A Shortest Path Representation for Video Summarisation

S.V. Porter, M. Mirmehdi and B.T. Thomas
Department of Computer Science, University of Bristol, Bristol BS8 1UB, U.K.
{porter,majid,barry}@cs.bris.ac.uk

Abstract

A novel approach is presented to select multiple key frames within an isolated video shot where there is camera motion causing significant scene change. This is achieved by determining the dominant motion between frame pairs whose similarities are represented using a directed weighted graph. The shortest path in the graph, found using the A^* search algorithm, designates the key frames. The overall method can be applied to extract a set of key frames which portray both the video content and camera motions, all of which are useful features for video indexing and retrieval.

1 Introduction

The amount of archived digital video data being made available on-line is ever increasing. Whilst the storage and delivery capabilities of computer systems are expanding to meet these demands, the use of on-line multimedia information is inhibited by the lack of efficient authoring and querying tools. To allow users efficient access it is necessary to develop tools that facilitate searching based on non-sequential browsing and visual content-based indexing. Managing large quantities of film and video material is also becoming an increasing problem throughout the broadcasting industry, particularly where indexing and annotating archived material is concerned. Manually indexing video content is currently the most accurate method but it is a very laborious and time consuming process. To search digital video material, wherever it is stored, users must be able to browse and index into video through visual content to enable efficient access.

A predominant approach to this problem is to use a video abstract for indexing and retrieval. A video abstract is defined as a short sequence of images, extracted from a longer video whilst still preserving the essential message [8]. The difficulty in composing such an abstract is determining which frames best represent the video contents. A common approach is to segment temporally a sequence into shots, where a shot is defined as a sequence of frames captured from a single camera operation, and then select a single representative key frame for each shot. The resulting ordered set of key frames is referred to as a filmstrip [2].

To allow efficient indexing, an abstract must represent the entire video content with as little redundancy as possible. Each key frame should represent a video segment which exhibits consistency in content. Hence, temporally segmenting a video into its constituent shots is a fundamental component in automatic video indexing. However, a shot can contain *camera motions* that may drastically change its content. This means that one key frame may not always be sufficient to represent each shot. In this paper, we propose an algorithm which assumes that a video sequence has already been temporally segmented into individual shots using the edit effect detection algorithm outlined in [9]. Then, estimates of the dominant motion between each frame pair are used to decide when there has been sufficient camera motion to require another key frame. A weighted directed graph is formed for each shot where the vertices represent the frames in the shot and the weight on each edge is a measure of similarity between each frame pair. The frames corresponding to the vertices forming the shortest path through the graph are used as representative key frames for each shot. Thus, a small subset of frames can be used to retrieve information from the video and enable content-based video browsing.

2 Video Summarisation

Different approaches to video summarisation often depend on the purpose of the abstract. Lienhart et al. [8] concentrate on the generation of trailers for movies as an important type of abstract. The abstracts are short and designed to attract the attention of the viewer without revealing too much of the story line. In contrast, an abstract for a documentary or a digital video library should capture all the content of the video. Two approaches to this latter problem are filmstrips, mentioned earlier, and skims [2]. A video skim incorporates both video and audio information and the abstraction is played rather than viewed stati-

0-7695-1948-2/03 $17.00 © 2003 IEEE

cally. Whereas many of the approaches to video summarisation rely on the explicit detection of shot changes [1, 7] other techniques have been proposed that commonly transform the image into a lower dimensional space, e.g. using PCA [6]. Typically, frames are then grouped using a clustering algorithm [6] or curve simplification [4] and the frame closest to each cluster centroid is chosen as a key frame.

Fundamentally, in order to create an efficient index, an algorithm for video abstraction should extract a set of key frames representing all the content of the video sequence whilst minimising redundancy. We define the video content to include objects and background that appear in the video and any events that occur in the video such as camera motions. We propose an approach to summarise a video that assumes the video sequence has been temporally segmented into shots using the edit effect detection algorithm outlined in [9]. It uses block matching motion compensation to generate an inter-frame difference metric. For each block in frame $f(n-1)$, the best match in a neighbourhood around the the corresponding block in frame $f(n)$ is sought. This is achieved by calculating the normalised correlation between blocks in the frequency domain and locating the maximum correlation coefficient, the value of which is used as a goodness-of-fit measure for each block. The estimated motion vectors are then used to track the blocks over time. In the present work, we use these motion vectors to estimate the dominant motion between each frame pair. Assuming the dominant motion was caused by camera motion, these estimates can then be used to identify shots containing significant camera motion that may require more than one key frame to represent their content.

