Color Normalization and Object Localization

L. Csink[1], D. Paulus[2], U. Ahlrichs[2], B. Heigl[2]

Accepted for 4. Farbworkshop 98, Koblenz

Contents

File `/home/paulus/text/papers/KEPAF-97/RCS/fws98p.tex,v`
Version `1.2` vom `14. August 1998`
LaTeX `24th September 1998`
Pages 6 (max: 7)

Color Normalization and Object Localization

L. Csink[1], D. Paulus[2], U. Ahlrichs[2], B. Heigl[2]

[1] Kandó Polytechnic of Technology
Institute of Informatics
POB 112, H-1431 Budapest, Hungary
Tel.: +36–1–3684610, FAX: +36–1–3689632
csink@novserv.obuda.kando.hu

[2] Universität Erlangen–Nürnberg
Lehrstuhl für Mustererkennung (Informatik 5)
Martensstr. 3, D–91058 Erlangen, Germany
Tel.: +49–9131–85-7775, FAX: +49–9131–303811
paulus@informatik.uni-erlangen.de

http://www5.informatik.uni-erlangen.de

Abstract

The distribution of color values in color images depends on the illumination which varies widely under real–world conditions. Object recognition based on color information is expected to compensate for these effects.

Several algorithms have been reported for color constancy or color normalization. We shortly review some of these: color rotation in various color spaces, whitening, and comprehensive color normalization. Then we use histogram backprojection to find our objects in various scenes using each of the above methods for normalization.

Object positions are hand–segmented in real–world office scenes, in which illumiation varied as well as different cameras were used. These positions are compared to the automatically computed results, using each algorithm outlined above, or no normalization. In some cases depending on the object, color normalization enhanced finding rate. The normalization module is integrated into our complete system for knowledge based visual exploration of scenes.

1 Introduction

Research on content–based retrieval from (color) image databases is getting more and more active. Image databases are today widely used by a large number of different professionals as well as by the average person with a computer.

However, tools for manipulating image databases are not sufficiently developed compared to those available for information retrieval systems. Usually keywords are used to index images, however, text-based indexing is not appropriate for images. Increasingly, there has been focus on content-based techniques for image retrieval instead. Along with texture and shape, color is an important element of visual information.

One typical simple question may be whether the image contains any or more of the elements of a given set of natural objects. To answer this question one needs to have effective methods for *object localization* and *color normalization*.

[1]This work was funded by DAAD, Germany, in 1998 and partially by MKM-FKFP grant no. 0908.

[2]This work was funded partially by the Deutsche Forschungsgemeinschaft (DFG) under grant number SFB 182. Only the author is responsible for the contents.

In our system described in Sect. 2, object localization is performed via histogram backprojection which has the advantage of little sensitivity to changes in orientation of objects and to partial occlusion. However, it is influenced by illumination that may have been different in case of object and scene in database. A different camera may also cause differences. Color normalization intends to tackle these kinds of problems. In [7, 8] we have developed two color normalization algorithms and in this project we compare them with some other implementations, applied to the localization problem in various scenes taken under varying conditions. In Sect. 3 we review these developments. In Sect. 4 we describe the testbed used in more than 4000 experiments which are interpreted in Sect. 5.

2 Object Localization with Histogram Backprojection

In the system for active knowledge–based scene exploration [6], we use a pan/tilt/zoom camera mounted on a linear sledge to search for objects in an office room. 3–D information is computed from a sequence of images recorded during linear motion on the sledge by tracking colored points. Individual objects are presented to the system and then removed from the scene. Using a difference image and erosion, a mask is generated for each object and an image of the object is extracted. A color histogram is then computed for each object afterwards. Each object is put into the scene at an arbitray position. Using histogram backprojection, the position of an object is hypothesized. The camera parameters are changed in such a way that a close–up view of the object is captured; this requires camera calibration and 3–D information. These images are segmented into color regions by an extended split–and–merge algorithm [2]. Features of these regions are compared to a knowledge base represented as a semantic network to verify the class of the object. As there may be an arbitrary delay between presentation of the object and search for the object in the scene, illumination may change. Therefore we normalize colors prior to backprojection.

