SUMMARIZING VIDEO USING A SHOT IMPORTANCE MEASURE AND A FRAME-PACKING ALGORITHM

Shingo Uchihashi and Jonathan Foote

FX Palo Alto Laboratory
3400 Hillview Avenue
Palo Alto, CA 94304
{shingo, foote}@pal.xerox.com

ABSTRACT

This paper presents methods of generating compact pictorial summarizations of video. By calculating a measure of shot importance video can be summarized by de-emphasizing or discarding less important information, such as repeated or common scenes. In contrast to other approaches that present keyframes for each shot, this measure allows summarization by presenting only the most important shots. Selected keyframes can also be resized depending on their relative importance. We present an efficient packing algorithm that constructs a pictorial representation from differently-sized keyframes. This results in a compact and visually pleasing summary reminiscent of a comic book.

1. INTRODUCTION

A “shot” is a segment of video or motion image that is typically contiguous in time and visual space. Many techniques exist to automatically segment video into its component shots, typically by finding large frame differences that correspond to cuts or shot boundaries. In many applications it is desirable to automatically create a summary of an existing video, motion picture or broadcast. Typically, this is done by segmenting video into shots, and representing the entire video with a collection of keyframes, one for each shot. In contrast, the system presented here abstracts video by selectively discarding or de-emphasizing redundant information. For example, repeated shots need not be included if they are similar to shots already shown. We present a measure of shot importance, given an existing segmentation. An immediate application is printing a video summary, where frames from important shots are printed, while those of lesser importance are not. In our system, less important keyframes are printed in a smaller size. A novel algorithm presented here efficiently packs different-sized keyframes into a compact and visually pleasing summary reminiscent of a comic book or Japanese *manga*.

2. RELATED WORK

Shahraray *et al.* at ATT Research have worked on using keyframes for an HTML presentation of video [5]. One keyframe was selected for each shot; uniformly sized keyframes laid out in a column along closed-caption text. Taniguchi *et al*. have summarized video using a 2-D packing of “panoramas” which are large images formed by compositing video pans [6]. In this work, keyframes were extracted from every shot and used for a 2-D representation of the video content. Because frame sizes were not adjusted for better packing, much white space can be seen in the summary results. Yeung *et al*. have made pictorial summaries of video using a “dominance score” for each shot [8]. Though they work towards a similar goal, their implementation and results are substantially different. The sizes and the positions of the still frames are determined only by the dominance scores, and are not time-ordered. Other tools have been built for browsing video content [1][2][7][9]. These do not attempt to summarize video but rather present video content “as it is”. Therefore, keyframes are typically extracted from every shot and not selected to reduce redundancy. Frame presentation is basically linear, although some approaches occasionally use other structures [2][7].

3. MEASURING SHOT IMPORTANCE

Many techniques exist to automatically segment video into its component shots, typically by finding large frame differences that correspond to cuts, or shot boundaries. Once detected, shots can be clustered by similarity such that similar shots (e.g. similar camera angles or subjects) are considered to be one shot or cluster. For example, a film dialog where the camera repeatedly alternates between two actors would typically consist of two clusters, one for each actor. We use a hierarchical clustering method, where initially each frame in the video (or a sub-sampled representation) is assigned a unique cluster. The number of clusters is reduced by iteratively merging the two closest clusters at each step, based on the minimum distance between all combinations of the two cluster member frames. To compare the distance between frames, a number of techniques are available, such as the color-histogram distances described in [3] or the transform-coefficient distance of [4]. Hierarchical clustering results in a tree-structured representation such that individual frames are on the leaves of the tree. At the root node of the tree is the maximal cluster consisting of all the frames. The children of each node are the sub-clusters that were merged to form the node, and so forth down to the leaves. If the distance of the merged clusters is stored with each node, it can be used to select a desired number of clusters by thresholding. Setting a threshold distance below which frames are assumed to be in the same cluster can adjust the number between one (the root of the tree) and the number of frames (the leaves of the tree). The optimal number of clusters depends on the type and length of the video. Once clusters have been selected, each shot is labeled with its corresponding cluster. Given C clusters in the video, a measure of normalized weight W_i for cluster i is computed as

$$W_i = \frac{S_i}{\sum_{j=1}^{C} S_j} \tag{1}$$

where S_i is the total length of all shots in cluster i, found by summing the length of all shots in the cluster. W_i is the proportion of shots from the whole video that are in cluster i.

