Video Summaries through Mosaic-Based Shot and Scene Clustering

Aya Aner and John R. Kender

Department of Computer Science,
Columbia University,
New York, NY 10027
{aya,jrk}@cs.columbia.edu

Abstract. We present an approach for compact video summaries that allows fast and direct access to video data. The video is segmented into shots and, in appropriate video genres, into scenes, using previously proposed methods. A new concept that supports the hierarchical representation of video is presented, and is based on physical setting and camera locations. We use mosaics to represent and cluster shots, and detect appropriate mosaics to represent scenes. In contrast to approaches to video indexing which are based on key-frames, our efficient mosaic-based scene representation allows fast clustering of scenes into **physical settings**, as well as further comparison of physical settings across videos. This enables us to detect **plots** of different episodes in situation comedies and serves as a basis for indexing whole video sequences. In sports videos where settings are not as well defined, our approach allows classifying shots for characteristic event detection. We use a novel method for mosaic comparison and create a highly compact non-temporal representation of video. This representation allows accurate comparison of scenes across different videos and serves as a basis for indexing video libraries.

1 Introduction

Advances in computer processing power, storage capacity and network bandwidth suggest that it may be possible to browse and index long collections of videos, just as search engines allow browsing over documents. However, frame, shot and scene indexing may not be sufficient at this level. People tend to recall and value videos at a higher level of organization, and this paper explores one way of departing from a strictly frame-centered view.

Most video genres can be represented by at least three levels of organization: frames, shots and scenes. A shot is a sequence of consecutive frames taken from the same camera, and many algorithms have been proposed for shot boundary detection (e.g. [11],[13],[1]). The definition of a scene is not as straightforward, and usually refers to a sequence of shots in the same physical location. There have not been as many suggestions for scene boundary detection [5][10] or, as it is also referred to, as "story units" detection [19]. Shots have been characterized to allow indexing and browsing, usually using key-frames as the basic representation

A. Heyden et al. (Eds.): ECCV 2002, LNCS 2353, pp. 388–402, 2002.

Fig. 1. (a) Hand-chosen key-frames. (Automatic key-frames generation often does not give complete spatial information). (b) Mosaic representation. Note that the whole background is visible, no occlusion by the foreground objects.

tool [18]. Indeed, we have observed that information from raw frames is usually sufficient for low-level video analysis such as shot boundary detection [1]. An example of further segmentation of video sequences into scenes is shown in [10]. In order to detect scene boundaries, a memory based model is employed, which also uses color histograms of the raw frames, and performs sufficiently accurate. However, a scene is determined, by definition, by its physical surroundings (4 to 10 cameras or camera placements are typical in a physical location) and it is usually hard to observe the background in isolated key-frames. Therefore, key-frames are not suitable for accurate comparison of long video sequences.

This work presents means to capture the highest level of a video, using a scene-based representation. It allows efficient browsing and indexing of video data, and solves the problems usually caused by a frame-based representation. We show that mosaics are more reliable for representing shots, and also—due to the concept of pan, zoom, and establishing shots—for scene representation. Unlike [7], where the mosaics were used to index frames, here we use the mosaics to spatially cluster scenes or shots, depending on the video genre. In our work we use only the background information of mosaics, since this information is the most relevant for the comparison process of shots; it is further used for shot and scene clustering, and for gathering general information about the whole video sequence. (A complete representation of a single shot also involves foreground information. Examples of single shot representations are the synopsis mosaic presented in [7] or the dynamic mosaic presented in [8], and motion trajectories shown in [3]). Mosaic comparison is not a straightforward task. However, by recognizing the special properties of different video genres, we can apply some constraints when comparing mosaics of the same video genre, and reach very accurate results.

2 Mosaic-Based Shot and Scene Representation

2.1 Motivation

Since in many video data the main interest is the interaction between the characters, the physical location of the scene is visible in all of the frames only as background. The characters usually appear in the middle area of the frame,

hence, the background information can only be retrieved from the borders of the frame. Oh and Hua [13] noticed this and segmented out the middle area; they used color information of the borderlines of the frames for shot boundary detection. However, when key-frames are used to represent shots and scenes [18] and to compare them [5], the information extracted from a single frame (or key frame) is not sufficient. Most of the physical location information is lost since most of it is concealed by the actors and only a fraction of the frame is used.

Consider the example in Figure 1(a), which shows a collection of key-frames taken from a panning shot. In that shot an actress was tracked walking across a room. The whole room is visible in that shot, but the actress appears in the middle of all key-frames, blocking significant parts of the background. This effect becomes worse in close-up shots where half of the frame is occluded by the actor. The solution to this problem is to use mosaics to represent shots. A mosaic of the panning shot discussed above is shown in Figure 1(b) . The whole room is visible and despite of camera zoom changes, the focal length changes are eliminated in the mosaic. We wish to note that our preference for mosaic representation of shots is motivated from its advantage in shot and scene comparisons, and not necessarily for complete video summarization. A good summary of a video should also include a description of its (foreground) characters.

