universität Bremen, Universität Dortmund, and Technische Universität Darmstadt. <http://www.germanteam.org/>

References

1. H. Bakstein. A complete dlt-based camera calibration, including a virtual 3d calibration object. Master's thesis, Charles University, Prague, 1999.
2. J.-Y. Bouguet. Camera calibration toolbox for matlab. http://www.vision.caltech.edu/bouguetj/calib_doc/.
3. J. Bruce, T. Balch, and 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)*, volume 3, pages 2061–2066, 2000.
4. J. Heikkilä and O. Silvén. A four-step camera calibration procedure with implicit image correction. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'97)*, pages 1106–1112, 1997.
5. M. Jüngel. Using layered color precision for a self-calibrating vision system. In *8th International Workshop on RoboCup 2004 (Robot World Cup Soccer Games and Conferences)*, Lecture Notes in Artificial Intelligence. Springer, 2005.
6. S. B. Kang and R. S. Weiss. Can we calibrate a camera using an image of a flat, textureless lambertian surface?, 2000.
7. S. Lenser, J. Bruce, and M. Veloso. Vision - the lower levels. Carnegie Mellon University Lecture Notes, October 2003. <http://www-2.cs.cmu.edu/robosoccer/cmrobobits/lectures/vision-low-level-lec/vision.pdf>.
8. R. Mohr and B. Triggs. Projective geometry for image analysis, 1996.
9. H. Nanda and R. Cutler. Practical calibrations for a real-time digital omnidirectional camera. Technical report, CVPR 2001 Technical Sketch, 2001.
10. W. Nistico, U. Schwiegelshohn, M. Hebbel, and I. Dahm. Real-time structure preserving image noise reduction for computer vision on embedded platforms. In *Proceedings of the International Symposium on Artificial Life and Robotics, AROB 10th*, 2005.
11. S. Russel and P. Norvig. *Artificial Intelligence, a Modern Approach*. Prentice Hall, 1995.
12. D. Schulz and D. Fox. Bayesian color estimation for adaptive vision-based robot localization. In *Proceedings of IROS*, 2004.