

Costume: A New Feature for Automatic Video Content Indexing

Gaël Jaffré

Philippe Joly

IRIT - Université Paul Sabatier
118 route de Narbonne
31062 Toulouse Cedex 4
France

Abstract. This paper deals with the introduction of costume as a new feature for automatic video content indexing. We present in this paper an application of person recognition using costumes, in order to show the relevance of costume for indexation. The recognition is carried out by extracting the costume of all the persons who appear in the video. Then, their reappearance in subsequent frames is performed by searching the reappearance of their costume. The human presence is detected by searching faces, and then a feature for costume is extracted according to the scale and the location of each face. The Bhattacharyya coefficient, which is a coefficient derived from the Bayes error, is used to compare the color distribution of the various costumes. Finally, some results of people recognition are presented, as well as different axes for further research.

1 Introduction

Our framework is the extraction of audio-visual production parameters, as lightning, framing, costumes, . . . and its interest for video indexing and content description. Among these parameters, many are already studied [1], but costume was never considered as a classification feature for content indexing. Although the use of costume improves the robustness for tracking in a video shot [2–4], it has not been yet employed for detecting a reappearance of a character in subsequent shots. We can note that [5, p. 99] and [6] consider costume as an entity for an audiovisual production description scheme (compliant with MPEG-7 tools [7] in case of [6]), but only from a theoretical point of view, without automatic detection [5, 6] or concrete processing application [6].

Our contribution in this paper is the introduction of costume as a feature for automatic content video indexing. In order to show the interest of costume in indexing, we propose an automatic application using costumes, which allows the recognition of all the persons present in a video, and the detection of each occurrence of each character. Experiments have been carried out on video sequences of various TV programs. Experimental results show good performances and may be improved by a more accurate costume extractor in further experiments.

This paper is organized as follows. Our application is presented in section 2. In section 3, we briefly present the face detection method we used. Section 4 presents the extraction of costumes, which will be processed in section 5. Finally, section 6 presents experimental results.

2 An Application of Person Recognition using Costume

The goal of this application is to automatically detect the reappearance of a character using a model of its costume, in order to show the significance of costume as a feature for content indexing. Some experiments made on automatic video summarization showed that the costume feature is one of the most significant clue for the identification of keyframes belonging

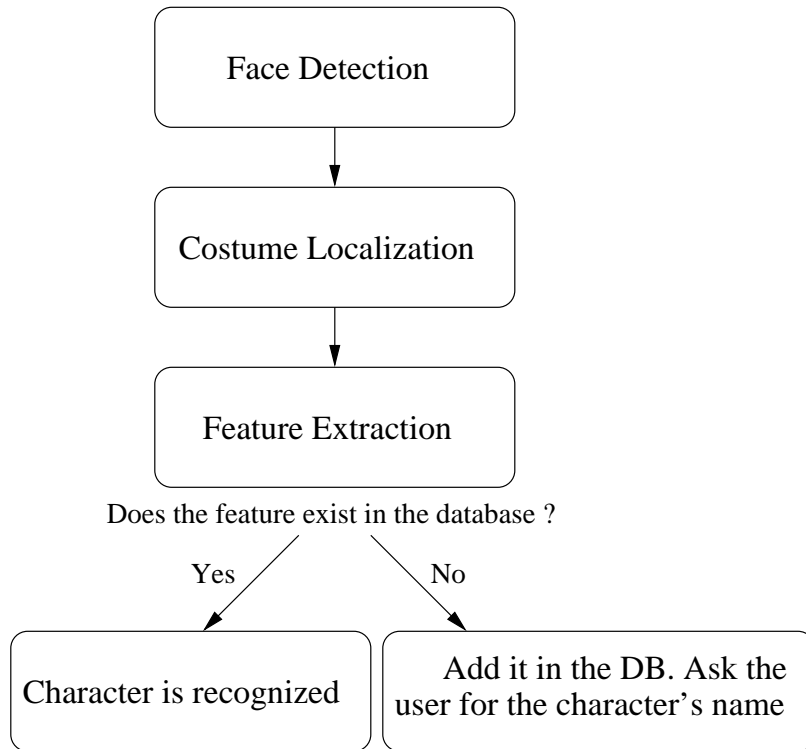


Fig. 1. Person recognition algorithm.

to some given excerpt. Authors justify this property by the fact that costumes are attached to character function in the video document [8].

