Spectral Structuring of Home Videos

Jean-Marc Odobez, Daniel Gatica-Perez, and Mael Guillemot *

IDIAP, Martigny, Switzerland
{odobez,gatica,guillemo}@idiap.ch

Abstract. Accessing and organizing home videos present technical challenges due to their unrestricted content and lack of storyline. In this paper, we propose a spectral method to group video shots into scenes based on their visual similarity and temporal relations. Spectral methods have been shown to be effective in capturing perceptual organization features. In particular, we investigate the problem of automatic model selection, which is currently an open research issue for spectral methods, and propose measures to assess the validity of a grouping result. The methodology is used to group scenes from a six-hour home video database, and is assessed with respect to a ground-truth generated by multiple people. The results indicate the validity of the proposed approach, both compared to existing techniques as well as the human ground-truth.

1 Introduction

The development of efficient browsing and retrieval techniques for home video is of great importance for video albuming and other multimedia applications [4, 5, 3], but represents a technical challenge due to the unrestricted content and the absence of storyline in consumer videos. These videos are composed of a set of scenes, each composed of few shots, visually consistent, localized in time, and randomly recorded, making them unsuitable for analysis approaches based on storyline models. However, recent studies have shown that people implicitly follow certain rules of attention focusing and recording [5, 3], and that the scene structure of home video can be disclosed from such rules [3].

At the same time, there is an increasing interest in computer vision and machine learning towards spectral clustering methods [11, 12, 14, 6], which aim at partitioning a graph based on the eigenvectors of its pairwise similarity matrix. These methods have provided some of the best known results for image segmentation and data clustering. However, the automatic determination of the number of clusters has not been fully addressed in most of these references.

In this paper, we propose a methodology to discover the cluster structure in home videos using spectral algorithms. Our paper has two contributions. In the first place, we present a novel analysis related to the problem of model selection in spectral clustering for the algorithm of [7], and study some measures to assess

* The authors thank the Swiss National Foundation for supporting this work through the National Center of Competence in Research (NCCR) on "Interactive Multimodal Information Management (IM)2". The authors also thank the Eastman Kodak Company for providing the Home Video database, and N. Triroj (University of Washington) for providing the multiple-subject third-party ground-truth.

the quality of a partition, discussing the balance between the number of clusters and the clustering quality. In particular, we discuss the use of the eigengap, a measure referred to as a potential tool for clustering evaluation [7], but for which we are not aware of any experimental studies showing its usefulness in practice. In the second place, we show that the application of spectral methods to home video structuring results in a powerful method, despite the use of simple features of visual similarity and temporal relations. The methodology shows good performance with respect to cluster detection and individual shot-cluster assignment, both compared to existing techniques and to people performing the same task, when evaluated on a six-hour home video database for which a third-party ground-truth generated by multiple subjects is available.

The paper is organized as follows. Section 2 describes the spectral clustering algorithm, discussing the model selection issue and the use of various clustering quality measures. Section 3 describes the application of the methodology to structuring of home videos. Section 4 describes the database and the performance measures, and presents results. Section 5 provides some concluding remarks.

2 The spectral clustering algorithm

First, we briefly describe the spectral algorithm (proposed in [7] and inspired by [12, 11]). We then analyze it for both ideal and general cases. Model selection is then discussed, and several measures of assessing clustering quality are presented.

2.1 The algorithm

Let us define a graph $\mathcal{G}$ by (S, A), where S denotes the set of nodes, and A is the affinity matrix encoding the similarity between any two nodes in the set S. We ensure that $A_{ii} = 0$ for all i in S. The affinity A_{ij} is often defined as :

$$A_{ij} = \exp^{-\frac{d^2(i,j)}{2\sigma^2}} , \tag{1}$$

where $d(i, j)$ denotes a distance measure between two nodes, and σ is a scale parameter. The algorithm consists of the following steps :

1. Define $D(A)$ to be the degree matrix of A (i.e. a diagonal matrix such that $D_{ii} = \sum_j A_{ij}$), and construct $L(A)$ by $L(A) = (D(A))^{-1/2} A (D(A))^{-1/2}$.
2. Find $\{x_1, x_2, \ldots, x_k\}$ the k largest eigenvectors of L (chosen to be mutually orthogonal in the case of repeated eigenvalues), and form the matrix $X = [x_1 x_2 \ldots x_k]$ by stacking the eigenvectors in columns.
3. Form the matrix Y from X by renormalizing each row to have unit length. The row Y_i is to the new feature associated with node i.
4. Cluster the rows Y_i into k clusters via K-means.
5. Assign to each node i the cluster number corresponding to its row.