In the experiments in Sect. 4 we additionally use different cameras for scenes and objects. This introduces another reason for color normalization.

3 Review of Color Normalization Algorithms

For a thorough account on color normalization — or color constancy — we refer the reader to the www.[3] In the following we restrict our attention to five selected methods which we tested in our experiments for object localization.

3.1 Color Rotation in RGB

Color rotation [7, 8] starts with color cluster analysis of a color image $[\boldsymbol{f}_{ij}]_{1\leq i\leq N, 1\leq j<M}$. In the following we simply write $\boldsymbol{f}$ for pixels $\boldsymbol{f}_{ij}$. First, we compute the cluster center of all pixels

[3] `http://wad.www.media.mit.edu/people/wad/color/constancy.html` (last checked 21/07/98).

$\boldsymbol{f}$ by $\boldsymbol{m} = E[\boldsymbol{f}]$ which is the vector pointing to the center of gravity. Let $\boldsymbol{C}$ be the (3×3)–covariance–matrix defined by $\boldsymbol{C} = E\left[(\boldsymbol{f}-\boldsymbol{m})(\boldsymbol{f}-\boldsymbol{m})^{\mathrm{T}}\right]$ whose eigenvalues $\lambda_1, \lambda_2, \lambda_3$ and eigenvectors are simply computed directly. Denote the eigenvector of the largest eigenvalue λ_1 by $\boldsymbol{v}$.

According to the *gray world assumption* (see e.g. the URL in footnote 3) it is assumed that the color in each sensor channel averages to gray over the entire image. If it is not so, we wish to rotate the cluster to the main diagonal (Fig. 1). To this aim we find the normal $\boldsymbol{n}'$ through the origin on the plane defined by the main diagonal in the RGB–cube and the principal component of the cluster whose direction is given by the eigenvector belonging to the greatest eigenvalue of the covariance matrix $\boldsymbol{C}$. The image cluster is translated to the origin, then rotated with ϕ' using $\boldsymbol{n}'$ as rotation axis – see Fig. 1 – and then pushed back to the middle of the cube along the diagonal axis. No rotation is performed if ϕ' is less than a threshold, say 1 degree. The overflows above 255 and the underflows under 0 are clipped to 255 and 0, respectively.

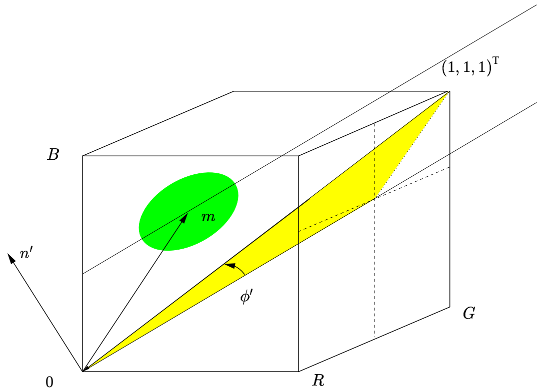

Fig. 1: Color rotation in RGB

3.2 Color Rotation in Other Color Spaces

Pomierski and Gross [9] propose to use an artificial neural network (ANN) to compute principal components of color clusters. The color space used in this work is (RG, BY, WB) (red–green, blue–yellow, white–black) which is motivated by neuro–physiology. The color space transformations from a color vector $\boldsymbol{f}$ in RGB to a vector $\tilde{\boldsymbol{f}} = \boldsymbol{A}\boldsymbol{f}$ in RG, BY, WB use the matrix $\boldsymbol{A}$.[4] Clearly the new covariance matrix will be $\widetilde{\boldsymbol{C}} = \boldsymbol{A}\boldsymbol{C}\boldsymbol{A}^{\mathrm{T}}$. The mean vector is $\widetilde{\boldsymbol{m}} = \boldsymbol{A}\boldsymbol{m}$. After finding the principal component $\tilde{\boldsymbol{v}}$ of the color cluster in (RG, BY, WB), the cluster is rotated so that this vector points to the WB direction of the $(RG, BY, WB)-$cube. We can apply the same formulas as in Sect. 3.1 to create a rotation matrix $\widetilde{\boldsymbol{R}}$ which rotates $\tilde{\boldsymbol{v}}$ to the WB–axis in (RG, BY, WB). The normalized color vectors $\boldsymbol{f}'$ are then computed entirely in RGB by $\boldsymbol{f}' = \boldsymbol{A}^{-1}\widetilde{\boldsymbol{R}}\boldsymbol{A}(\boldsymbol{f}-\boldsymbol{m}) + (128, 128, 128)^{\mathrm{T}}$.