2.2 Construction of Color Mosaics

The use of mosaics for the indexing and analysis of shots was proposed by Irani et. al. [8]. We will describe the generation of mosaics in a brief manner, and refer the reader to [8] for further details. The first step is the generation of affine transformations between successive frames in the sequence (in this work we sample 6 frames/sec). One of the frames is then chosen as the reference frame, that is, as the basis of the coordinate system (the mosaic plane will be this frame's plane). This frame is "projected" into the mosaic using the identity transformation. The rest of the transformations are mapped to this coordinate system and are used to project the frames into the mosaic plane. The value of each pixel in the mosaic is determined by the median value of all of the pixels that were mapped into it. Since we construct color mosaics, we first convert the frames to gray level images while maintaining corresponding pointers from each gray-level pixel to its original color value. For each mosaic-pixel, we form an array of all values from different frames that were mapped onto it, find the median gray value of that array, and use the corresponding color value for that pixel in the mosaic. We use outlier rejection, described in [8], to both improve the accuracy of the affine transformations constructed between frames as well as to extract all of the moving objects from the mosaic. This results in "clear" mosaics where only the background of the scene is visible, as shown in Figure 1(b).

3 Mosaic Comparison

Comparing mosaics is not a straightforward task. In video genres where a physical setting appears several times, as is the case for video genres we have tested

- situation comedies (or in short: "sitcoms") and basketball games - it is often shot from different view points and at different zooms, and sometimes also in different lighting conditions. Therefore, the mosaics generated from these shots are of different size and shape, and the same physical setting will appear different across several mosaics. Moreover, different parts of the same physical scene are visible in different mosaics, since not every shot covers the whole scene location. Thus, comparing color histograms of whole mosaics is not sufficiently accurate.

A solution to this problem is to divide the mosaic into smaller regions and to look for similarities in consecutive relative regions. When comparing mosaics generated from certain video genres, we can make assumptions about the camera viewpoint and placement, noting the horizontal nature of the mosaics. Cameras locations and movements are limited due to physical set considerations, therefore the topological order of the background is constant throughout the mosaics. Therefore, the corresponding regions only have horizontal displacements, rather than more complex perspective changes.

This work is therefore similar to that in the area of wide baseline stereo matching, which relies on detecting corresponding features or interest points. Recent work [15] has used texture features to match between corresponding segments in images, which are not only invariant to affine transformations, but also do not depend on extracting viewpoint invariant surface regions. They do, however, rely on an accurate segmentation. Our method handles significant local distortions in the mosaics, where it would have been impossible to segment these regions, or to detect lines, corners, or other features. It is also not sensitive to occluding objects, indoor lighting changes and global distortions.

3.1 Rubber Sheet Matching

We follow the idea of rubber-sheet [4] matching, which takes into account the topological distortions among the mosaics, and the rubber-sheet transformations between two mosaics of the same physical scene. The comparison process is done in a coarse-to-fine manner. Since mosaics of common physical scenes cover different parts of the scene, we first coarsely detect areas in every mosaic-pair which correspond to the same spatial area. We require that sufficient portions of the mosaics will match, therefore the width of the corresponding areas detected for a matching mosaic pair should be close to the original frame width. The height of this area should be at least $\frac{2}{3}$ of the original frame height, a reason which is motivated by cinematography rules, concerning the focus on active parts [2]. Our method enables matching of mosaics which were generated from shots taken from different locations and with different focal length. After smoothing noise in the mosaics with a 5×5 median filter (choosing color median was described in section 2.2), we divide the mosaics into relatively wide vertical strips and compare these strips. By coarsely aligning a sequence of k strips with a sequence of up to $2k$ strips, the scale between the mosaics could vary between $1:1$ and be as big as $2:1$. We use the results of the coarse stage to detect similar areas in every mosaic-pair and to crop the mosaics accordingly. By repeating the matching process with the cropped mosaics using narrower strips, we finely verify

similarities and generate final match scores for each mosaic pair. This second, finer step is necessary since global color matches might occur across different settings, but not usually in different relative locations within them.

Difference Measure and Color Space. We first explain how we define the distance measure between image regions. To address changes in lighting intensity, which cause variations along all three axes in RGB space, we adopt a color space based on hue, chrominance (saturation) and lightness (luminace), where such changes correspond only to a variation along the intensity axis. The main advantage of these color spaces is that they are closer to the human perception of colors. We chose to use HSI color pace in polar coordinates, and we compute intensity channel as luminance (instead of brightness - average of RGB). The HSI space forces non-uniform quantization when constructing histograms. However, the appropriateness of any such quantization can be easily validated by converting the quantized HSI values back to RGB space and inspecting the resulting color-quantized images. This procedure allows us to tune the quantization and to predict the results of the three-dimensional HSI histogram difference computations. After some experimentation, we concluded that uniform quantization works well for hue values, and we used 18 values. Since both saturation and intensity behave badly for small values, we applied a non-uniform quantization for both. For saturation, a threshold was empirically chosen; for pixels with saturation values below this threshold (i.e., for grays), the hue values were ignored. Saturation values above this threshold were equally divided into 5 bins. For intensity, another threshold was empirically chosen; for pixels with intensity values below this threshold (i.e., for black), saturation values were ignored. Intensity values above this threshold were equally divided into 2 bins. After determining the appropriate quantization, a simple L_1 norm between the HSI color histograms was fast to compute and performed well.