Camera motions can be grouped into two broad classes: (i) tripod motion and (ii) free motion. If a camera is fixed to a tripod it can only exhibit three types of motion; pan, tilt and zoom. If there is free motion of the camera it can additionally track, boom or dolly. The effect of a pan on the change of contents in a shot is very similar to that of a track. For example, if a camera pans or tracks right, the background and objects on the left will gradually leave the shot while new background and objects may appear on the right. Such similarities can also be drawn between "tilt and boom" and "zoom and dolly". For this reason, we use a simple motion model which only represents the scale and translation in x and y between two frames. The point $p_i = (x, y)$ in frame $f(n-1)$ is transformed to the point $p'_i = (x', y')$ in frame $f(n)$, with respect to a reference point (x_r, y_r) according to

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} {}^s\theta_n(x - x_r) + {}^x\theta_n \\ {}^s\theta_n(y - y_r) + {}^y\theta_n \end{pmatrix}. \quad (1)$$

In practice, the frame centre is taken as the reference point. The parameter vector $\boldsymbol{\theta}_n = ({}^s\theta_n, {}^x\theta_n, {}^y\theta_n)$ corresponds to the scale, translation in x and translation in y respectively, between the frames $f(n-1)$ and $f(n)$. The model parameters are estimated using the robust estimator MSAC [10] which provides good estimates in the presence of outliers. Outliers may be present in the data where the motion equation is invalidated or where points correspond to secondary motions.

3 Model Parameter Estimation

Robust estimators such as RANSAC [5] are commonly used when least squares estimators can not handle significant numbers of outliers. Using the RANSAC algorithm we can select two points in frame $f(n-1)$ at random and estimate the model parameters $\boldsymbol{\theta}_n$ using their corresponding motion vectors. The algorithm then finds how many of the remaining data items fit the model with parameter vector $\boldsymbol{\theta}_n$ within a given tolerance, T. Each remaining point p_i in frame $f(n-1)$ is transformed to point p'_i in frame $f(n)$ according to $\boldsymbol{\theta}_n$ and to point p''_i using its corresponding motion vector. The error e_i between the model and the actual measurements for point p_i is then the Euclidean distance $e_i = |p'_i - p''_i|$. We allow the error e_i to be a maximum of one pixel in x and y, giving $T = \sqrt{2}$. RANSAC would then proceed in the same manner seeking to maximise the number of inliers. The RANSAC algorithm has proven very successful for robust estimation, but Torr and Zisserman have proposed improvements in [10]. They showed that RANSAC finds the minimum of a cost function $C = \sum_i \rho(e_i^2)$ where

$$\rho(e_i^2) = \begin{cases} 0 & e_i^2 < T^2 \\ 1 & e_i^2 \geq T^2. \end{cases} \quad (2)$$

In other words, inliers score nothing and each outlier scores a constant penalty. If T was set sufficiently high then all points would be inliers to all solutions leading to a poor estimate of the model parameters $\boldsymbol{\theta}_n$. In [10], this was improved on by minimising a new cost function where the robust error term $\rho()$ was redefined as

$$\rho(e_i^2) = \begin{cases} e_i^2 & e_i^2 < T^2 \\ T^2 & e_i^2 \geq T^2. \end{cases} \quad (3)$$

That is to say, outliers are still given a fixed penalty but inliers are scored on how well they fit the data. This new robust estimator (MSAC: m-estimator sample consensus) [10] was used in our algorithm to estimate the motion model parameter vector $\boldsymbol{\theta}_n$.

4 Minimising Representational Redundancy

To portray all of the video content, we must determine when there has been sufficient scene change due to camera motion to warrant more than one key frame to represent a shot. We formulate this as a shortest path problem. Given a directed weighted graph the shortest path between two vertices is the path of minimum total weight. Edge weights are often used to represent distance, time, cost, or any other quantity that accumulates linearly along a path, that one wishes to minimise. We need to minimise the similarity between key frames. Therefore, we find the shortest path through a graph where vertices correspond to frames in the shot and edge weights are a measure of similarity between two frames. The frames corresponding to vertices forming the shortest path through the graph are then used as representative key frames for each shot.