When the value of K corresponds to its true value, the rows of Y should cluster in K orthogonal directions. Thus, the K initial centroids $(Y_i^c)_{i=1,\ldots,K}$ in the fourth step of the algorithm can be selected by first finding the row of Y for which the $Ninit$ neighbours form the tightest cluster, and then recursively selecting the row whose inner product to the existing centroids is the smallest according to :

$$Y_{i+1}^c = \underset{Y_j}{\operatorname{argmin}} \max_{(Y_l^c)_{l=1:i}} (Y_l^c . Y_j) .$$

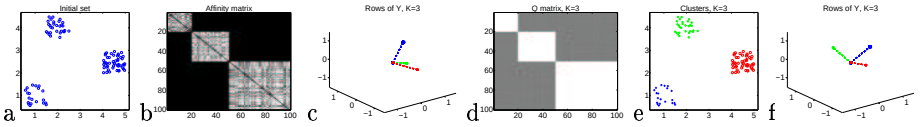


Fig. 1. Clustering example: (a) initial points; (b) affinity matrix; (c) rows of Y (in $\mathbb{R}^3$) when K=3 and eigensystem solved with *eig* from matlab; (d) Q matrix; (e) clustering result; (f) rows of Y, but with the eigensystem solved with the *eigs* function.

2.2 Algorithm analysis

Figure 1(e) and 4(b) show examples of clustering results that can be obtained with this algorithm. It was shown in [7] that the above algorithm is able to find the true clusters under the condition that K corresponds to the true number of clusters (whenever such a value exists). In this section we extend this result by analyzing the behaviour for the case when K is above or below this ideal number. Two cases are considered: the ideal case, when the true clusters are well separated; and the general case, when noise due to inter-cluster similarity exists.

The ideal case To understand the behaviour of the algorithm, we consider an ideal case in which the different clusters have infinite separation. Without loss of generality, if we additionally suppose that K_{ideal}=3, the set of all node indexes is given by $S = S_1 \cup S_2 \cup S_3$, where S_i denotes the i^{th} cluster of size n_i. We also assume that the node indexes are ordered according to their cluster. An example obeying these assumptions is illustrated in Fig. 1, where the distance employed to define the affinity between two nodes is the usual euclidian distance between the 2D coordinates, and affinity is computed by Eq. 1.

In this case, A (resp. L) is a diagonal matrix composed of 3 blocks $(A^{(ii)})_{i=1,2,3}$ (resp. $(L^{(ii)})_{i=1,2,3}$) which are the intra-cluster affinity matrices for L. It follows that (i) its eigenvalues and eigenvectors are the union of the eigenvalues and eigenvectors of its blocks $L^{(ii)}$ (the latter appropriately padded with zeros); (ii) its highest eigenvalue is unity; (iii) unity is a repeated eigenvalue of order 3; (iv) the 4th eigenvalue is stricly less than 1 (assuming $A^{(ii)}_{jk} > 0, j \neq k$), and (v) the eigenspace of the unity eigenvalue has dimension 3, and thus, the eigenvectors provided by a particular decomposition algorithm are not unique. X_3 (where X_K denotes the first K eigenvectors stacked in columns) has the form

$$X_3 = \begin{bmatrix} v_1^{(1)} & 0 & 0 \\ 0 & v_1^{(2)} & 0 \\ 0 & 0 & v_1^{(3)} \end{bmatrix} \times R\,, \qquad \text{with } R = \begin{bmatrix} \mathbf{r}_1 \\ \mathbf{r}_2 \\ \mathbf{r}_3 \end{bmatrix}\,,$$

where R is a 3×3 rotation matrix composed out of the three row vectors $\mathbf{r}_i$, and $v_l^{(j)}$ denotes the l^{th} eigenvector of the matrix $L^{(jj)}$. Thus, each row of X_3 is of the form $v_{1i}^{(j)} \times \mathbf{r}_j$, where $v_{1i}^{(j)}$ is a scalar (the i^{th} component of $v_1^{(j)}$). Therefore, after renormalizing the rows of X_3 (step 3 of the algorithm), the matrix Y has rows that fulfill $Y_i = \mathbf{r}_j \ \forall i \in S_j$. Fig. 1(c) illustrates this result for the data set of Fig. 1(a). Fig. 1(f) shows the result obtained by changing the matlab function