Finding Best Diagonal. All comparison stages are based on the same method of finding the best diagonal in a distance matrix, which corresponds to finding horizontal or vertical alignment. Let $D(i,j)$ be an $N \times M$ matrix where each entry represents a distance measure. We treat this matrix as a four-connected rectangular grid and search for the best diagonal path on that grid. If $P\{(s,t) \rightarrow (k,l)\}$ is a path from node (s,t) to node (k,l) of length L, then its weight is defined by the average weight of its nodes:

$$WP\{(s,t) \rightarrow (k,l)\} = \frac{1}{L} \sum_{(i,j) \in (s,t)..(k,l)} D(i,j). \tag{1}$$

We search for the best diagonal path P with the minimum weight of all diagonals, that also satisfies the following constraints:

1. $\text{Length}(P) \geq T_{length}$.

2. $\text{Slope}(P) \leq T_{slope}$.

The first constraint is determined by the width and height of the original frames in the sequence which determine the minimum mosaics' size (352×240 pixels in our experiments), since we require that sufficient portions of the mosaics will match. For example, the width of the strips in the coarse stage (discussed below) was set to 60 pixels and T_{length} was set to 5 for the horizontal alignment, so that the width of the matched region is at least 300 pixels. The second constraint relates to the different scales and angles of the generated mosaics due to different camera placements. We have found that the scale difference could be as big as $2:1$, resulting in a slope of approximately $26°$. We therefore examine diagonals of slopes that vary between $25°$ - $45°$ in both directions (allowing either the first mosaic to be wider or the second mosaic to be wider). We use intervals of $5°$, a total of 9 possible slopes. However, in order to determine the weight of each diagonal we have to interpolate the values along this straight line. Bilinear interpolation is time consuming and experiments have proved that nearest neighbor interpolation gives satisfactory results. With the use of look-up tables and index arithmetice, we were able to implement it more efficiently.

Coarse Matching. In this stage we perform coarse horizontal alignment of two consecutive strip-sequences in a mosaic-pair in order to detect a common physical area. This is achieved by generating a distance matrix $S[i,j]$, where each entry corresponds to the distance between a strip s_i from one mosaic to a strip s_j in the second mosaic:

$$S[i,j] = Diff(s_i, s_j). \tag{2}$$

where $Diff(s_i, s_j)$ is the difference measure between the strips discussed below. An example of $S[i,j]$ is shown in Figure 2(f), where each grey level block corresponds to an entry in the matrix.

In comparing two strips that correspond to the same actual physical location, we cannot assume that they cover the same vertical areas, but we can assume that they both have overlapping regions. Therefore, in order to detect their vertical alignment, we further divide each strip into blocks and generate a block-to-block distance matrix $B[k,l]$ for each pair of strips. Each entry in this matrix corresponds to the distance between a block b_k from the first strip and a block b_l from the second strip:

$$B[k,l] = Diff(b_k, b_l). \tag{3}$$

where $Diff(b_k, b_l)$ is the distance defined in section 3.1. We look for the best diagonal (as explained in section 3.1) in the distance matrix $B[k,l]$ and record its start and end points. The value of this diagonal is chosen as the distance measure between the two strips:

$$S[i,j] = \min_{1 \le k \le N, 1 \le l \le N} \{B[k,l]\}. \tag{4}$$

The comparison process is shown in Figure 2. Two mosaics are shown in Figure 2(c) and Figure 2(d). The strip comparison process (equation 3) is graphically displayed in Figure 2(e) and the matrix S is graphically displayed in Figure 2(f).

