Image Clustering for the Exploration of Video Sequences

Vicenç Torra and Sergi Lanau
Institut d'Investicagió en Intel-ligència Artificial (IIIA-CSIC), E-08193 Bellaterra, Spain. E-mail: vtorra@iiia.csic.es

Sadaaki Miyamoto
Institute of Engineering Mechanics and Systems, University of Tsukuba, Ibaraki 305-8573, Japan.
E-mail: miyamoto@esys.tsukuba.ac.jp

In this article we present a system for the exploration of video sequences. The system, GAMBAL for the Exploration of Video Sequences (GAMBAL-EVS), segments video sequences, extracting an image for each shot, and then clusters such images and presents them in a visualization system. The system allows the user to find similarities between images and to proceed through the video sequences to find the relevant ones.

Introduction

The amount of information currently available in the Internet and in proprietary databases is increasing every day. Whereas in the past most of the stored information was textual, today there is more and more information that has a multimedia basis. In this article we consider a particular type of such multimedia data: sequences of video images.

Databases of video sequences are currently huge because of ubiquitous video cameras and invasive television. Nevertheless, gaining access to individual images and selecting relevant (or interesting) shots are still an arduous task. In fact, users are required to put a lot of effort into analyzing the images to obtain good results as a result of the sequential nature of the video sequences and the current limitations of computational systems.

Current research in the multimedia field is oriented to the application of existing data mining methods and the development of new tools for exploration and retrieval (Crestani & Pasi, 2000; Amores & Radeva, 2005).

That is, focus is on systems that extract some kind of knowledge from multimedia data and systems that help with navigation among images. In this article we describe a system for the exploration of video sequences.

At present, several systems have been developed for exploration of image databases. See, e.g., the Query by Image Content (QBIC) (QBIC, 2004) and VIPER (Müller et al., 2000) systems. Research is described by, e.g., Zhang and Zhong (1995), Sethi and Coman (1999), and Chen, Bouman, and Dalton (2000). Zhang and Zhong (1995) and Sethi and Coman (1999) base their approach on hierarchical self-organizing maps (HSOMs) (see Merkl & Rauber, 2000, for a review of such maps, and Kohonen, 1997 for the original nonhierarchical ones). In such systems, images are organized according to a two-dimensional grid structure in which each cell (neuron) in the grid contains a subset of the images. Cells located near each other in the grid contain similar images. Traversing the hierarchical structure of HSOMs, users can explore the database, focusing on those images that have greatest interest to them. In such works the structure for exploring the database is fixed (kept constant since its construction); it is not fixed in the system proposed by Chen, Bouman, and Dalton (1998), have introduced active browsing. The main idea is that the users can modify the database organization when using it. Nevertheless, the organization is still hierarchical and thus similar to hierarchical SOMs. The authors place large importance is on computational efficiency (see Chen, Bouman, & Dalton, 2000 for details). Recently, Stan and Sethi (2003) introduced a system for image exploration based on *k*-means. Again, a hierarchical structure is built from images so that the user can navigate to find the desired ones.

An alternative approach has been proposed by Rodden (2002). Instead of listing images at a particular level of the hierarchical structure, images are located in a space according to their visual similarity. That is, images that have more similarity are found in closer positions than images that have less. Such an arrangement permits a better understanding of the image database.

In this article we introduce GAMBAL[1] for the Exploration of Video Sequences (GAMBAL-EVS), a system for video sequence exploration. This system permits the user to analyze the scenes and images in a video sequence. The

Accepted May 18, 2004

 ● Published online 18 January 2006 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/asi.20311

[1]GenerAdor d'estrats per a recuperació d'inforMació BAsat en mètodes de cLassificacio.

system, based on the GAMBAL system (Lanau, 2003) (a system for exploring textual information in the Web), constructs a hierarchical structure based on a variation of the *c*-means algorithm. The system represents the images in the surface of a sphere in such a way that similar images are located closer to each other. In this way, it is easy to comprehend the structure of the images in the video sequence. Accordingly, the approach presented here combines the advantages of systems similar to those of Rodden (2002) with those of systems that build hierarchical structures.