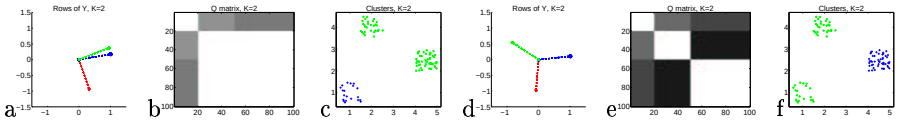


Fig. 2. Same data as in figure 1, with K=2 : (a,b,c) when the eigensystem is solved with *eig* on matlab ; (d,e,f) when using *eigs*. (a) (d) denote the rows of Y (in $\mathbb{R}^2$); (b) (e) denote the Q matrix; (c)(f) show the clustering result.

that solves the eigensystem. Note that the three vectors are still orthogonal, but have a different configuration. An alternative formulation defines $Q = Y \times Y^T$ [11]. In the ideal case, we have $Q(i,j) = 1$ for nodes i and j belonging to the same cluster, and $Q(i,j) = 0$ otherwise (see Fig. 1(d)).

Variation in the number of clusters in the ideal case We can now consider the two cases when $K \neq K_{ideal}$, which have not been previously studied in [7]:
1. For $K < K_{ideal}$, X_K corresponds to the first K columns of $X_{K_{ideal}}$. After normalization, we get a Y matrix whose entries are (in our example, with K=2) :

$$Y_j = (r_{i1}, r_{i2})/\|(r_{i1}, r_{i2})\| \doteq \mathbf{r}'_i \qquad \forall i \text{ and } \forall j \in S_i.$$

This simply corresponds to projecting and normalizing the initial orthogonal vectors $\mathbf{r}_i$ into a lower dimensional space. Since (as pointed out above) the vectors $\mathbf{r}_i$ can be in any orthogonal configuration, there is no general rule about the configuration of their projections $\mathbf{r}'_i$. As an example, Fig. 2 shows these projections in the case of the data of Fig. 1. Depending on the specific eigensystem solver, the projections and the clustering results can differ.
2. For $K > K_{ideal}$, consider for simplicity that $K = 4$. X_4 now consists of the matrix X_3 with the 4^{th} eigenvector appended as an extra column. As mentioned above, this eigenvector comes from one of the $L^{(ii)}$ matrices. More precisely $\lambda_4 = \max_i \lambda_2^{(i)}$. Assume that we choose this 4^{th} eigenvector in the first cluster,

$$X_4 = \begin{bmatrix} v_1^{(1)}\mathbf{r}_1 & v_2^{(1)} \\ v_1^{(2)}\mathbf{r}_2 & 0 \\ v_1^{(3)}\mathbf{r}_3 & 0 \end{bmatrix}.$$

After row normalization, it can be shown that $Q(i,j) = 0, \forall i \in S_k, \forall j \in S_l, k \neq l$. i.e., the true original clusters remain orthogonal to each other [8]. Furthermore, $Q(i,j) = 1, \forall (i,j) \in S_k^2, k = 2$ or 3, indicating that the second and third cluster remain unchanged. Only the first cluster is affected (divided into two parts). The same reasoning applies for higher values of K. Summarizing, if $K > K_{ideal}$, the resulting clustering corresponds to an overclustering of the ideal case (Figs. 3-4).

The general case In the ideal case, we have seen that the Q matrix should only have 0 and 1 entries when the true K is selected, and that there might be other entry values when $K \neq K_{ideal}$ (esp. $K > K_{ideal}$). Indeed, this can be

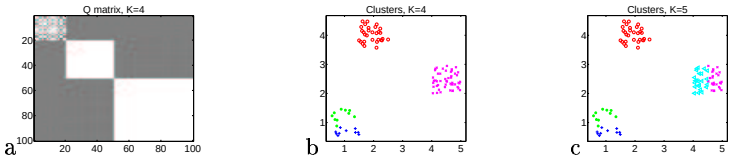


Fig. 3. Same data as in Fig. 1. (a) Q for K=4; (b) clustering; (c) clustering for K=5.