$$\boldsymbol{A} = \begin{pmatrix} 0.5 & -1 & 0.5 \\ -0.875 & 0 & 0.875 \\ 1.5 & 1.5 & 1.5 \end{pmatrix}$$

This algorithm can be applied in any other color space which is a linear invertible transformation of the RGB space, such as XYZ or YUV. If we choose the matrix $\boldsymbol{A} = \boldsymbol{Id}_3$, the formula also applies to the RGB case in Sect. 3.1. Here $\boldsymbol{Id}_3$ is the identity matrix.

3.3 Whitening

In Fukunaga [4] the *whitening transform* is introduced, which is an orthonormal transform mapping the principal components of a cluster into the (orthogonal) eigenvectors. At the same time a scaling is done with $\frac{1}{\sqrt{\lambda_i}}$ in the direction of each corresponding axis. We apply a modified version of this approach to color normalization. We first perform the preliminary clustering steps

[4] Notation: all quantities in a color space other than RGB are marked with a tilde; all normalized values are marked with a prime.

as described in Sect. 3.1 and then – keeping the notation of the previous subsection – compute the eigenvector matrix $\boldsymbol{V}$ of $\boldsymbol{C}$, and denote $\boldsymbol{\Lambda} = \frac{255}{\sqrt{\lambda_1}}\boldsymbol{Id}_3$ where λ_1 is the greatest eigenvalue of $\boldsymbol{C}$. We note that here we modified the original transform not wanting to scale each principal component with the corresponding fraction involving its eigenvalue, as this would change the shape of the cluster more than it is desirable. For each pixel $\boldsymbol{f}$, let us form $\boldsymbol{f}' = \boldsymbol{\Lambda}\boldsymbol{V}^{\mathrm{T}}(\boldsymbol{f} - \boldsymbol{m})$. Rotate the cluster along the B axis by 45 degrees in the positive direction, and then rotate the color vectors by 45 digrees around the axis $\begin{pmatrix} 2^{-0.5} \\ -2^{-0.5} \\ 0 \end{pmatrix}^{\mathrm{T}}$, and shift the cluster along the main axis of the RGB-cube by $(128, 128, 128)^{\mathrm{T}}$. After clipping the values by 255 (so that they should not point outside the RGB-cube) we get the result. The result again is a color image which has a normalized color distribution; the mean of the color vectors is on the main diagonal of the RGB–cube; the first principal component of the cluster is on the same diagonal.

3.4 Comprehensive Color Normalization

The so called comprehensive color image normalization (CCN) presented in [3] is shown to increase localization and object classification results in combination with color indexing. The idea of this iterative two–stage algorithm is to normalize each color pixel first, using the rgb color space in which the intensity is normalized; second, each color channel is normalized separately so that the sum of the color components is equal to one third of the number of pixels. These two steps are repeated, until no more changes occur. Since global intensity changes are eliminated from the normalized images, and since the number of color values may be reduced considerably, the images may look less natural to the observer.

To put that formally: we normalize the color image $\boldsymbol{f}^{(t)} = [\boldsymbol{f}_{ij}^{(t)}]_{i=1...N, j=1...M}$ which consists of color vectors $\boldsymbol{f}_{ij}^{(t)} = \left(r_{ij}^{(t)}, g_{ij}^{(t)}, b_{ij}^{(t)}\right)^{\mathrm{T}}$. First we compute $S_{ij} := r_{ij}^{(t)} + g_{ij}^{(t)} + b_{ij}^{(t)}$ and from that $r_{ij}^{(t+1)} = r_{ij}^{(t)}/S_{ij}$, $g_{ij}^{(t+1)} = g_{ij}^{(t)}/S_{ij}$, and $b_{ij}^{(t+1)} = b_{ij}^{(t)}/S_{ij}$. In the second step we compute $r' = \frac{3}{N*M} * \sum_{i=1}^{N}\sum_{j=1}^{M} r_{ij}^{(t)}$, and normalize $r_{ij}^{(t+2)} = r_{ij}^{(t+1)}/r'$ and do the same thing for b' and g'. We repeat these two steps until the changes between iteration t and $t+2$ are less than some threshold. After CCN, the components of each pixel add up to one. We slightly modify and extend the original idea as we recall the input *intensity* values and do a histogram equalization on them. For each output pixel, we use the color computed by CCN and the normalized intensity from the input image. If we project each pixel to the plane $r + g + b = 1$, we get the CCN result.