The embedding of our exploration system in the GAMBAL environment, which includes a Web crawler and is oriented to information access on the Web, augmentation of the files the system can process with nontextual ones. In particular, video sequences (and images) can be downloaded now and later browsed by the user using the video extension described in this article.

Although this system is to be used for information access on the Web, the system requires, as most search engines do, that the Web crawler first download the files (the video sequences) before navigation because the visualization system requires that similarities between images be known, otherwise they cannot be properly displayed. Accordingly, the system does not provide dynamic access to the Web but static access.

The structure of the article is as follows. In Section 2 we give an overview of the exploration system and focus on the process of video segmentation. In Section 3 we describe in detail the clustering process. Section 4 gives some examples. The paper finishes in Section 5 with the conclusions.

GAMBAL-EVS

As stated in the Introduction, GAMBAL-EVS is a system for video sequence exploration. The system has been built on the top of GAMBAL (see Lanau, 2003, for details), a system for clustering and visualization of textual documents based on clustering techniques.

The architecture of GAMBAL-EVS is shown in Figure 1. On the one hand we have a video segmentation module that decomposes a video sequence into a set of shots. For each shot a representative image and a representative histogram are given. Shot detection is done on the basis of image histograms (i.e., comparing histograms). This is detailed in the section Video Segmentation Module.

Once the sequence is decomposed into a set of shots, an extended version of the GAMBAL system (denoted by GAMBAL* in the figure) is applied. The extension was carried out so that images can be processed and represented. In fact, GAMBAL clusters the images so that similar images are put together (in a hierarchical structure). At this point, image histograms are used to compute similarities between images. Then, GAMBAL* uses such hierarchical structures, the image histograms, and the images themselves for presenting the results to the user. In this way, the images that define the video sequence can be explored by the user.

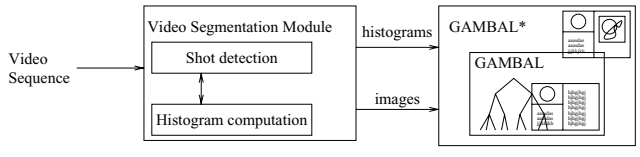

FIG. 1. The GAMBAL-EVS system for exploration of video sequences.

In this section we give the details on the video segmentation process (the video segmentation module in Figure 1). Then, in the section Clustering and Visualization our clustering approach is described in more detail.

Video Segmentation Module

Video segmentation and vector representation are based on color histograms. More precisely, the system builds a histogram for each image and uses such histograms as their numerical representatives. Segmentation is based on differences on the histograms.

Images are considered in terms of their RGB color representation. Then, to define the histogram of an image, each RGB color is reduced from 256 to 8 different possible values. Then, for a given image *im*, each pixel $p \in im$ is *counted* in the following position:

$$index(p) = indexR(p) * 64 + indexG(p) * 8 + indexB(p)$$

where $indexR(p) = floor(R(RGB(p))/32)$, $indexG(p) = floor(G(RGB(p))/32)$, and, $indexB(p) = floor(B(RGB(p))/32)$

Therefore, the histogram of image *im* corresponds to

$$h^{im}(i) = |\{p \in im | index(p) = i\}|$$

The histograms of two consecutive images (h, h') are compared by using the Kolmogorov-Smirnov test. This is:

$$ks = \max_i |CDF_h(i) - CDF_{h'}(i)|$$

where CDF_h is the cumulative distribution function of h.

This test was proposed for video segmentation by Sethi and Patel (Sethi & Patel, 1995) and compared positively by Ford, Robson, Temple, and Gerlach (2000) against other approaches based on histograms, e.g., the chi-square.