A shot is important if it is both long and rare, that is, it does not resemble most other shots. Thus weighting the shot length with the inverse of the cluster weight yields a measure of shot importance. Thus the importance I of shot j (from cluster k) is

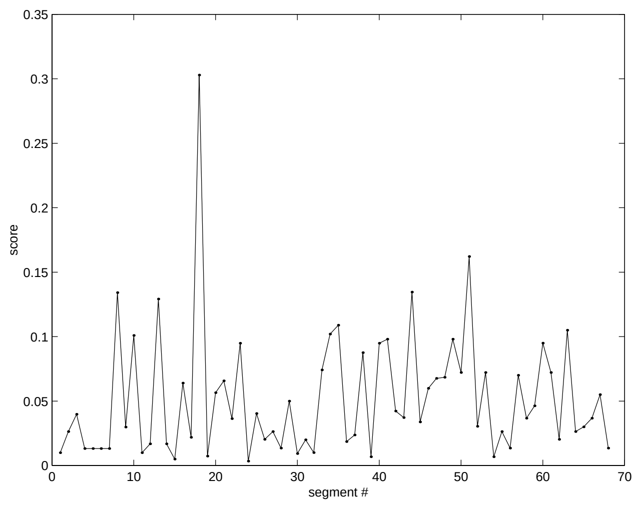

Figure 1. Shot importance versus shot number

$$I_j = L_j \log \frac{1}{W_k} \tag{2}$$

where L_j is the length of the shot j.

The importance measure becomes larger if the shot is long, and smaller if the cluster weight is large (meaning the shot is common). The balance of the contribution from the shot length and the cluster weight can be adjusted by weighting the reciprocal of W_i by a factor other than unity. Shot importance versus time for a meeting video is shown in Figure 1.

4. A KEYFRAME PACKING ALGORITHM

The importance calculated for each shot can be thresholded to select a desired number of shots, and hence frames for a pictorial summary. Once frames have been selected, they may be laid out in one or more dimensions to form a pictorial abstract of the video sequence. Two dimensions might be most appropriate for a printed synopsis, such as a "comic book" format. Thresholding the importance score allows the desired number of frames to be displayed at the appropriate level. To facilitate layout, frames may be displayed in smaller or bigger sizes, depending on their importance score, and may be resized to best fit the available space.

Given that frames can be selected from shots, the layout problem reduces to finding a sequence of frame sizes that both fills space efficiently and represents the original video sequence well. It is not hard to come up with an appropriate cost function to define a degree of matching. However, it is difficult to find an optimal sequence because the number of possible sequences increases enormously with the size of the space.

Existing techniques such as Dynamic Programming (DP) and greedy algorithms can be used to find an optimal or near-optimal layout. A greedy approach is simple, yet often fails to produce a good result. The DP algorithm is guaranteed to find the optimal solution for this kind of problem, but there is no need to optimize the layout over the entire video. We present a packing algorithm that is simpler to apply than DP yet provides a much better layout than a purely greedy strategy.

The algorithm can be described as "block exhaustive" as it selects the best sequence of frames to pack a particular "block" or sub-region of the entire space. The best sequence is found by exploring all combinations of layouts for a particular block. Because blocks are relatively small, this does not result in a combinatorial explosion and an optimal layout can be easily found using a plain tree search.

The space to be packed is divided into a grid, such that one unit of the grid will hold the smallest frame. This is used to lay out frames as follows. One "row block", or row of columns across the grid, is packed at a time. The matching score for a block varies with its height. The height for a particular block is set to the one that yields the best packing as determined by the packing score. Once a row block as been packed with frames, further row blocks are considered iteratively until all frames have been packed.