4 Experiments

In the final version of the testbed we used 15 objects of different colors and arranged them on a table. We recorded 15 images (see Fig. 2) with the same technique as in Sect. 2, each showing one object (see Fig. 3, 4). Experiments showed that the greater the variation of colors is in the scene, the better the object is normalized. We then recorded 19 scenes using different cameras, different sizes, and different lighting, and annotated the object positions in each image manually. For each object and for each scene, the following steps were performed:

- No normalization: the object histogram was computed from the masked image and the object was searched by histogram–backprojection

- Color normalization using the algorithms described in Sect. 3:
  1. the object arrangement was color–normalized,
  2. the object histogram was computed from the masked normalized image,
  3. the scene image was color–normalized with the same algorithm as in the first step,
  4. the object was searched by histogram–backprojection.

For each search, backprojection was performed using a $64 \times 64 \times 64$ RGB or a 32×32 UV histogram. The resulting positions were compared to the manually annotated positions. The overal number of experiments thus was over 4000. An example of a scene and the applied normalization algorithms are shown in Fig. 5–10. Results can be found in Tab. 1.

5 Conclusion

In [5] it is concluded that color constancy algorithms do not fulfil the high expectations many experts hoped for. We claim

- that color normalization can facilitate more reliable object localization under changing lighting condition,
- the best choice for the proposed normalization algorithms, however, depends on the object to be localized, but not on the scene,
- using a different camera for taking the object and the scene may cause problems which color normalization usually cannot solve,
- the normalization methods show certain robustness regarding orientation and partial occlusion of objects,
- with histogram backprojection the UV space performed better than RGB,
- the fact that the success rate without normalization was comparable to the rates using the various color normalization methods shows that further research is welcome.

Further work will use histogram intersection as well as backprojection. In the context of the complete system described in Sect. 2, the best normalization algorithm for a given object will be selected by the control algorithm which analyses an image using a knowledge base.

We note that the images of the objects and scenes as well as some accompanying tables regarding the experiments can be found at our URL given on the title page.

References

[1] H. Burkhard and B. Neumann, editors. *Computer Vision — ECCV '98*, number 1406 in Lecture Notes in Computer Science, Heidelberg, 1998. Springer.

[2] J. Denzler, B. Heigl, and D. Paulus. Farbsegmentierung für aktives Sehen. In V. Rehrmann, editor, *Erster Workshop Farbbildverarbeitung*, volume 15 of *Fachberichte Informatik*, pages 9–12, Universität Koblenz–Landau, 1995.

[3] G.D. Finlayson, B. Schiele, and J.L. Crowley. Comprehensive colour image normalization. In Burkhard and Neumann [1], pages I/475–490.