The application described in this paper is semi-automatic: the first time a character appears, the user must give his name, and then the detection of reappearance becomes automatic. We could change it to an automatic one if for each new costume, we would store it without asking the user, and give the different appearances of each (anonymous) character.

Our algorithm is structured in three parts, and is applied on every frame of video sequences. We use a database of labelled costumes, which can initially be empty. First, a face detection is run, so as to detect the different possible characters who are present in the current frame, and their approximate position and scale. Then, the costume of each character is extracted from the image according to the location and the scale of his face. Finally, we compare the features extracted from the costume to those of the database. If one costume corresponds, then the character is recognized. Else, the user is asked to give the name of the character, and the new costume is added in the database. This algorithm is summed up in Fig.1.

3 Face Detection

The first step of our algorithm is to find all the faces present in each frame. In this section, we present the method used in our application.



Fig. 2. Some examples of frame by frame face localization. Each box represents a detected face. In these sequences, the false alarms only occurs in one or two single frames. This situation often occurs with noisy images. A temporal smoothing will allow to remove these false alarms, while keeping the good detections.

3.1 The Face Detection Algorithm

There are many methods for face detection in literature (see [9] for a recent review), but we do not use a specific one. We intend to make an application which is independent of the face detector, when this one is able to produce some results of at least a given minimal quality.

We used the method presented in [10] and improved in [11], because a fast implementation is available in the Intel library OpenCV [12]. We do not explain this method in details, because we only use this algorithm as a black box. If we would replace it by another one, obviously the results would differ, but the approach of our application would remain the same.

3.2 Face Detection Improvement

Our algorithm of costume localization is based upon face detection. However, frame by frame face localization introduce many false alarms, due to some noise present in the data. Only one false detection in a frame is enough to involve a false alarm on costume detection. An example is illustrated in Fig.2.

In order to reduce these false detections, we must exploit the properties of a video sequence by using a temporal approach. The use of temporal information was proposed in [13] for robust face tracking, with the CONDENSATION algorithm [14] for prediction over time. As our problem is not face tracking in a shot, we do not use the same approach: we propose a “smoothing” over time of face detection by using a temporal window.

For each frame, we detect all the faces using a static approach (at the moment we use the algorithm proposed in [11]). Then, we take a temporal window (subsequence) of $2N + 1$ frames. For each candidate face, we count its number of occurrences in the N previous frames, and in the N next frames. Recall that all these detections are made independently. Then, we keep a candidate face if it appears at least N_2 times in this subsequence. In our application, we took $N = 2$ (which leads to a subsequence of 5 frames) and $N_2 = 4$.

We consider that two detected faces correspond to the same face if there are roughly at the same location. The position parameters may slightly vary considering camera works or character motions. So, a small variation of these parameters is borne to take into account these effects. Moreover, to avoid the detection of faces in dissolves, as presented in Fig.3, we consider that two faces correspond to the same face if the costumes detected from these faces are also identical (in terms of features). An example of results is given in Fig.4.



Fig. 3. A dissolve example. In the middle frame a face is detected, which involves the extraction of a very noisy costume.



Fig. 4. Face detection improvement. Upper sequence represents the frame by frame face localization, whereas the bottom sequence is the improved face detection. We can note that faces are not detected in the dissolve, while the three main characters are all detected at least one time in the sequence.

4 Costume Localization

Our indexation is carried out by localizing the costume of detected persons. When a unknown costume is found, we have to store it in a database, in order to compare it with subsequent detected costumes.

4.1 Set of Costumes

In order to store the various models, we use a database of costumes. Each one is represented by a labelled image. When a unknown costume is found, it is added in the database, and the user is asked for the corresponding label. An example of database is given in Fig.5.

4.2 Face-Based Costume Extraction

The costumes are extracted according to the localization and the scale of the detected faces. At the moment, we estimate the costume by the area under the face. The size of this area is proportional to the one of the face. In our examples, we used a width size of 2.3 times the one of the face, and for the height size a ratio of 2.6. Examples are given in Fig.6. We chose these coefficients by taking the ones which give the best fitting of the box in our learning images.

4.3 Blind Costume Extraction

Basing costume detection upon face localization presents some drawbacks that we would like to avoid. In particular, even if the face detection is robustified (cf. section 3.2), there are many



Fig. 5. A small example of a costume database. Several models can correspond to the same character; this can be used to robustify the models. One can note that the ratio between height and width can change: this is due to the fact that some costumes are on a border of the frame, and hence are truncated.