As histogram metrics produce the best results when computed for blocks rather than globally (see, e.g., Ford, Robson, Temple, & Gerlach, 2000), our system uses block-based histograms. The implementation is parametric, but to prevent computational complexity we use images of 16 blocks. Accordingly, the representative of an image *im*, h^{im}, is a set of histograms $\{h_j^{im}\}_j$ for $j < 16$. Note that as 8 different possible values are considered for each RGB color, this corresponds to block-histograms with a dimension equal to 512. Therefore, each image is represented by (vectors of) $512 \times 16 = 8{,}192$ values.

When relevant differences are found between the histograms of two consecutive images, they are considered to belong to two different shots. To determine when such a difference is relevant, a threshold θ, determined in a heuristic

DOI: 10.1002/asi

way, has been used. This process corresponds, in fact, to the detection of a shot cut.

Therefore, a shot is detected between images *im* and *im'* when

$$\max_{i} \max_{j \leq 16} |CDF_{h_j^{im}}(i) - CDF_{h_j^{im'}}(i)| > \theta \qquad (1)$$

Taking into account the process just described, a sequence can be decomposed into a set of shots. The next step is to compute representatives for each shot. Such representatives again take the form of histograms. In fact, each representative is defined as the set of histograms (i.e., one histogram for each of the 16 blocks) of the last image of the shot. Histograms are normalized per block so that within a block the values add upto 1.

Thus, the system produces a pair *(image(s), h^s)* for each shot *s*. Here, the image that represents the shot *s* is the last image in *s*, and h^s is the corresponding histogram. Naturally, that image is the one that will be displayed when the images from the video sequence are represented within the clustering system. So, whereas *image*(*s*) is the *graphical* representation of the shot, *h*(*s*) (the histogram) is the numerical representation and the one used to compute similarities/distance, between shots.

Therefore, and putting all together, we have that a video sequence *VS* defined by shots $s \in VS$ is translated into a set of pairs $\{(image(s), h^s)\}_{s \in VS}$. As all images in a shot are supposed to be similar enough, these will be the only images represented in our exploration system and the only images considered in the clustering process.

Note that of the two failures of a shot detection algorithm, false negative failures (not detecting a new shot that should be detected) are more relevant than false positive failures (detecting a shot that should not). This is so because for false positive results several representatives are computed and displayed for the same shot. Nevertheless, such representatives will probably be clustered together and finally visualized in near positions in the graphical interface of GAMBAL. In contrast, false negative results imply that only one representation is extracted for several shots. This causes some shots not to be visualized, and they are only detected if the user displays the whole sequence. As a consequence of this fact, the threshold θ in Equation (1) has been selected in such a way that false negative results are minimized, although this causes some false positive results.

Implementation Issues

GAMBAL-EVS (and GAMBAL*) is fully implemented by using the Java programming language. All image processing elements and the video segmentation module have also been implemented in Java. We have used for this purpose the Java Media Framework 2.1.1e API provided by Sun. This permits the system to be run in different platforms and allows different video formats to be considered. The system was developed and tested by using Linux (Redhat and Fedora Core). The files considered for this article followed the MPEG standard.

Clustering and Visualization

Clustering and visualization in GAMBAL-EVS rely on the GAMBAL system. Roughly speaking, the GAMBAL system builds a dendrogram (a hierarchical structure) of the data by using a clustering method and then such dendrograms are visualized in a graphical interface.

Here, the clustering process is applied to the images obtained from the video sequence, and, thus, the dendrogram is defined with images in its leaves and clusters of images in its internal nodes. Clusters are defined by putting together similar images (according to their histograms).

It has to be said that the clustering process is independent of the shot detection algorithm in what concerns the comparison of images. As detailed in localization of objects on the surface (Definition of *C*1), the Euclidean distance is used in the clustering prcoess and the Kolmogorov-Smirnov is used for shot detection. Nevertheless, other methods for computing the similarities between images could be considered. From our point of view this independence is important and meaningful because one aspect is shot detection (which is application independent) and the other is the similarity between images (which is application dependent).