Given:

- A sequence of frame sizes $f_1, f_2, \ldots, f_N$, expressed as multiples of the smallest frame size. This is determined from the shot importance score as above. f_i takes one of K values $\{s_1, s_2, \ldots, s_K\}$.
- The allowable range of block row heights, as M values $\{r_1, r_2, \ldots, r_M\}$ and the block width to the fixed value W.
- A function $c(x, y)$ which is the cost of placing frame of size x in available remaining space of size y. A typical cost function might be the size difference between x and y. Any arbitrary cost function can be used.
- A space-filling rule. A typical rule might be column-major, that is from top to bottom and left to right. This is the order in which frames will be preferentially packed.

Given the above, the packing algorithm consists of three nested loops that optimize the packing for one row block as described below.

1. Set starting frame s to 1.
2. Set row height r to one of M values, $\{r_1, r_2, \ldots, r_M\}$
3. Find all frame sequences $\{\boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_{Lr}\}$ that fit the "row block".
4. From the above sequences, find a sequence $\boldsymbol{q}_l$ of length n_l which fits a portion of the original sequence between s and $s + n_l$, where f_i indicates ith frame size of the original sequence. q_{ij} is the jth element of sequence $\boldsymbol{q}_i$. w_i is an additional weighting factor which is determined by remaining space

$$l = \operatorname*{argmin}_i \left(\frac{1}{n_i} \sum_{j=1}^{n_i} c(f_{s+j-1}, q_{ij}) + w_i \right) \tag{3}$$

5. Repeat 2 through 4 to find the best row height r and corresponding sequence $\boldsymbol{q}$ of length n. This is the optimal packing for this row block.
6. Increase s by n.
7. Repeat 2 to 6 until s reaches N, the length of the original sequence (all frames are packed).

Step 2 above is done by 1) exhaustively generating all sequences of length from 1 to $r \times W$ whose jth element is one of K values $\{s_1, s_2, \ldots, s_K\}$, and 2) eliminating all generated sequences that do not fit the "row block." For example, if the block height is 2, sequences containing a 3 element do not fit, or if the block width W is 8 and the maximum row height is 3, sequences of all 3s longer than 3 do not fit. Though there are a possible $(r \times W)^K$

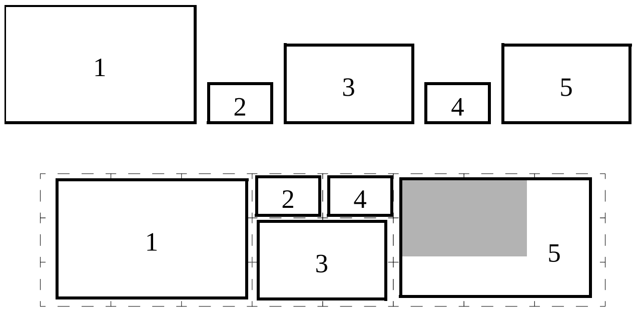

Figure 2. Packing keyframes into a row block

sequences to consider, *r*, *W*, and *K* are very small so efficiency is not a great concern.

An example of this "row block" packing procedure is depicted in Figure 2. The rectangles at the top are the original frame sequence, sized by importance. The bottom picture illustrates the frames packed into a row block of height 3 and width 8. Note that the size of frame 5 has been increased from the original (indicated as a gray rectangle) for a better packing with minimal white space.

5. EXPERIMENTS

We have tested the above algorithm on a corpus of videotaped staff meetings, though it should work on any video type. Space permits a presentation of only one example result. The source video is approximately 50 minutes long and consists of 49 shots, counting between actual camera changes. Still frames were extracted from the video every 5 seconds, and were clustered into 51 clusters using the Euclidean distance between high-variance DCT coefficients as a distance metric [4]. Occasionally, frames from the same shot were segregated to different clusters because of large changes such as camera motion or changes in room lighting. As a result, the entire video was clustered into 92 segments. For each segment, an importance measure was calculated based on its length and cluster weight as described above.