Fig. 6. Examples of costume localization. The costumes are in the boxes under the localized faces. Their size is computed according to the scale of the corresponding face.

frames where the face is occluded, or where the person is shot from behind, and thus will not be detected.

So as to deal with these cases, we also used the localization method proposed in [15], which is not based on face localization. This method allows the detection in an image of all the objects which correspond to a color model, without a priori information about their number. First, a classification of the pixels is done: from an object model, a new image is created, where each pixel represents a membership measure to the model. This image represents the repartition of the most probable pixels to be part of the searched object. This approach consists in considering this binary image as a cluster in \mathbb{R}^2 : by using the values of the image as weights associated with each pixel location, the task of object localization reduces to the detection of local modes in the cluster. This search is carried out by applying a statistical method: the mean shift procedure [16]. Each mode is then associated with an object, which corresponds to the model (a mode is a local density maximum). This method needs a prior information about the scale of the object to search. To give up this information, we run this algorithm several times, with different scales, and keep the scale which provides the best coefficient. Moreover, we did not maximize the density estimate, as expected in [15], but the Bhattacharyya coefficient (cf. section 5.1).

However, this blind approach for costume detection needs a too long computational time. Hence, it has to be applied for each model of costume, with many different scales. Although we reduced the computational time by using a simple heuristic (only searching costumes for

which the histogram intersection [17] with the image histogram is above a threshold), this method takes more than one second per frame, which is too slow for real-time processing. In the sequel, we will detect the new occurrences of each costume using only the face-based detection, because it significantly reduces the computational time for costume search.

5 Costume Representation

5.1 Similarity Measure

When a costume is detected in the current frame, it has to be compared with the costumes which are already present in the database. At the moment, we only use the color distribution, but we want to extend it to the texture distribution.

In our experiments, we used the Bhattacharyya coefficient, which is closely related to the Bayes error [18, p. 38]. The general form (derived from the Bayes error) is

$$\rho(p, q) = \int_{\mathbb{R}^d} \sqrt{p(\mathbf{z})q(\mathbf{z})} d\mathbf{z} \quad (1)$$

where p is the color distribution of the costume found in the current frame, q the one of the costume we want to compare to, and d the dimension of the color distribution. The derivation of the Bhattacharyya coefficient from sample data involves the estimation of the densities p and q . Although color histogram is not the best nonparametric density estimate [19], we used it because it is a method of low computational cost (which is imposed by real-time processing). When the discrete densities (normalized histograms) $\hat{q} = \{\hat{q}_u\}_{u=1\dots m}$ and $\hat{p} = \{\hat{p}_u\}_{u=1\dots m}$ are computed (m is the number of bins in the color histograms), the Bhattacharyya coefficient can be estimated by [20, 21]

$$\rho(\hat{p}, \hat{q}) = \sum_{u=1}^m \sqrt{\hat{p}_u \hat{q}_u} \quad (2)$$

The coefficient interval is the real interval $[0, 1]$. A value of 1 means a perfect match, whereas a value of 0 means a mismatch.

We used other similarity measures computed from color histograms (histogram intersection [17], χ^2 , and correlation measures), but the best results are obtained with the Bhattacharyya coefficient.

5.2 Color Systems

Our images extracted from the video streams are coded using the RGB color system [22]. Although it is not perceptually uniform, this system provides satisfactory results in our framework. However, we tested other color systems, in order to see if we can have a performance improvement.

First, we tried the HSV (Hue-Saturation-Value) system. For computational time, we did not use the standard conversion formula [23], but the approximate one

$$\begin{aligned}
 V &= \max(R, G, B) \\
 S &= \frac{255 (V - \min(R, G, B))}{V} \quad \text{if } V \neq 0, \quad 0 \text{ otherwise} \\
 H &= \begin{cases} \frac{60 (G - B)}{S} & \text{if } V = R \\ 180 + \frac{60 (B - R)}{S} & \text{if } V = G \\ 240 + \frac{60 (R - G)}{S} & \text{if } V = B \end{cases} \\
 &\text{if } H < 0 \text{ then } H = H + 360
 \end{aligned}$$

We made several tests with this approximate HSV, and results are roughly the same than with the RGB one, except that computational time is slightly larger with the HSV conversion. Then, we tried to give up the brightness by using only the two components H and S. Although it should deal with the changes of brightness, the experimental results grew weaker. Various costumes were confused, and the Bhattacharyya coefficient (cf. section 5.1) has less contrasted values.