The Graphical Interface

The graphical representation of the dendrogram is done by using the GAMBAL visualization system. Figure 2 gives a snapshot of the system. In the system, objects and clusters are located on the surface of a sphere. Moreover, different α-cuts of the same dendrogram (corresponding to different partitions of the elements) are represented in different concentric spheres. At the user's request, the system moves from one sphere to an adjacent one, so that the user can navigate through the hierarchy. In other words, the user can change the degree of granularity in which the objects in the hierarchy are seen. Moreover, the user can zoom or rotate the current sphere as desired.

Figure 3 illustrates the GAMBAL system. This figure represents (on the left-hand side) a dendrogram constructed by objects {*a*, *b*, *c*, *d*, . . . *m*} and (on the right-hand side) their representation on a set of concentric spheres *C*0, *C*1, *C*2, *C*3 (*circles* in this case). Each sphere (circle) represents

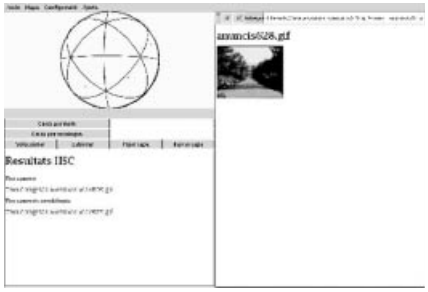

FIG. 2. Snapshot of the GAMBAL visualization system.

DOI: 10.1002/asi

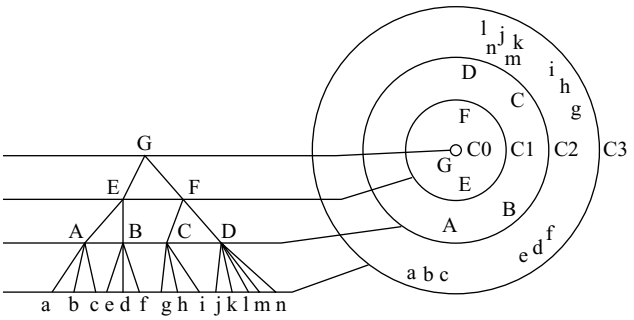

FIG. 3. A conceptualization of the GAMBAL clustering and its visualization system.

one of the α cut in the dendrogram (in the left-hand side of the figure). For example, $C2$ corresponds to the α-cut defined by $\{A, B, C, D\}$ and $C1$ to the one defined by $\{E, F\}$. Although in the figure all the spheres are displayed at once, the interface displays only one at a time. With this interface, the user can navigate through the dendrogram, changing from one sphere to another adjacent one, and this process corresponds to change from one α-cut to another one.

At the surface of the sphere, only dots are represented. Clicking a particular dot, a window of the system displays the information (e.g., the file) corresponding to the clicked point as well as of the nearest objects. In the extended version of GAMBAL, the image can be displayed as well; see Figure 2.

Although we have considered here a bottom-up description of the process (from low-level clusters to higher-level ones), the method implemented is top-down. That is, at first, a large cluster with all images is considered, and in successive steps the sets of images are further split. In this way, each splitting corresponds to a further refinement of the cluster considered. The whole process is described in the next section.

The Clustering Process

As described in the previous section, the clustering process follows a top-down design. In successive steps, clusters are split into new clusters. To bootstrap the process, we have considered an initial partition of the images. Each partition element defines a cluster. As the initial partition is defined taking into account the localization of the objects on the surface of the first sphere ($C1$ in Figure 3) the process is also described.

Note that following Figure 3 a $C0$ sphere containing a single cluster with all objects can also be defined. As such a sphere is not relevant to exploring sets of documents/images we have not implemented its visualization and in the clustering process we start directly building $C1$.