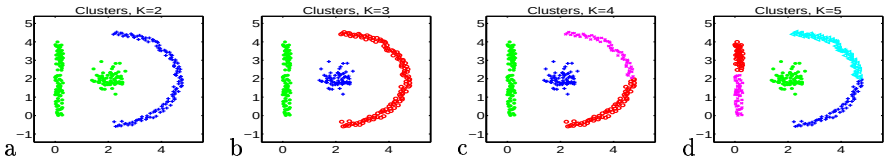


Fig. 4. Another example. Result with (a) K=2, (b) K=3, (c) K=4 (d) K=5

related to the distortion obtained at the end of the K-means algorithm :

$$MSE = \frac{1}{n} \sum_{i=1}^{K} \sum_{j \in cluster_i} \|Y_j - Y_i^c\|, \tag{2}$$

where Y_i^c represent the centroids at the end of the K-means. In the ideal case, and when $K = K_{ideal}$ the distortion should be 0. Given the correct K value, the authors in [7] use the distortion as a measure to select the clustering result from a set of results obtained by varying the scale parameter σ in the affinity matrix computation (Eq. (1)). However, there is no indication of how this measure would behave for varying values of K. Note in particular that the distortion measure is computed in spaces of different dimension (Y_j lie in $\mathbb{R}^K$), so distortion values may not be easily compared, and that the MSE may be low or not when $K < K_{ideal}$.

2.3 Automatic model selection

The selection of the "correct" number of clusters is a difficult task. We have seen in the previous Section that the analysis of the MSE measures for different K is not trivial. For this reason, we considered other criteria stemming from matrix perturbation and spectral graph theories to perform model selection.

We have adopted the following strategy. The spectral clustering algorithm is employed to provide candidate solutions (one per value of K), and the selection is performed based on the criteria discussed in the following sections.

The eigengap The eigengap is an important measure in spectral methods [6, 7] (see [1] for basic definitions). The eigengap of a matrix A is defined by $\delta(A) = 1 - \frac{\lambda_2}{\lambda_1}$ where λ_1 and λ_2 are its two largest eigenvalues [6]. In practice, the eigengap is often used to assess the stability of the first eigenvector[1]

[1] Or the first k eigenvectors, in cases where we have a k-repeated, largest eigenvalues.

of a matrix and it can be shown to be related to the *Cheeger constant* [1], a measure of the tightness of clusters. To clarify this relation, let us define the *cut value* of the partitioning $(\mathcal{I}, \bar{\mathcal{I}})$ of a graph characterized by its affinity matrix A by $Cut_A(\mathcal{I}, \bar{\mathcal{I}}) = \sum_{i \in \mathcal{I}} \sum_{j \notin \mathcal{I}} A_{ij}$, the *volume* of the subset $\mathcal{I}$ by $Vol_A(\mathcal{I}) = \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{I}} A_{ij}$, and the *conductance* ϕ of the partitioning $(\mathcal{I}, \bar{\mathcal{I}})$ by

$$\phi_A(\mathcal{I}) = \frac{Cut_A(\mathcal{I}, \bar{\mathcal{I}})}{\min(Vol_A(\mathcal{I}), Vol_A(\bar{\mathcal{I}}))}.$$

The Cheeger constant h_G is defined as $h_G(A) = \min_{\mathcal{I}} \phi_A(\mathcal{I})$ and can be shown to be bounded by the eigengap [7, 6] : $h_G(A) \geq \frac{1}{2}\delta(A)$. The conductance indicates how well $(\mathcal{I}, \bar{\mathcal{I}})$ partitions the set of nodes into two subsets, and the minimum over $\mathcal{I}$ corresponds to the best partition. Therefore, if there exist a partition for which (i) the weights A_{ij} of the graph edges across the partition are small, and (ii) each of the regions in the partition has enough volume, then the Cheeger constant will be small. Starting from $K = 1$, we would like to select the simplest clustering model (i.e., the smallest K) for which the extracted clusters are tight enough (hard to split into two subsets). This is equivalent to request that the Cheeger constant is large enough for each cluster, or to request that the eigengap is large for all clusters. Our first criterion is

$$\delta_K = \min_{i \in 1 \ldots K} \delta(L(A_K^{(ii)})), \tag{3}$$

where $A_K^{(ii)}$ are the submatrices extracted from A according to the model obtained by the spectral algorithm, and L is defined in Section 2.1. The algorithm selects the smallest K for which the eigengap δ_K exceeds a threshold.