	total	no norm (RGB)	no norm (UV)	Sect. 3.1 (RGB)	Sect. 3.1 (UV)	Sect. 3.4 (RGB)	Sect. 3.4 (UV)	Sect. 3.3 (RGB)	Sect. 3.3 (UV)	(1) (RGB)	(1) (UV)	(2) (RGB)	(2) (UV)	found
book	15	0	5	0	4	1	4	0	0	0	5	0	5	9
notepad	11	0	0	4	0	0	0	0	0	1	0	0	0	4
toy dog	15	0	5	0	3	2	9	0	1	0	4	0	3	12
adhesive tape	14	0	1	0	1	0	0	0	0	0	1	0	0	2
pencil	15	0	3	0	4	0	1	3	3	0	3	0	3	5
bottle	15	0	1	0	2	0	1	0	0	0	1	0	1	3
tape container	12	1	4	1	3	1	4	0	1	1	4	1	4	4
shoe	15	1	0	1	0	0	0	0	1	0	0	0	0	1
paper-clipper	15	0	0	0	0	0	0	0	0	0	0	0	0	0
mousepad	15	0	2	0	1	0	1	0	2	0	2	0	2	3
tennis ball	14	1	11	3	13	1	6	1	1	0	9	0	9	14
yellow dish	15	2	5	2	5	1	5	1	1	2	5	2	5	6
Bart Simpson	15	1	4	0	3	0	1	1	1	1	3	1	3	5
puncher	15	0	3	2	6	0	5	0	5	0	3	0	3	7
red-blue tape cont.	12	0	1	0	1	0	0	0	2	0	2	0	2	4

Tab. 1: Results for object localization. Column (1,2) is computed by the algorithms in Sect. 3.2; (1) uses the color transform $\boldsymbol{A}$; (2) converts to XYZ color space. The column "total" reflects the number of occurences in scenes; "found" is the number of scenes in which the object was found by at least one method.

[4] K. Fukunaga. *Introduction to Statistical Pattern Recognition*. Academic Press, Boston, 1990.

[5] B. Funt, K. Barnard, and L. Martin. Is machine colour constancy good enough? In Burkhard and Neumann [1], pages I/445–459.

[6] D. Paulus, U. Ahlrichs, B. Heigl, and H. Niemann. Wissensbasierte aktive Szenenanalyse. In P. Levi, editor, *Mustererkennung 1998*, Heidelberg, September 1998. Springer. accepted.

[7] D. Paulus and L. Csink. On color normalization. In T. Szirányi and J. Berke, editors, *Magyar Képfeldolgozók és Alakfelismerők Országos Konferenciája, Konferenciakiadvány*, pages 222–229, Keszthely, 1997.

[8] D. Paulus, L. Csink, and H. Niemann. Color cluster rotation. In *Proceedings of the International Conference on Image Processing (ICIP)*, Chicago, October 1998. IEEE Computer Society Press. accepted.

[9] T. Pomierski and H.M. Groß. Verfahren zur empfindungsgemäßen Farbumstimmung. In G. Sagerer, S. Posch, and F. Kummert, editors, *Mustererkennung 1995*, pages 473–480, Berlin, September 1995. Springer.

Fig. 2: Example of a scene

Fig. 3: Object arrangement

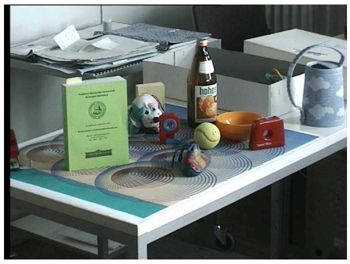

Fig. 4: Object added to Fig. 3

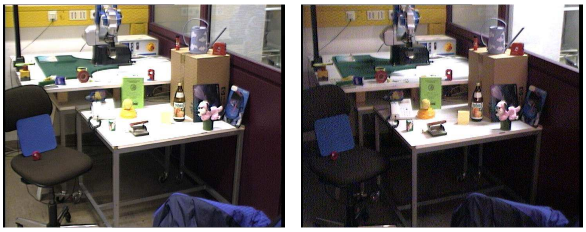

Fig. 5: Original images

Fig. 6: Algorithm of Sect. 3.4

Fig. 7: Algorithm of Sect. 3.3

Fig. 8: Algorithm of Sect. 3.1

Fig. 9: Alg. of Sect. 3.1 with RG, BY, WB

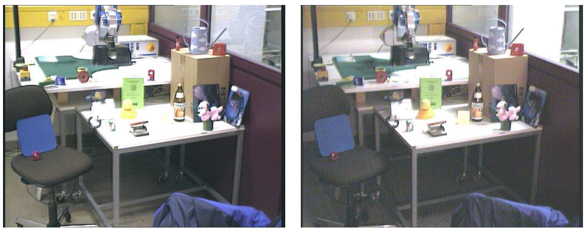

Fig. 10: Alg. of Sect. 3.1 with XYZ