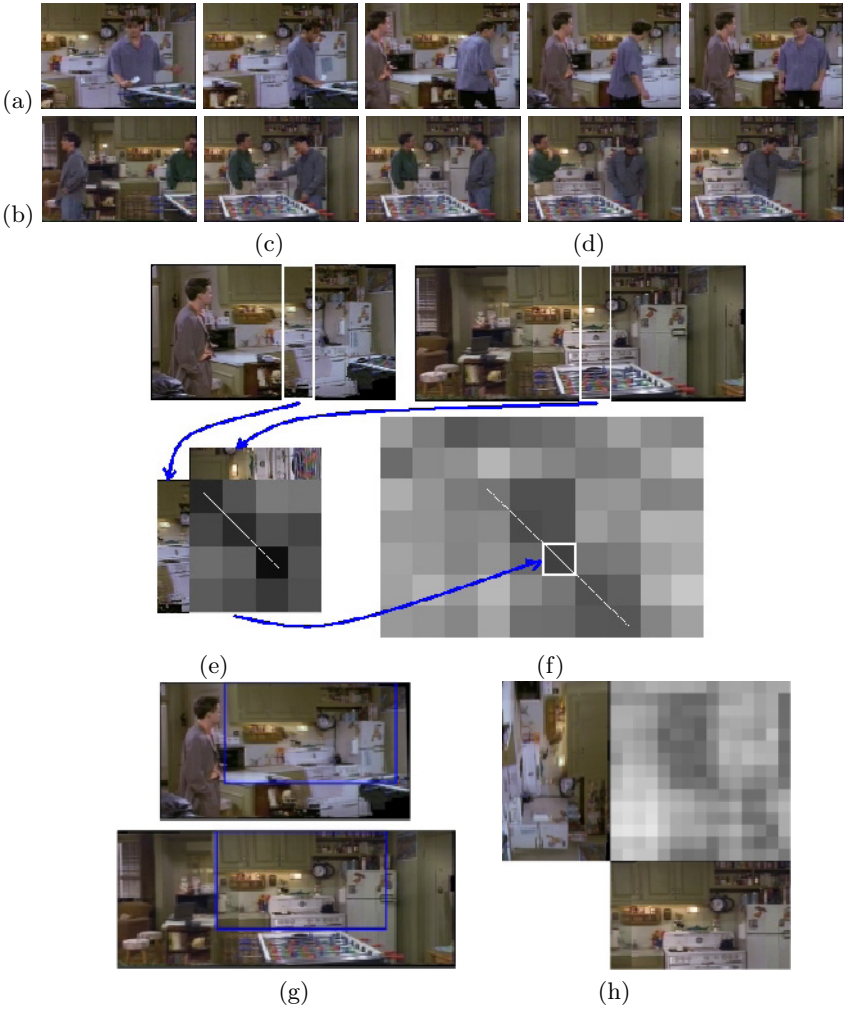


Fig. 2. (a) Key-frames of the shot from which the mosaic in (c) was generated. (b) Key-frames of the shot from which the mosaic in (d) was generated. (e) Each strip-pair (example marked in white) is divided into blocks. The values of the B matrix (e) for the marked strips is visualized by grey squares, where each entry represents the difference measure between a block in the example strip taken from (c) and a block in the example strip taken from (d). Coarse vertical alignment between the strips is determined by the "best" diagonal (thin white line); the average of the values under the diagonal defines the distance between the two strips. (f) The S matrix for the two mosaics, with the strip score from (e) highlighted. Horizontal alignment between strip sequences is determined by the "best" diagonal path in S (thin white line) and is used to (g) crop the mosaics to prepare for the verifying finer match in (h), which shows a similar S matrix for the finer stage.

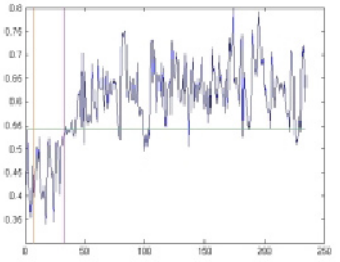


Fig. 3. Illustration of threshold determination for mosaics distance after the coarse stage. Values in the two leftmost group belong to mosaic pairs with a known common background, and the rest of the values are on the right. The leftmost group of 10 distance values is the sample group. The horizontal line is the maximum value among this sample group.

The height of this block matrix S is the same as width of the mosaic in Figure 2(c) and its width is the width of the mosaic in Figure 2(d), such that each block represents the strip-to-strip difference between the two mosaics.

We next find the best diagonal in the matrix of strip-to-strip differences $S[i, j]$ in order to find a horizontal alignment between strip sequences, and we record its start and end points. The area of interest is represented by a thin diagonal line across the matrix along which entries have low values (corresponding to good similarity). We use these start and end points to crop each mosaic such that only the corresponding matching areas are left. The start and end points of the diagonal in the S matrix set the vertical borders of the cropped mosaics. In order to determine the horizontal borders, we inspect the results of the vertical alignment for every pair of strips along the diagonal path found in S. Since for every strip pair we recorded the start and end points of the best diagonals in its corresponding B matrix, we take the average of these values and use it to set the horizontal border.

Determining Threshold. Once all mosaic pairs are processed, we check the distance values of the diagonal paths found for each pair. We expect that mosaic-pairs from different settings or with no common physical area will have high distance values, and we discard them according to a threshold. We determine the threshold by sampling several distance values of mosaic-pairs which have common physical areas, which therefore should have the lowest distance values. We choose the highest of these sampled values as our threshold. This sampling method proved correct after inspecting all mosaic-pairs distances as shown in Figure 3. To illustrate our results, we manually separated mosaic-pairs with common physical areas (two groups on left) from the rest of the mosaic-pairs (group on right). The sampled pairs are the leftmost group. The threshold chosen from the hand-labeled samples (horizontal line) quite accurately rejects mosaic pairs known to be from different physical areas, although it does permit a few false positive matches. If a diagonal path distance value exceeds this threshold, we determine that the two mosaics do not match. If we find a match, we continue

to the finer step where we only use the cropped mosaics: those parts of the mosaics corresponding to the best sub-diagonal found, an example of which is shown in Figure 2(g).