Video segments having high importance scores were chosen to generate a still synopsis. Segments scoring higher than 1/8 of the maximum score were selected. For each segment chosen, the frame nearest the segment mean was extracted as a representative keyframe, again using the Euclidean distance of reduced DCT coefficients. Frames were sized according to the importance measure of their originating segments, so higher importance frames were larger. In our experiment, if the importance of a given frame was between 1/8 and 1/4 of the maximum, it was assigned the smallest frame size. Frames scoring more than 1/4 but less than 1/2 of the maximum were sized twice the smallest size, and frames scoring higher than 1/2 were sized three times larger than the smallest. In the case of our sample video, the number of extracted frames of size 1, 2, 3 were 9, 13, and 2 respectively. We have chosen a straightforward assignment of frame sizes to importance scores; many other assignments, including continuously variable sizes, are possible. The result of the packing algorithm is shown in Figure 3. The sizes of only six frames out of 24 needed adjustment for this packing.

6. CONCLUSIONS

This paper has introduced a shot importance measure and shown its application to video summarization. The importance score was used to assign sizes to keyframes extracted from video shots A algorithm for efficiently packing different-sized keyframes was presented and demonstrated to be effective.

We are interested using other information in the importance calculation. One good factor is the magnitude of the change that starts the shot, computed as the difference of the color histogram, pixel difference, or transform coefficient difference. Including this would decrease a shot's importance if it is not greatly different from the preceding shot.

A particularly valuable enhancement might be to preferentially weight certain shot categories. For example close-ups of a person might be preferable to wide crowd shots. Our importance value can be easily modified to reflect this kind of information, for example as produced by a face-detection algorithm.

7. ACKNOWLEDGMENTS

Thanks to John Doherty for producing the meeting videos in our corpus.

8. REFERENCES

[1] Aigrain, P., Joly, P., and Longueville, V., "Medium Knowledge-Based Macro-Segmentation of Video into Sequences," *Intelligent Multimedia Information Retrieval*, AAAI Press/ The MIT Press, pp. 159--173, 1997.

[2] Arman, F., Depommier, R., Hsu, A., Chiu, M.-Y., "Content-based Browsing of Video Sequences," *Proc. ACM Multimedia 94*, San Francisco, October 1994, pp. 97--103.

[3] Boreczky, J. and Rowe, L., "Comparison of Video Shot Boundary Detection Techniques," *Proc. SPIE Conference on Storage and Retrieval for Still Image and Video Databases IV*, San Jose, CA, February, 1996, pp. 170--179.

[4] Girgensohn, A., and Foote, J., "Video Frame Classification Using Transform Coefficients," Submitted to *ICASSP-99*.

[5] Shahraray, B. and Gibbon, D. C., "Automated Authoring of Hypermedia Documents of Video Programs", *Proc. ACM Multimedia 95*, San Francisco, November, pp. 401--409, 1995.

[6] Taniguchi, Y., Akutsu, A., Tonomura, Y., "PanoramaExcerpts: Extracting and Packing Panoramas for Video Browsing", *Proc ACM Multimedia 97*, pp. 427--436, 1997.

[7] Yeo, B-L., and Yeung, M., "Classification, Simplification and Dynamic Visualization of Scene Transition Graphs for Video Browsing," *Proc. IS&T/SPIE Electronic Imaging '98: Storage and Retrieval for Image and Video Databases VI.*

[8] Yeung, M., and Yeo, B-L., "Video Visualization for Compact Presentation and Fast Browsing of Pictorial Content," *IEEE Trans. Circuits and Sys. for Video Technology*, Vol. 7, No. 5, pp. 771--785, Oct. 1997.

[9] Zhang, H. J., Low, C. Y., Smoliar, S. W. and Wu, J. H., "Video Parsing, Retrieval and Browsing: An Integrated and Content-Based Solution," *Proc. ACM Multimedia 95*, San Francisco, November 1995, pp. 15--24.