The relative cut The measure defined by Eq. (3) has a drawback, as it only considers intra-cluster information. When part of the data have no clearly defined clusters, the algorithm may over-estimate the number of clusters so that all clusters (possibly reduced to a single element) are tight enough. We thus considered a second criterion that characterizes the overall quality of a clustering. This criterion is defined as the fraction of the total weight of edges not covered by the clusters,

$$\text{rcut}_K = \frac{\sum_{k=1}^{K} \sum_{l=1, l \neq k}^{K} \sum_{i \in S_k} \sum_{j \in S_l} A_{ij}}{\sum_i \sum_j A_{ij}}. \tag{4}$$

The algorithm outputs the largest K for which rcut is below a threshold.

3 Spectral structuring of home videos

Home videos contain series of ordered and temporally adjacent shots that can be organized in groups usually related to distinct scenes. In our approach, shot grouping exploits visual similarity and temporal adjacency, two of the main criteria that allows people to identify clusters in video collections, when nothing else is known about the content (unlike the filmmaker, who knows details of

context). Previous formulations in the literature are based on similar concepts [13]. We use spectral clustering as follows.

Home video shots usually contain more than one appearance, due to hand-held camera motion. Consequently, a shot is represented by a small fixed number of key-frames, $N_{kf} = 5$. The $i-th$ key-frame f_i of a video is characterized by a color histogram h_i in the RGB space (uniformly quantized to $8 \times 8 \times 8$ bins).

The pairwise affinity matrix A is directly built from the set of all key-frames in a video by defining

$$A_{ij} = e^{-\left(\frac{d_v^2(f_i,f_j)}{2\sigma_v^2} + \frac{d_t^2(f_i,f_j)}{2\sigma_t^2}\right)}, \tag{5}$$

where A_{ij} is the affinity between key-frames f_i and f_j, d_v and d_t are measures of visual and temporal similarity, and σ_v^2 and σ_t^2 are scale parameters.
Visual similarity is computed by the metric based on Bhattacharyya coefficient, which has proven to be robust to compare color distributions [2],

$$d_v(f_i, f_j) = (1 - \rho_{BT}(h_i, h_j))^{1/2}, \text{ with } \rho_{BT} = \sum_k (h_{ik} h_{jk})^{1/2} \tag{6}$$

Temporal similarity exploits the fact that distant shots along the temporal axis are less likely to belong to the same scene. It is defined by $d_t(f_i, f_j) = \frac{||f_j| - |f_i||}{|v|}$ where $|f_i|$ denote the absolute frame number of f_i in the video, and $|v|$ denotes the entire video clip duration. Note that the range for both d_v and d_t is $[0, 1]$.
We further set the scale parameters σ_v and σ_t in the following way. Building upon a previous study [3], we fixed the σ_v value to 0.25 which represents a good threshold for separating intra and inter-cluster similarities distributions in home videos. Similarly, it was shown in [3] that in average 70% of home video scenes are composed of four or less shots. Thus, the σ_t value was set to the average temporal separation between four shots in a given video.

The spectral method is applied as discussed in Section 2. A cluster number is assigned to each shot using a majority rule on the cluster labels of its key-frames. In the case of a tie, the cluster is randomly selected from the candidates.

4 Experiments

4.1 Data set and ground-truth

The data set consists of 20 MPEG-1 home videos, digitized from VHS tapes provided by seven different people, and with approximate individual duration of 20 minutes. The videos depict vacations, school parties, weddings, and children playing in indoor/outdoor scenarios. A ground-truth (GT) at the shot level was semi-automatically generated, resulting in a total of 430 shots. The number of shots per video varies considerably (between 4 and 62 shots).

There are two typical options to define the GT at the scene level. In the first-party approach, the GT is generated by the video creator [9]. This method incorporates specific context knowledge about the content (e.g., family links, and location relationships). In contrast, a third-party GT is defined by a subject not familiar with the content. In this case, there still exists human context

a)

	H	PHC	SM
SIE_{min}	0.078	0.156	0.116
SIE_{med}	0.275	0.362	0.271
SIE_{max}	0.535	0.532	0.539

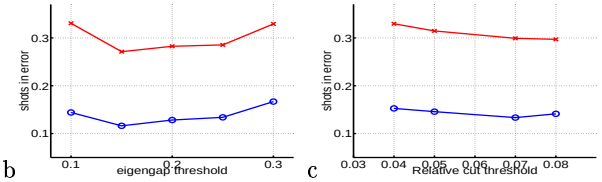


Fig. 5. (a) Average of the percentage of shots in error for humans (H), baseline (PHC), and spectral method (SM). (b)-(c) Variation of the average of percentage of shots in error (average of the median in red, of the min in blue) for different criteria as function of their threshold: (b) eigengap; (range: (0.1,0.3)); (c) relative cut (range: (0.04,0.08)).

understanding, but limited to what is displayed. This "blind context" makes third-party GTs a fairer benchmark strategy for automatic algorithms [10].