Fine Matching. After discarding mosaic-pairs which had diagonal paths with large distance values, we refine the measure of their closeness, in order to detect false positives and to more accurately determine physical background similarity. In this finer stage we only compare the cropped parts of the mosaics, applying a more precise method than the one used in the coarse stage. The cropped mosaics are displayed in Figure 2(h). We now use thinner strips (20 pixels wide) and also take into account the scale difference between the two mosaics. Assuming that the cropped mosaics cover the same regions of the physical setting, if we divide the narrower of the two cropped mosaics into K thin strips, then the best match will be a one-to-one match with K somewhat wider strips of the wider mosaic, where each strip pair covers the exact physical area. Let $\alpha \geq 1$ be the width ratio between the two mosaics. We re-compute histograms of 20×20 blocks for the narrower cropped mosaic, and histograms of $20\alpha \times 20\alpha$ blocks for the wider cropped mosaic. The best path in the new distance matrix should now have a slope of approximately $45°$. Matching in the finer stage has three advantages over the matching in the coarse stage. First, only few mosaic pairs are re-matched. Second, having adjusted for the mosaic widths, only diagonals parallel to the main diagonal need to be checked. Third, since the cropped mosaics cover the same regions, only complete diagonals rather than all possible sub-diagonals need to be checked. These diagonals are automatically known to satisfy the diagonal length and boundary constraints. These operations restrict the total number of strip and block comparisons to be performed and greatly lower computation time, even though there are more strips per mosaic. This final verification of mosaic match values is very fast.

The matching of mosaics could be further enhanced to support a more accurate alignment, by finding a coarse affine transformation between corresponding blocks. However, this coarse-fine approach yields a fast and simple matching method, which runs in real time and performs sufficiently accurately.

4 Clustering of Shots, Scenes, and Physical Settings

We present two examples of applying the mosaic comparison. We cluster shots in basketball videos in order to classify shots and detect repeating characteristic events. We generate a hierarchical representation of sitcoms by clustering scenes according to physical location. The video sequences are first divided into shots using the shot boundary detection technique described in [1]. For sitcoms the video is further divided into scenes using the method described in [10].

4.1 Clustering Shots in Sports Videos for Event Classification

We used the coarse stage of our mosaic comparison method to cluster shots from basketball videos. This stage allowed us to easily distinguish between shots

showing wide side views of the basketball court ("court shots"), close-up shots of people, shots taken from above the basket and shots showing the court from a close-up side view. The clustering of the mosaics led to a more meaningful categorization than clustering key-frames of the same shots. Automatic key-frame selection would not always capture the correct side of the court; some shots have varying zoom, so if a key-frame is chosen when the camera has zoomed in, it will not match other key-frames showing a wide view of the court; key-frames do not easily distinguish between court shots which show the right right, the left court, or both. Figure 5(a) shows a mosaic generated from a panned and zoomed court shot, and the corresponding key-frames are shown in Figure 5(b).

Preliminary results from the clustering of basketball videos allowed us to classify shots and to determine temporal relations between clusters. Filtering rules, corresponding to video grammar editing rules, were manually determined to extract events of human interest. For example, in the video of a European basketball game, we detected all foul penalty throws, and used them as bookmarks in a quick browsing tool shown in Figure 5(c). Foul penalty throws were characterized by a three-part grammatical sequence: a court shot, followed by a close-up shot, followed by a shot from above the basket. In another example, we analyzed an NBA basketball video and discovered that field goal throws were usually characterized by a court shot followed by a close-up shot of the player making the goal. These instances were easily detected after the mosaic clustering was used as input to classify court shots and close-up shots. A similar detection of field goal throws has been performed by [12], who relied on the analysis of the scoreboard image superimposed upon the frames and the audio detection of cheers from the crowd. We tested our field goal throw detection technique on a different basketball video, and received comparable results. Further analysis of the mosaics distinguishes between different categories of field goal throws and labels each with its corresponding team.

4.2 Mosaic-Based Scene Representation in Sitcoms

A scene is a collection of consecutive shots, related to each other by some spatial context, which could be an event, a group of characters engaged in some activity, or a physical location. However, a scene in a sitcom occurs in a specific physical location, and these locations are usually repetitive throughout the episode. Some physical locations are characteristic of the specific sitcom, and repeat in almost all of its episodes. Therefore, it is advantageous to describe scenes in sitcoms by their physical location. These physical settings can then be used to generate summaries of sitcoms and to compare different episodes of the same sitcom. This property is not confined to the genre of sitcoms. It could be employed for other video data which have the hierarchical structure of scenes and which are constrained by production economics, formal structure, and/or human perceptive limits to re-use their physical settings.