The same remarks can be made with the perceptually uniform L*a*b* system [22, p. 167]. Using the three components, the results are approximatively the same (apart from the computational time). However, when we use only two components so as to deal with illumination changes, the results become weaker. Those experiments were made essentially to remove effects of lightning variations during the recognition process. Getting back to the RGB color space induces less light variation filtering, and so may increase the false detection rate. Actually, our method is devoted to TV talk-shows indexing for now and in that kind of content, we can observe some really stable conditions of shooting with no variation of the global illumination. This is the main reason why, on that kind of content, the RGB color space provides better results than the other ones. Hence in the sequel, we only use the RGB color system.

6 Experiments

Experiments have been carried out on different video sequences extracted from TV programs, especially TV talk-shows.

6.1 Semi-Automatic Approach

In case of semi-automatic application, the user has to give the name of the new characters, i.e. the characters whose model is not yet in the database. In order to take advantage of the user intervention, we added some keywords to avoid some false alarms: the user can type "I" to ignore a detected character (his model will not be inserted in the database). This can occur for example in presence of partial occlusion, when a face is present but its corresponding body is occluded. So as to avoid multiple typing of "I", when the error is present in many successive frames, the user can also type "error": the wrong model will be added in the database, but will be ignored for subsequent processing (like percentage of appearance of each character). In order to have a graphical visualization of the results, the name of each detected character is written in the current frame, as shown in Fig.7.a.

During the various runs of the application, we measured the computational time, as well as the number of human interventions for the semi-automatic approach.



Fig. 7. Graphical results. The frame (a) is the result which we obtain when the semi-automatic approach is used, whereas the frame (b) represents the automatic approach, with automatic labels.



Fig. 8. Example of failure in recognition. In these two frames, the characters are identical, but the system needs two different models.

First, we ran it on a part of seven minutes of a TV game, which contains 10 623 frames. The frames were processed at the rate of 13 fps, on a 1.7 Ghz PC, with a C implementation, without any special optimization. In this sequence, the application succeeded in detecting the five main characters (one speaker and four candidates). The user had to type the name of the detected character 16 times, among which 8 for the audience, and one “I” (to ignore a detection). Two main characters needed two models, because of partial occlusion and change of the scale (an example is given in Fig.8). Most of the user entries were made in the first two minutes, when each character appears for the first time. Then, the number of user requirements decreases: only the audience detection and some few failures in recognition remain, when the extracted costume was already found, but is too different from the one in the database, as shown in Fig.8. Hence, we could let the application be semi-automatic during some minutes, and then afterwards let it be automatic, by ignoring the unknown new characters.

Afterwards, we tried our application on a TV detective film. The biggest problem is that many characters wear the same suit. So, the system can detect an appearance, but only of a person wearing this suit, it cannot recognize him. This case will be discussed in section 7.1. Moreover, unlike the talk-shows, the characters may change their clothes from one shot to the next. In this case, the manual intervention is needed for each new costume worn. Finally, each costume need more models than for TV talk-shows. The average number of models needed



Fig. 9. Classification of character framings.

for a costume was less than 2 for the talk-shows, but it is approximately 3 or 4 for this movie, due to different changes in lightness, contrast, indoor/outdoor sequences, ...

6.2 Automatic Approach

In order to evaluate the robustness of our application on bigger sequences, we ran the fully automatic approach on a talk-show broadcast of 31 minutes, which represents 46 428 frames. First, this video sequence was manually indexed: in each frame, we counted the number of characters, as well as their character framing. We call “character framing” the significance of the character according to his position and size in the frame. We considered three classes of framing: the first one corresponds to a character who is centered, and has a sufficient size to be the most important visual interest in the frame. The second one corresponds to characters who are important components of the frame, among several others. The third one corresponds to background characters, or characters who are not easily identifiable. Figure 9 shows an example of the different characters framings.