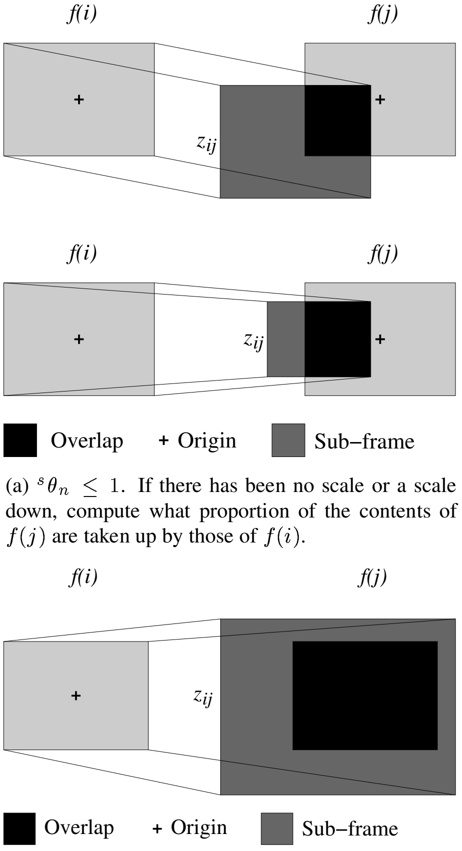
We define the similarity metric $\psi(i,j)$ as the amount of overlap between the contents of the frames $f(i)$ and $f(j)$ where $0 \leq \psi(i,j) \leq 1$, based on the assumption that the content is only significantly changed by camera motion. If there has been no camera motion between the frame pair $f(i)$ and $f(j)$ then they will contain the same content and $\psi(i,j) = 1$. As the amount of camera motion between two frames increases the amount of overlap will decrease until the contents of each frame are disparate and $\psi(i,j) = 0$. For each shot we have an estimate of the motion parameter vector $\boldsymbol{\theta}_n$ for each consecutive frame pair $f(n-1)$ and $f(n)$. Given any two frames $f(i)$ and $f(j)$, we accumulate the motion parameters between them to obtain $\boldsymbol{\Delta}_{ij} = ({}^s\Delta_{ij}, {}^x\Delta_{ij}, {}^y\Delta_{ij})$ where

$$ {}^s\Delta_{ij} = \prod_{n=i+1}^{j} {}^s\theta_n, \tag{4}$$

$$ {}^x\Delta_{ij} = {}^x\theta_j + {}^s\theta_j \cdot {}^x\Delta_{i(j-1)}, \tag{5}$$

$$ {}^y\Delta_{ij} = {}^y\theta_j + {}^s\theta_j \cdot {}^y\Delta_{i(j-1)} \tag{6}$$

and ${}^x\Delta_{i(j-1)} = {}^y\Delta_{i(j-1)} = 0$ when $i = j-1$. Hence, ${}^s\Delta_{ij}$, ${}^x\Delta_{ij}$ and ${}^y\Delta_{ij}$ are the total amount of scale, translation in x and translation in y respectively, between $f(i)$ and $f(j)$. This accumulated motion parameter vector is then used to compute the amount of overlap between the contents of the two frames. There are two cases to be considered when computing the amount of overlap depending on the scale parameter ${}^s\Delta_{ij}$: (i) ${}^s\Delta_{ij} \leq 1$ and (ii) ${}^s\Delta_{ij} > 1$. In each case, assume any frame $f(i)$ has constant width w, height h and area $A = w \cdot h$ with the position $(0,0)$ at its centre. We apply the motion transformation $\boldsymbol{\Delta}_{ij}$ to frame $f(i)$ to obtain what we shall call a sub-frame z_{ij} with a new width ${}^w z_{ij} = w \cdot {}^s\Delta_{ij}$, height ${}^h z_{ij} = h \cdot {}^s\Delta_{ij}$, area ${}^A z_{ij} = {}^w z_{ij} \cdot {}^h z_{ij}$ and its centre at the position $({}^x\Delta_{ij}, {}^y\Delta_{ij})$. The sub-frame conveys the size and position of the contents of frame $f(i)$ relative to frame $f(j)$. In other words, computing the amount of physical overlap between $f(i)$ and the sub-frame z_{ij}, defined by $\phi(i,j)$, is equivalent to computing the amount of overlap between the contents of frame $f(i)$ and frame $f(j)$. If ${}^s\Delta_{ij} \leq 1$, there has either been no scaling or a scale down plus possibly translation in x and y. In this case, we must compute what proportion of $f(j)$ is taken up by the contents of $f(i)$ hence, $\psi(i,j) = \phi(i,j)/A$ which is illustrated in Figure 1(a). In case (ii) when ${}^s\Delta_{ij} > 1$, there has been a scale up plus possibly translation in x and y so we must compute what proportion of the contents of $f(i)$ are still in $f(j)$, thus $\psi(i,j) = \phi(i,j)/{}^A z_{ij}$ which is shown in Figure 1(b).