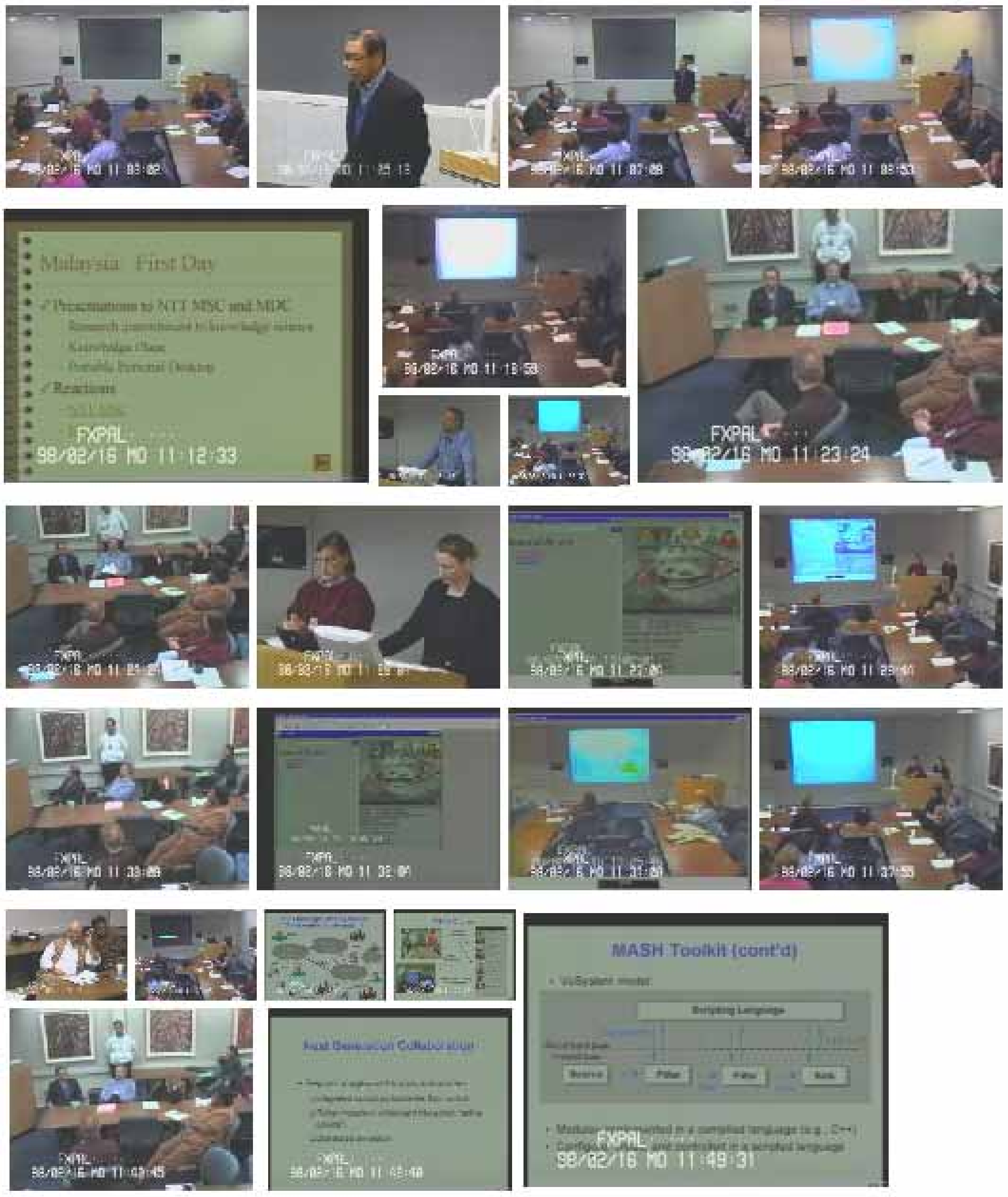

Figure 3. Result of keyframe selection, resizing, and automatic layout