Localization of objects on the surface (definition of $C1$). Objects are located in the surface of the $C1$ sphere according to their similarities. Roughly speaking, objects are located in such a way that similar objects are located in closer positions, and dissimilar objects are located in farther positions.

In fact, this approach corresponds to multidimensional scaling. To implement this method we have adapted the Sammon's map (a method for multidimensional scaling) so that the distance computed between pairs of objects is compared with the distance of their location/position on the sphere.

To give additional details, we need some formal definitions. Let x_i denote the images, and z_i their localization on the surface of the sphere; then with $d^o(x_i, x_j)$ we denote the distance between the images x_i and x_j according to their histograms and with $d^s(z_i, z_j)$ we denote the distance according to their position on the surface (i.e., the angle that defines z_i and z_j). Given such definitions, locating all data x_i on the surface is equivalent to finding their position z_i so that the following expression is minimized:

$$\sum_{i>j} \frac{(d^o(x_i, x_j)/d^o{}_{\max} - d^s(z_i,z_j)/d^s_{\max})^2}{d^o(x_i, x_j)/d^o{}_{\max}}$$

In our application we have defined the distance between two images *im* and *im′* in terms of their (set of) histograms h and h' using the Euclidean distance as follows:

$$d^o(h, h') = \sqrt{\sum_i (h(i) - h'(i))^2}$$

At this point, other distances $d^o(h, h')$ might be considered as well. Also, it might be possible to define distance taking into account the whole image and not only the histograms. As argued in Clustering and Visualization, computing similarities between images in the exploration stage is dependent on the application: the kind of *similarities* that the user is interested in detecting.

Initial clusters. The surface of the sphere is divided into six triangular regions, all of the same size. Images are assigned to the corresponding cluster. Regions were defined as triangular because such a shape permits a homogeneous recovering of all the sphere surface (i.e., a triangularization of the surface).

Cluster splitting and new cluster formation. The process of splitting a cluster corresponds in our system to spliting a triangular region into three new (but smaller) triangular regions.

This process is achieved by using a variation of the c-means (Duda & Hart, 1973; Miyamoto & Umayahara, 2000) clustering algorithm. In fact, the variation was defined so that the new partition is consistent with the triangularization of the surface.

Accordingly, we consider an objective function based on c-means that takes into account some additional elements related to the triangular shape. In particular, the objective function considers that only three new clusters are built and that the centroids of these clusters should be *near* the neighboring triangles. So, if a triangular region $T0$ (see Figure 4) is split into regions *Ta*, *Tb,* and *Tc*, the centroids of *Ta*, *Tb,*

DOI: 10.1002/asi

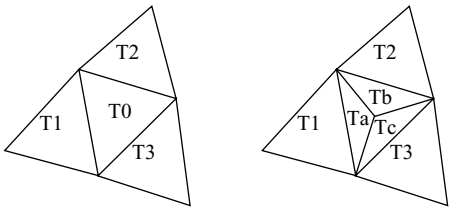

FIG. 4. Splitting a triangle into three new triangles.

and Tc (namely, v_1, v_2, and v_3) should be near the centroids of the limiting triangles $T1$, $T2$, and $T3$ (namely, a_1, a_2, and a_3).

This is expressed by means of the following objective function:

$$J(U, V) = \alpha \sum_{k=1}^{n} \sum_{i=1}^{3} u_{ik} d(x_k, v_i)^2 + (1 - \alpha) \sum_{i=1}^{3} d(v_i, a_i)^2$$

where x_k are the elements to cluster, v_i and a_i are as defined earlier, and c_{ik} is a Boolean value representing whether the x_k belongs to the cluster with centroid v_i. As it is assumed that a certain element can only belong to a single class, the matrix (c_{ik}) should belong to the set

$$M = \left\{ (u_{ik}) | u_{ik} \in \{0, 1\}, \sum_{i=1}^{3} u_{ik} = 1 \text{ for all } k \right\}$$

Here, α is assumed to be constant, and it corresponds to a selected trade-off between the usual c-means, not considering the neighbors (i.e., the result of minimizing the expression $\sum_{k=1}^{n} \sum_{i=1}^{3} u_{ik} d(x_k, v_i)^2$) and just putting the centers at the neighbor's position (i.e., the result of minimizing $\sum_{i=1}^{3} d(v_i, a_i)^2$).