We are only interested in shots that have the most information about the scene. Once these shots are determined, their corresponding mosaics are chosen as the representative mosaics ("R-Mosaics") of the scene. One example of a "good

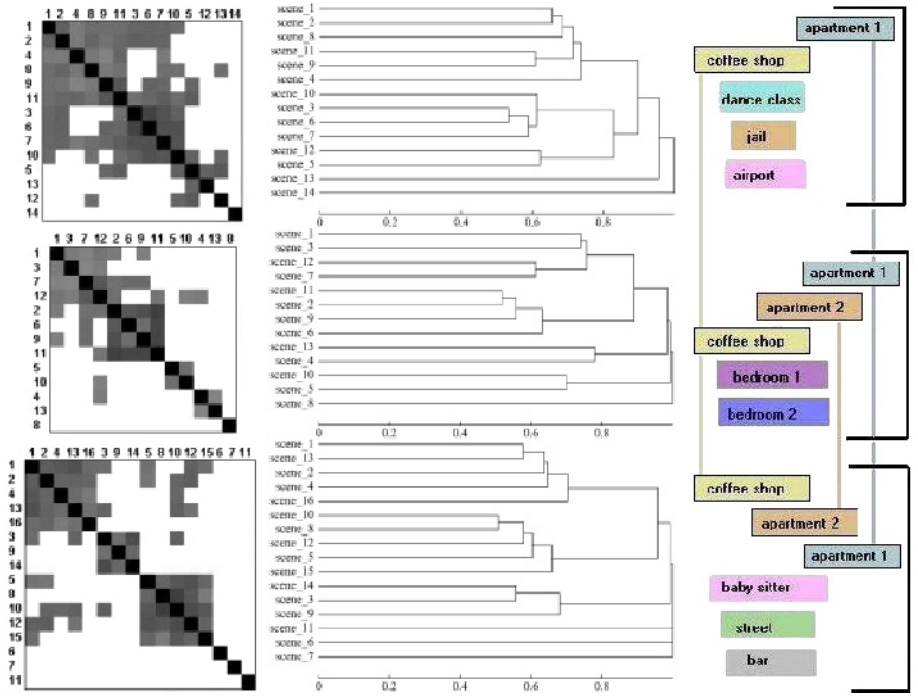
shot" is an extended panning shot, as shown in Figure 1. Another example is a zoom shot; the zoomed-out portion of the shot will most likely show a wider view of the background, hence expose more parts of the physical setting. Detecting "good" shots is done by examining the registration transformations computed between consecutive frames in the shot. By analyzing these transformations, we classify shots as panning, zoomed-in, zoomed-out, or stationary. Once shots are classified, a reference frame for the mosaic plane is chosen accordingly. For panning and stationary shots the middle frame is chosen, whereas for zoomed-in and zoomed-out shots the first and last frames are chosen, respectively. We derived a threshold experimentally that allowed us to only choose shots with significant zoom or pan. For mosaics generated from sitcoms, further processing is needed, since some shots have significant parallax motion. We segment such shots into two parts and construct two separate mosaics for each part.

However, some scenes are mostly static, without pan or zoom shots. Moreover, sometimes the physical setting is not visible in the R-Mosaic because a pan shot was close up. For these scenes, an alternative shot which better represents the scene has to be chosen. We observed that most scenes in sitcoms (and all the static scenes that we processed) begin with an interior "establishing shot", following basic rules of film editing [2]. The use of an establishing shot appears necessary to permit the human perception of a change of scene; it is temporally extended wide-angle shot (a "full-shot" or "long-shot"), taken to allow identification of the location and/or characters participating in that scene. Therefore, we choose the first interior shot of each scene to be an R-mosaic; for a static scene, it is the only R-mosaic. In our experiments, using these pan, zoom, and establishing shot rules leads to up to six R-Mosaics per scene.

Once R-mosaics are chosen, we can use them to compare scenes within the same episode as well as scenes across different episodes. The similarity between two scenes is determined by the maximum similarity between any of their R-Mosaics. This is due to the fact that different shots in a scene might contain different parts of the background, and when attempting to match scenes that share the same physical location, we need to find at least one pair of shots (mosaics) from the two scenes that show the same part of the background.