After execution of the application, we updated the index by giving a name for each detected character, in order to compare them with the ones of the manual index. The results are given in Table 1 and Table 2. At the end of the processing, the database contains 42 costumes. This sequence was processed at the rate of 5.33 frames per second. The recognition rate for the characters who belong to the first class is pretty good: recall this rate takes into account all the first class characters of all the frames of the whole sequence. The lack of our application for detecting second and third class characters is foreseeable, since the character detection is based on a face detection, which usually fails with those classes. We can note that computational time is bigger than the one of the test with the semi-automatic approach: this will be explained in section 6.3.

Table 1. Recognition ratio in the frame-by-frame approach

Class	Number of characters	Number of recognized characters
1	19 692	16 508 (83.83 %)
2	34 978	696 (1.99 %)
3	56 857	2 724 (4.79 %)

Table 2. Recognition errors in the frame-by-frame approach

Number of false alarms	12
Number of misclassified characters	55 (0.32%)
Number of non-detection in class 1	3 184 (16.17 %)
Number of non-detection in class 2	34 282 (98.01 %)
Number of non-detection in class 3	54 133 (95.21 %)

6.3 Shot-Based Approach

Although the recognition rate for the frame-by-frame approach is satisfactory for the first class characters, it presents some drawbacks. In a same shot, frames are quite similar, but they are all independently processed: when two frames are nearly identical, we should avoid processing by copying results. Moreover, a character who is detected in a frame can be missed in the next frame. The computational time is correlated with this deficiency: the time needed for processing a frame increases with the number of models in the database, which makes problematic the useless processing. That is why the computational time is lower in the first test, with only 15 models, compared to the previous test with 42 models.

In order to avoid this kind of miss-detections, and these useless processing, we used a shot-based approach. In the beginning of a shot, we run the classical method, but each character who is detected is supposed to appear in all the frames of the shot. Moreover, when we detect a first class character, we stop the processing until the end of the shot. Although this approach gives more false alarms, the number of miss-detections becomes weaker, and the computational time becomes lower than the real-time one. The numerical results are detailed in Tables 3, 4 and 5.

Table 3. Computational times

Frame by frame based approach	5.33 fps
Shot based approach	43.04 fps

Table 4. Recognition ratio in the shot-based approach

Class	Number of characters	Number of recognized characters
1	19 692	17 823 (90.51 %)
2	34 978	2 191 (6.26 %)
3	56 857	3 727 (6.56 %)

6.4 Limits

The face detection algorithm we use only detect faces when they have a sufficient size (greater than approximatively 40×40 pixels). It involves that costumes are not detected in global views. It would be interesting to try a face detector which allows the detection of smaller faces, so as to see whether the results are significantly improved or not.

Table 5. Recognition errors in the shot-based approach

Number of false alarms	135
Number of misclassified characters	1 488 (7.43 %)
Number of non-detection in class 1	1 869 (9.49 %)
Number of non-detection in class 2	32 787 (93.74 %)
Number of non-detection in class 3	53 130 (93.44 %)

7 Conclusion and direction of further research

7.1 Conclusion

We presented in this paper a framework for video content indexing based on costumes. We showed that costume can be used to detect the reappearance of characters in different shots, even with a simple costume model.

However, the use of costume as the only feature for indexing can produce bad results when many characters wear identical clothes. In this case, this feature can be used in addition of another feature, to achieve a reliable performance for character identification. Let’s remind here that considering the primitive task of costume detection and identification, detecting identical clothes, even worn by different characters, does make sense. Furthermore, this kind of clue can be of interest to identify roles or characters associated to a same ”corporation” in a document: for instance, in the movie “Men in Black”, all the characters who wear the same black suit belong to the same organization, in a detective film the policemen can wear an blue uniform, . . .

To evaluate this tool, we will have to propose two protocols: one to evaluate the ability of this tool to identify characters and one for costumes identification. Obviously, the goals are not the same and maybe the tool will have to be slightly optimized considering the task to be achieved. Those protocols will have also to deal some other problems such as the ground truth production: what shall be done when several persons are present on the screen at the same time? How the audience shall be considered? How recognition of characters among the audience shall be taken into account? These kinds of situations are not so rare in TV talk-shows and are problems we will have to deal with.

7.2 Direction of further research

The algorithm presented in this paper use a simple costume extractor, so as to show the interest of costume as a feature for content indexing. In subsequent works, we will focus on the robustification of our algorithm, as well as on the information brought by the costumes.