As at this point we are considering the splitting of the regions on the surface, the distance d in $J(U, V)$ corresponds to the distance on the surface.

The minimization problem stated previously is solved by using the general algorithm for c-means:

Step 1: Start with an initial set $\overline{V}$.
Step 2: Solve $min_{U \in M} J(U, \overline{V})$ and let the optimal solution be $\overline{U}$.
Step 3: Solve $min_V J(\overline{U}, V)$ and let the optimal solution be $\overline{V}$.
Step 4: Repeat steps 1–3 while (U, V) is not convergent.

These steps are computed in the following way:

Step 1: $\overline{V} = (v_1, v_2, v_3)$ are defined as the average value between the center of the triangle we are splitting and the center of the corresponding neighboring triangle.
Step 2: Elements are assigned to the nearest v_i.
Step 3: To find the $\overline{V} = (\overline{v}_1, \overline{v}_2, \overline{v}_3)$ that minimizes

$$J(v) = \alpha \sum_{k=1}^{n} \sum_{i=1}^{3} u_{ik} \psi^2_{x_k, v_i} + (1 - \alpha) \sum_{i=1}^{3} \psi^2_{a_i, v_i}$$

the gradient method is used.
Accordingly, an iterative process is applied whereby at the kth step the values for v^k are computed by using the ones at the $k - 1$ step (i.e., v^{k-1}) and the gradient ∇J by using the following expression:

$$v^k = v^{k-1} - \gamma \nabla J$$

where γ is a learning rate constant.
Step 4: To detect convergence, the distance between centers at time k and $k - 1$ is checked. When such distance is less than a certain threshold, the algorithm is stopped.

Examples

The system has been applied to several video sequences. The examples reported here correspond to two advertisements and one fragment from a TV entertainment program. In Figure 5 we display two images of the advertisement sequences considered (color images in the original video sequence). The image on the left corresponds to a sequence that led to 13 different images corresponding to 13 different shots. This set is used for illustration.

Figure 6 gives a snapshot of the system when a particular image is selected. It can be observed (upper part in the left-hand side of the image) that the sphere with its triangularization and the dots corresponding to the images are represented. In the left-hand side of the image (lower part) a list of links can be observed. Such a list is obtained by clicking on the surface. In this case, the clicked object as well as nearest objects are listed. On the right-hand side of the figure, the selected image is given.

FIG. 5. Two images from the video sequences considered.

DOI: 10.1002/asi

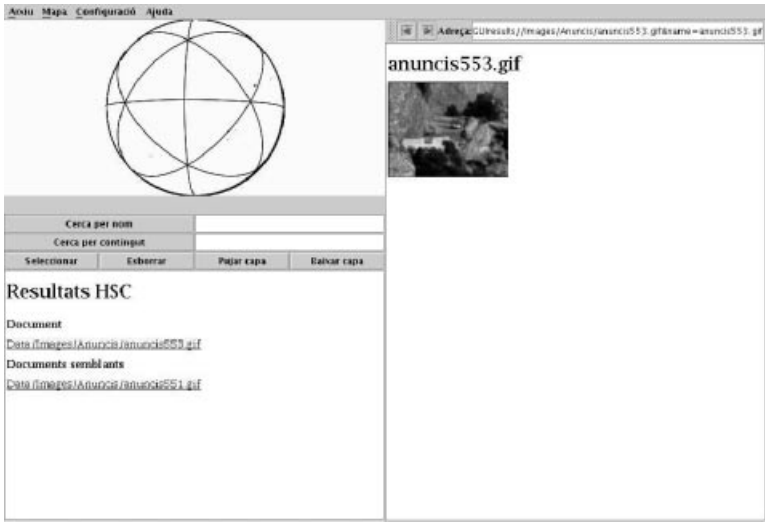

FIG. 6. Example of the clustering system.