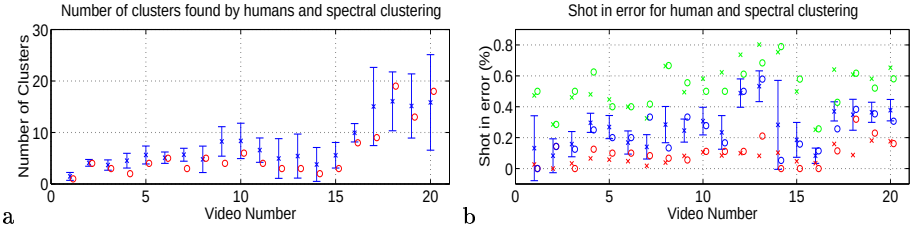
In this paper, we use a third-party GT based on multiple subject judgement that takes into account the fact that different people might generate different results. Scenes for each video were found by twenty subjects using a GUI that displayed a key-frame-based video summary (no real videos were displayed). A very general statement about the clustering task, and no initial solution were provided to the subjects at the beginning of the process. The final GT set consists of about 400 human segmentations.

4.2 Performance measures

The performance measures that we consider are (i) the number of clusters selected by the algorithm and (ii) the shots in errors (SIE). For the number of clusters, we report the value we obtain and compare it with the numbers provided by people. For shot in errors, let us denote $GT^i = \{GT^i_j, j \in 1, ..., N_i\}$ the set of human GTs for the video V_i, and $\mathcal{C}^i$ the solution of an algorithm for the same video. The SIE between $\mathcal{C}^i$ and a ground-truth GT^i_j is defined as the number of shots whose cluster label in $\mathcal{C}^i$ does not match the label in the GT. For each video, the SIEs between $\mathcal{C}_i$ and each GT^i_j are computed. Then, from the ranked SIE values, we keep the minimum, the median and the maximum, denoted SIE^i_{min}, SIE^i_{med} and SIE^i_{max} respectively. The minimum value indicates how far an automatic clustering is from the nearest segmentation provided by a human. The median value can be considered as a fair measure of how well the algorithm performs, taking into account the majority of the human GTs and excluding the largest errors. These large errors may come from outliers and are taken into account by SIE^i_{max}, which gives an idea of the spread of the measures. For the overall performance measure, we computed the average SIE measures over all the videos, of the percentage of shots in errors w.r.t. the number of shots in each video. Note that this normalization is necessary because the number of clusters (and shots) varies considerably from one video to another.

4.3 Results

The best result with our method was obtained using the eigengap criterion and a threshold $\delta_K = 0.15$. We compared it with a probabilistic hierarchical clustering



a b

Fig. 6. (a) Determination of number of clusters. (b) Percentage of shot in error. The blue bar indicates the spread of the human performances of the SIE_{med} value.

method (PHC) described in [3], as well as with human performance. The latter was obtained in the following way : for each video, the minimum, median and maximum shots in error were computed for each human GT against all the others. These values were then averaged over all subjects. These averages are plotted in Fig. 6 for each video. Finally we computed the average over all the videos to get the overall performance.
The Table of Fig. 5(a) summarizes the results. We can first notice from the minimum and maximum values that the spread of performances is high, given the performance measure. Secondly, the spectral method is performing better than PHC, as can be seen from the median and minimum value, and approximately as well as the humans.
Fig. 6 displays the results obtained for each video. First, in Fig. 6(a), we show the number of detected clusters (the red circles) as predicted by the algorithm and compare them to the mean of the number of clusters in the GT. The spread of the cluster numbers in the GT is represented by the blue bar (plus or minus one standard deviation). Note that the videos have been ordered according to their number of shots. The detected cluster numbers are in good accordance with the GT, though slightly underestimated. Fig. 6(b) displays the values of the shot in error measures in comparison to the average of human performance. The circles depict the measures obtained with our method and the crosses denote human performance. The color represents the differents measures (minimum in red, median in blue, and maximum in green). The median performance of our algorithm is better than the average human in eight cases and worse in six cases. Notice that in 25% of the cases, our algorithm provides a segmentation that also exists in the GT.