First, we want to try it with a more and a less robust face detector. For now, the detector we use detects only frontal views. We would like to combine it with a profile view detector [13]. This approach would avoid all the miss-detections due to characters who are not looking at the camera. We will also use other approaches for costume extraction. Instead of taking only the color histogram of the area under the face, we would like to use a texture distribution, in order to keep spatial informations. Moreover, we will add a weight to every pixel in the selected area [17, 21], so as to reduce the background influence. Finally, we want to separate a costume in different parts: tie, jacket, hat, trousers, . . . in order to study the interest of costume for character function, i.e. the information that costume brings to the document, and to the role of the characters.

8 Acknowledgment

This work has been conducted on the behalf of the KLIMT project.

References

1. Snoek, C., Worring, M.: Multimedia Video Indexing: A Review of the State-of-the-art. *Multimedia Tools and Applications* (2004) (to appear).
2. Koschan, A., Kang, S., Paik, J., Abidi, B., Abidi, M.: Color active shape models for tracking non-rigid objects. *Pattern Recognition Letters* **24** (2003) 1751–1765
3. Lerdsudwichai, C., Abdel-Mottaleb, M.: Algorithm for Multiple Faces Tracking. In: *IEEE International Conference on Multimedia & Expo*, Baltimore, Maryland, USA(2003)
4. Pérez, P., Hue, C., Vermaak, J., Gangnet, M.: Color-Based Probabilistic Tracking. In: *Proceedings of the 7th European Conference on Computer Vision*. Volume 1, Copenhagen, Denmark(2002) 661–675
5. Nack, F.: AUTEUR: The Application of Video Semantics and Theme Representation for Automated Film Editing. PhD thesis, Lancaster University, UK(1996)
6. Bui Thi, M.P., Joly, P.: Describing video contents: the semiotic approach. In: *Proceedings of the 2nd Content-Based Multimedia Indexing Workshop*, Brescia, Italy(2001) 259–266
7. Manjunath, B.S., Salembier, P., Sikora, T., eds.: *Introduction to MPEG 7: Multimedia Content Description Language*. Wiley-Interscience (2002)
8. Yahiaoui, I.: *Construction automatique de résumés vidéos*. PhD thesis, Télécom Paris, France (2003) in french.
9. Yang, M.H., Kriegman, D., Ahuja, N.: Detecting faces in images: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **24** (2002) 34–58
10. Viola, P., Jones, M.: Rapid Object Detection using a Boosted Cascade of Simple Features. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Kauai, Hawaii, USA(2001) 511–518
11. Lienhart, R., Maydt, J.: An Extended Set of Haar-like Features for Rapid Object Detection. In: *Proceedings of the IEEE International Conference on Image Processing*. Volume 1, Rochester, New York, USA(2002) 900–903
12. OpenCV - <http://www.intel.com/research/mrl/research/opencv/>
13. Mikolajczyk, K., Choudhury, R., Schmid, C.: Face detection in a video sequence - a temporal approach. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Volume 2, Kauai, Hawaii, USA(2001) 96–101
14. Isard, M., Blake, A.: CONDENSATION—Conditional Density Propagation for Visual Tracking. *International Journal of Computer Vision* **29** (1998) 5–28
15. Jaffré, G., Crouzil, A.: Non-Rigid Object Localization from Color Model using Mean Shift. In: *Proceedings of the IEEE International Conference on Image Processing*. Volume 3, Barcelona, Spain(2003) 317–320
16. Comaniciu, D., Meer, P.: Mean Shift: A Robust Approach Toward Feature Space Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **24** (2002) 603–619
17. Swain, M., Ballard, D.: Color Indexing. *International Journal of Computer Vision* **7** (1991) 11–32
18. Andrews, H.: *Introduction to Mathematical Techniques in Pattern Recognition*. Wiley-Interscience (1972)
19. Silverman, B.: *Density Estimation for Statistics and Data Analysis*. Chapman & Hall (1986)
20. Aherne, F., Thacker, N., Rockett, P.: The Bhattacharyya Metric as an Absolute Similarity Measure for Frequency Coded Data. *Kybernetika* **32** (1997) 1–7
21. Comaniciu, D., Ramesh, V., Meer, P.: Kernel-Based Object Tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **25** (2003) 564–577
22. Wyszecki, G., Stiles, W.S.: *Color Science: Concepts and Methods, Quantitative Data and Formulae*. Second edn. Wiley-Interscience (1982)
23. Pratt, W.: *Digital Image Processing*. Second edn. Wiley-Interscience (1991)