FIG. 7. Images located in near positions by the GAMBAL-EVS system.

In Figure 7 the two similar images of Figure 6 are displayed. It can be observed that the similarities do not correspond to objects located in the same position (i.e., a car in the center of the image) but to color and texture. Note that a significant percentage of both figures contains the same *rocky mountain*. This color- and texture-based similarity is due to the system approach of computing similarity on the basis of histograms. Moreover, the segmentation process will assign to all those images from the same shot (and having objects in the same position) a single image representation. Therefore, such similar images, unless they are from different shots (as in TV news), are not duplicated.

In Figure 8, some images of the entertainment program are displayed. This sequence is 4 minutes 11.12 seconds long and its segmentation took (according to Linux *time* command) 4 minutes 13 seconds of real time, 3 minutes and 2.9 seconds of which were devoted to the user and the rest to the system. In fact, the segmentation takes place while the video is displayed (with sound) *in real time* on the screen. With a $\theta = 0.7$, 102 shots have been generated. Naturally, variations of θ lead to variations on the number of shots being generated.

In Figure 8a and Figure 8b, we show two images that have been located in near positions on the sphere (near the center of the sphere). They correspond to two contiguous shots in the original video. Then, in Figure 8c, we display another image that was considered similar but that is not contiguous to the previous ones. In Figure 8d we have another image, which has greater similarity to the previous ones. Figure 8e and Figure 8f we include two images that were located on the other side of the sphere and, thus, have more dissimilarity to Figure 8a and Figure 8b. It can be seen that the last figures are again similar to each other.

Conclusions and Future Work

In this article we have presented a tool for image clustering and visualization to help in the exploration of video sequences. We have described the clustering and visualization system and given an example that proves its interest.

As future work we plan to extend the clustering system with fuzzy clustering techniques (following, e.g., Kraft, Bordogna, & Pasi, 1999; Torra, Miyamoto, & Lanau, 2005; Miyamoto, 2003) and then to extend the approach to explore and visualize video sequences. This corresponds to finding similarities between the sequences. See, e.g., Adjeroh, Lee, and King (1999) for an example of a similarity function on sequences.

DOI: 10.1002/asi

FIG. 8. The example of the entertainment program: Images for shot representatives.

Additionally, we plan to combine GAMBAL-EVS with filtering systems (Herrera-Viedma & Peis, 2003) so that GAMBAL-EVS will permit the exploration of the results of a query or multiple queries in large databases.

Acknowledgments

This work was partly funded by the Generalitat de Catalunya (AGAUR, 2004XT 00004) and the MICYT/MEC (projects TIC2001-0633-C03-02 and SEG2004-04352-C04-02).

References

Adjeroh, D.A., Lee, M.C., & King, I. (1999). A distance measure for video sequences. Computer Vision and Image Understanding, 75, 25–45.

Amores, J., & Radeva, P. (2005). Retrieval of IVUS images using contextual information and elastic matching. International Journal of Intelligent Systems, 20, 541–559.

Chen, J.-Y., Bouman, C.A., & Dalton, J.C. (1998). Similar pyramids for browsing and organization of large image database. Proceedings of the SPIE/IS&T Conference on Human Vision and Electronic Imaging III (Vol. 3299, pp. 563–575). San Jose, CA: SPIE.

Chen, J.-Y., Bouman, C.A., & Dalton, J.C. (2000). Hierarchical browsing and search of large image databases. IEEE Transactions on Image Processing, 9(3), 442–455.