5 Results

The result of the mosaic-based scene clustering of sitcoms are shown in Figure 4. The order of the entries in the original scene difference matrices were manually arranged to reflect the order of the reappearance of the same physical settings. Dark blocks represent good matches (low difference score). As can be seen in the left column of the middle sitcom, there are 5 main clusters representing the 5 different scene locations. For example, the first cluster along the main diagonal is a 4×4 square representing scenes from "Apartment1". The false similarity values outside this square are due to actual physical similarities shared by "Apartment1" and "Apartment2", for example, light brown kitchen closets. Nevertheless, it did not effect our scene clustering results, as can be seen in the



(a) (b) (c)

Fig. 4. Results from three episodes of the same sitcom. (a) Similarity graphs for scene clustering results, in matrix form (darker = more similar). Scene entries are arranged manually in the order of physical settings. For example, there are 13 scenes in the sitcom represented in the middle grouping, scenes $1, 3, 7, 12$ are in "Apartment1", scenes $2, 6, 9, 11$ are in "Apartment2", scenes $5, 10$ are in "Coffee Shop", scenes $4, 13$ are in "Bedroom1" and scene 8 is in "Bedroom2". (b) Corresponding dendrograms (generated using [6]). Each cluster in each dendrogram represents a different physical setting. (c) Illustration of the results of inter-video comparison of physical settings. Settings appear in the same order they were clustered in (b). Lines join matching physical settings, which are common settings in most episodes of this sitcom: "Apartment1", "Apartment2" and "Coffee Shop". The rest of the settings in each episode identify the main plots of the episode, for example, the main plots in the first episode took place in the settings: jail, airport and dance class.

corresponding diagram in the middle column in Figure 4(b), and the dendrogram generated by the results of applying a weighted-average clustering algorithm [9] to the original scene difference matrix on its left.

When grouping information from several episodes of the same sitcom, we detect repeating **physical settings**. The rest of the scenes in each episode are usually unique for that episode and suggest information about the main **plots** of that episode. We summarized the clustering information from three episodes of the same sitcom and computed similarities across the videos based on physical settings. We use all the mosaics from all the scenes in a physical setting to

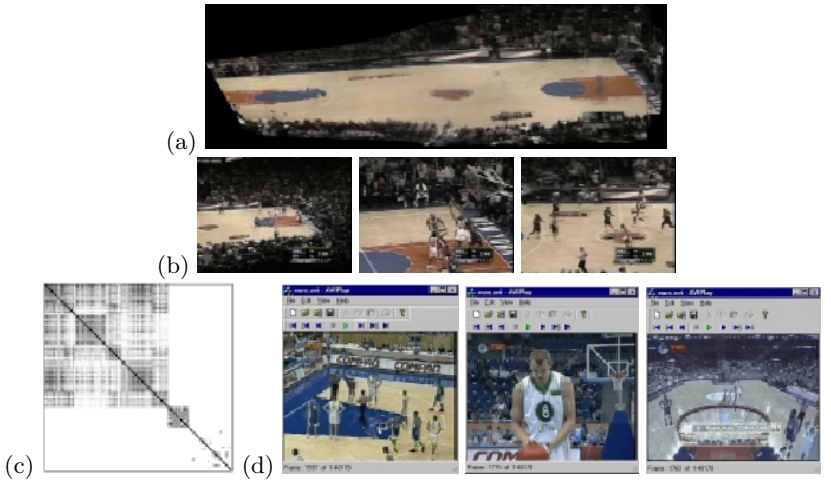


Fig. 5. (a) Example of a mosaic from an NBA basketball sequence with (b) corresponding key-frames. Note that the zoom in the key frames changes, making it harder to compare and classify them, whereas the zoom of the mosaics is uniform. (c)First stage (coarse) comparison of shots from a European championship basketball video. The first cluster along the main diagonal represents court shots which cluster together in this stage, the following cluster represents foul penalty shots that were taken from above the basket. The rest of the shots are various close-ups of basketball players and the coach. (d) Screen captures of a video player that has been augmented with bookmarks. It indexes to those important events that have been derived using a grammar rule applied to mosaic clusters. At left, beginning of court shot of foul (where the bookmark is set), followed by a close-up shot, then the characteristic shot taken from above the basket.

represent it, and choose the minimum distance between a mosaic pair to define the setting distance. To compare between three long video sequences, each about 40K frames long, we only need to compare 5 settings of the first episode with 5 settings of the second episode and 6 settings of the third episode. The results are illustrated in Figure 4(c).

Comparing settings across episodes leads to a higher-level contextual identification of the **plots** in each episode, characterized by settings which are unique to that episode. For example, in the episode at the top of Figure 4 the main plots involve activities in a dance class, jail and airport. We have observed that most sitcoms do in fact involve two or three plots, and anecdotal feedback from human observers suggests that people do relate the plot to the unusual setting in which it occurs. This non-temporal indexing of a video within a library of videos is directly analogous to the Term Frequency/Inverted Document Frequency (TFIDF [14]) measure of information retrieval used for documents. That is, what makes a video unique is the use of settings which are unusual with respect to the library.

Results from shots clustering of 15 minutes segment of a European basketball video are shown in Figure 5(c). The figure shows the results of clustering mosaics after only the coarse stage. The clusters at the top-left (about $\frac{2}{3}$ of the matrix)

are all court shots. The following cluster is of shots taken from above the basket, used for foul penalty shots. The rest of the data is of various close-up shots. We also analyzed the first quarter of an NBA basketball video and detected 32 sequences of a court shot followed by a close-up shot, which were good candidates for representing field goals. Of these, 18 were regular field goals, 7 were assaults, 2 occurred just before a time-out and the rest of the 5 instances showed the missed field goals of a well-known NBA player (This happened to be Michael Jordan's first game after retirement, which explains why cameras switched to close-ups even though he missed). All of these instances serve as interesting events in the game. They became bookmarks for an augmented video player which allows regular viewing of the game as well as skipping from one bookmark to the other. Screen captures of this video player for a foul shot bookmark are shown in Figure 5(d).