Two examples of the generated clusters are shown in Fig. 7. Each cluster is displayed as a row of shots, which in turn are represented by one keyframe. Qualitatively, the method provides sensible results.

Fig. 5 shows the obtained results using the two criteria. The selection with the eigengap criterion slightly outperforms the results obtained with the relative cut. Notice that the results are quite consistent over a relatively large range of thresholds, and better than the probabilistic hierarchical clustering algorithm.

Finally, let us mention that, given the distance matrices, the clustering algorithm implemented in matlab takes around 4 seconds per video in average.

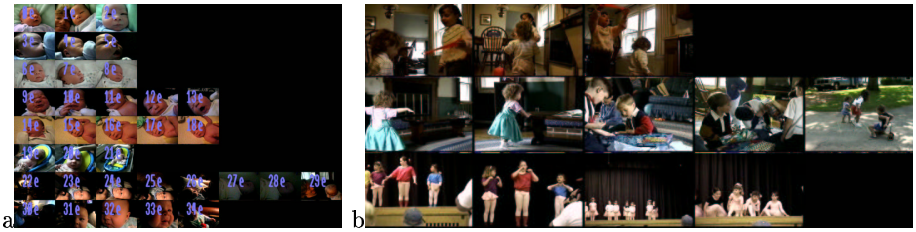


Fig. 7. Home video structuring examples. Only one keyframe per shot is displayed.

5 Conclusion

We have presented an approach for structuring home videos using a spectral method. We investigated the automatic selection of the number of clusters, which is currently an open research issue for spectral methods. We have shown in our experiments that the eigengap measure could be used to estimate this number. The algorithm was applied to a six-hour home video database, and the results are favorably compared to existing techniques as well as human performance.

References

1. F. R.K. Chung, *Spectral Graph Theory*, American Mathematical Society, 1997.
2. D. Comaniciu, V. Ramesh, and P. Meer, "Real-Time Tracking of Non-Rigid Objects using Mean Shift," in *Proc. IEEE CVPR.*, Hilton Head Island, S.C., June 2000.
3. D. Gatica-Perez, A. Loui, and M.T. Sun, "Finding Structure in Home Videos by Probabilistic Hierarchical Clustering," *IEEE Trans. on Circuits and Systems for Video Technology,* Vol. 13, No. 5, Jun. 2003.
4. G. Iyengar and A. Lippman, "Content-based browsing and edition of unstructured video," in *Proc. IEEE ICME*, New York City, Aug. 2000.
5. J.R. Kender and B.L. Yeo, "On the Structure and Analysis of Home Videos," in *Proc. ACCV*, Taipei, Jan. 2000.
6. S. Vempala R. Kannan and A. Vetta, "On clusterings - good, bad and spectral," in *Proc. 41st Symposium on the Foundation of Computer Science, FOCS*, 2000.
7. A. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: analysis and an algorithm," in *Proc. NIPS*, Vancouver, Dec 2001.
8. J.-M. Odobez, D. Gatica-Perez and M. Guillemot, "On Spectral Methods and Structuring of Home Videos," IDIAP Technical Report, IDIAP-RR-55, Nov. 2002.
9. J. Platt "AutoAlbum: Clustering Digital Photographs using Probablisitic Model Merging," in *Proc. IEEE Workshop on CBAIVL*, Hilton Head Island, S.C.,2000.
10. A. Savakis, S. Etz, and A. Loui, "Evaluation of image appeal in consumer photography," in *Proc. SPIE Conf. on Human Vision and EI*, Jan. 2000.
11. G.L. Scott and H.C. Longuet-Higgins, "Feature grouping by relocalisation of eigenvectors of the proximity matrix," in *Proc. BMVC*, 1990, pp. 103–108.
12. J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, 2000.
13. M. Yeung, B.L. Yeo, and B. Liu, "Segmentation of Video by Clustering and Graph Analysis," *Comp. Vision and Image Underst.*, Vol. 71, No. 1, pp. 94-109, July 1998.
14. Y. Weiss, "Segmentation using eigenvectors: a unifying view," in *Proc. ICCV*, 1999.