Crestani, F., & Pasi, G. (Eds.). (2000). Soft computing in information retrieval. Germany: Physica Verlag.

Duda, R., & Hart, P. (1973). Pattern classication and scene analysis. New York: John Wiley & Sons.

Ford, R.M., Robson, C., Temple, D., & Gerlach, M. (2000). Metrics for shot boundary detection in digital video sequences. Multimedia Systems, 8, 37–46.

Herrera-Viedma, E., & Peis, E., (2003). Evaluating the informative quality of documents in SGML format from judgements by means of fuzzy linguistic techniques based on computing with words. Information Processing and Management, 39, 233–249.

Kraft, D.H., Bordogna, G., & Pasi, G. (1999). Fuzzy set techniques in information retrieval. In J.C. Bezdek, D. Didier, & H. Prade (Eds.), Fuzzy sets in approximate reasoning and information systems: Vol. 3. The handbook of fuzzy sets series. Norwell, MA: Kluwer Academic.

Lanau, S. (2003). Clasifición y visualización de datos complejos. Unpublished MS. thesis, Universitat Autònoma de Barcelona, Spain.

Merkl, D., & Rauber, A. (2000). Document classification with unsupervised neural networks. In F. Crestani & G. Pasi (Eds.), Soft computing in information retrieval (pp. 102–121). Berlin, Germany: Physica Verlag.

DOI: 10.1002/asi

Miyamoto, S. (2003). Information clustering based on fuzzy multisets. Information Processing and Management, 39, 195–213.

Miyamoto, S., & Umayahara, K. (2000). Methods in hard and fuzzy clustering. In Z.-Q. Liu & S. Miyamoto (Eds.), Soft computing and human-centered machines (pp. 85–129). Tokyo: Springer-Tokyo.

Müller, W., Müller, H., Marchand-Maillet, S., Pun, R., Squire, D.M., Pecenovic, Z., et al. (2000). MRML: An extensible communication protocol for interoperability and benchmarking of multimedia information retrieval systems. In SPIE Photonics East—Voice, Video, and Data Communications (pp. 961–968). Boston: SPIE.

Nagasaka, A., & Tanaka, Y. (1992). Automatic video indexing and full video search for object appearances. Proceedings of the IFIP TC2/WG2.6 Second Working Conference on Visual Database Systems (Visual Database System II) (pp. 113–127).

QBIC(™). (2004). IBM's Query By Image Content. See http://wwwqbic.almaden.ibm.com/ and its application at http://www.hermitagemuseum.org/html En/07/hm7 41 1.html

Rodden, K. (2002). Evaluating similarity-based visualizations as interfaces for image browsing. (Tech. Rep. No. 543, UCAM-CL-TR-543, ISSN 1476-2986). Cambridge, England: University of Cambridge, Computer Laboratory.

Sethi, I.K., & Coman, I. (1999). Image retrieval using hierarchical self-organizing feature maps. Pattern Recognition Letters, 20, 1337–1345.

Sethi, I.K., & Patel, N. (1995). A statistical approach to scene change detection. Proceedings of Storage and Retrieval for Image and Video Databases III (Vol. 2420, pp. 329–338). San Jose, CA: SPIE.

Stan, D., & Sethi, I.K. (2003). eID: A system for exploration of image databases. Information Processing and Management, 39, 335–361.

Torra, V., & Miyamoto, S. (2002). Hierarchical spherical clustering. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 10(2), 157–172.

Torra, V., Miyamoto, S., & Lanau, S. (2005). Exploration of textual databases using a fuzzy hierarchical clustering algorithm in the GAMBAL system. Information Processing and Management, 41(3), 587–598.

Zhang, H., & Zhong, D. (1995). A scheme for visual feature based image indexing. Proceedings of SPIE/IS&T Conference on Storage and Retrieval for Image and Video Databases III (Vol. 2420, pp. 36–46). San Diego, CA: SPIE.

DOI: 10.1002/asi