6 Conclusions and Future Work

This paper presents a compact approach for summarizing video, which allows efficient access, storage and indexing of video data. The mosaic-based representation allows direct and immediate access to the physical information of scenes. It therefore allows not only efficient and accurate comparison of scenes, but also the detection and highlighting of common physical regions in mosaics of different sizes and scales.

We also presented a new type of video abstraction. By clustering scenes into an even higher level representation, the **physical setting**, we create a non-temporal organization of video. This structured representation of video enables reliable comparison between different video sequences, therefore allows both intra- and inter-video content-based access. This approach leads to a higher-level contextual identification of **plots** in different episodes. This structured mosaic-based representation is not confined to the genres of situation comedies; it is also suitable for other TV series, dramas and news. In news videos, for example, it allows classifying broadcasts from well-known public places. It could also help to classify events in different sports videos such as basketball, soccer, hockey and tennis according to the characteristics of the playing field.

Our mosaic comparison method gave satisfactory results even for distorted mosaics when shots had parallax motion, or for mosaics with global or local distortions due to bad registration. However, we are aware of methods to improve the appearance of mosaics such as the ones suggested in [20], [16] or [17], if one is able to afford the additional computation time.

The method described here would serve as a useful tool for content-based video access. Video sequences are represented in their multiple levels of organization in the following hierarchical structure: frames, shots, scenes, settings and plots. This allows both a temporal representation of video for fast browsing purposes as well as a non-temporal representation for efficient indexing.

Acknowledgements. We wish to thank Michal Irani from the Weizmann Institute of Science in Israel for her wise advice and substantial support. We are grateful for her allowing us to use the facilities in the lab as well as the code for image registration and grey level mosaic generation. We also wish to thank Lihi Zelnik, Tal Hassner and Yaron Caspi for their help.

References

1. A. Aner and J. R. Kender. A unified memory-based approach to cut, dissolve, key frame and scene analysis. In *ICIP*, 2001.
2. D. Arijon. *Grammar of the Film Language*. Silman-James Press, 1976.
3. M. Gelgon and P.Bouthemy. Comparison of automatic shot boundary detection algorithms. In *ECCV*, 1998.
4. R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Addison Wesley, 1993.
5. A. Hanjalic, R. L. Lagendijk, and J. Biemond. Automated high-level movie segmentation for advanced video retrieval systems. In *IEEE Transactions on Circuits and Systems for Video Technology*, volume 9, Jun. 1999.
6. http://odur.let.rug.nl/ kleiweg/clustering/clustering.html.
7. M. Irani and P. Anandan. Video indexing based on mosaic representations. In *Proceedings of the IEEE*, volume 86, 1998.
8. M Irani, P. Anandan, J. Bergenand R. Kumar, and S. Hsu. Efficient representation of video sequences and their applications. In *Signal processing: Image Communication*, volume 8, 1996.
9. Anil K. Jain and Richard C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, Englewood Cliffs, NJ, 1988.
10. J. R. Kender and B.L. Yeo. Video scene segmentation via continuous video coherence. In *CVPR*, 1998.
11. R. Lienhart. Determining a structured spatio-temporal representation of video content for efficient visualisation and indexing. In *SPIE Storage and Retrieval for Image and Video Databases VII*, volume 3656, 1999.
12. S. Nepal, U. Srinivasan, and G. Reynolds. Automatic detection of 'goal' segments in basketball videos. In *ACM Multimedia*, 2001.
13. J. Oh, K. A. Hua, and N. Liang. Scene change detection in a MPEG compressed video sequence. In *In SPIE Multimedia Computing and Networking*, Jan. 2000.
14. G. Salton and M. McGill. *Introduction to modern information retrieval*. New York: McGraw-Hill, 1983.
15. F. Schaffalitzky and A. Zisserman. Viewpoint invariant texture matching and wide baseline stereo. In *ICCV*, 2001.
16. R. Szeliski and S. Heung-Yeung. Creating full-view panoramic image mosaics and environment maps. In *SIGGRAPH*, 1997.
17. N. Vasconcelos. A spatiotemporal motion model for video summarization. In *CVPR*, 1998.
18. M. Yeung and B. Liu. Efficient matching and clustering of video shots. In *ICIP*, 1995.
19. M. Yeung and B.L. Yeo. Time-constrained clustering for segmentation of video into story units. In *ICPR*, 1996.
20. A. Zomet, S. Peleg, and C. Arora. Rectified mosaicing: Mosaics without the curl. In *CVPR*, 2000.