(a) ${}^s\theta_n \leq 1$. If there has been no scale or a scale down, compute what proportion of the contents of $f(j)$ are taken up by those of $f(i)$.

(b) ${}^s\theta_n > 1$. If there has been a scale up, compute what proportion of the contents of $f(i)$ are still present in $f(j)$.

Figure 1. Computing the similarity metric between frames $f(i)$ and $f(j)$.

5 Graph-based Shot Representation

We now use this similarity metric to represent each individual shot as a graph. Let us define a graph $G = \{V, E\}$ comprised of a set V of N vertices, $\{v_1, .., v_N\}$, and a set $E \subseteq V \times V$ of directed weighted edges connecting vertices in V. In a directed graph, each edge also has a direction, so edges (v_i, v_j) and (v_j, v_i), where $i \neq j$, are distinct. The weight of an edge connecting two vertices v_i and v_j is defined by $\omega(v_i, v_j)$. A path from vertex v_i to vertex v_m is a set of connected edges $\{(v_i, v_j), (v_j, v_k), ..., (v_l, v_m)\}$ from E. The weight of path $p = \langle v_0, v_1, ..., v_k\rangle$ is the sum of the weights of its constituent edges:

$$\Omega(p) = \sum_{i=1}^{k} \omega(v_{i-1}, v_i). \tag{7}$$

If one or more paths exist from v_0 to v_k the shortest path is defined as the path p with the minimum total weight, $min\{\Omega(p)\}$ [3].

The connectivity of a graph can be represented as an adjacency matrix M in which each element (i, j) represents the edge between vertices v_i and v_j. If there is an edge (v_i, v_j) then $M_{ij} = \omega(v_i, v_j)$ otherwise $M_{ij} = 0$. Our goal is to form an adjacency matrix for each shot where vertices correspond to individual frames and $\omega(v_i, v_j) = \psi(i, j)$. Hence, the shortest path from the first frame to the last frame will minimise the amount of overlap between the contents of the representative key frames. If there is no overlap between two frames $f(i)$ and $f(j)$ then by definition, $M_{ij} = \psi(i, j) = 0$. This implies the key frames corresponding to the vertices in the shortest path must always have some overlap of their contents. In fact, we define a threshold T_{min} to specify the minimum amount of overlap there must be between key frames. Thus an edge (v_i, v_j) only exists if $\psi(i, j) \geq T_{min}$. Additionally, to preserve temporal coherence in the video index, a directed edge (v_i, v_j) can only exist if $f(j)$ succeeds $f(i)$ in the video sequence. Figure 2 shows a visualisation of the adjacency matrix representing a shot where the camera pans continuously to the right with $T_{min} = 0.2$. The shortest path from the first vertex to the final vertex is $p = \langle 0, 75, 134\rangle$. The frames corresponding to these vertices which are used to summarise this shot are shown in Figure 3(a). The weight of the two edges that form the shortest path are $\omega(0, 75) = 0.261$ and $\omega(75, 134) = 0.205$ and the total weight of the shortest path is $\Omega(p) = 0.466$. For comparison, the frames representing the second shortest path $p' = \langle 0, 70, 92, 134\rangle$ are shown in Figure 3(b) with $\Omega(p') = 1.443$. This illustrates that comparatively the shortest path minimises representational redundancy.

Figure 4(a) shows the adjacency matrix representing a shot of 336 frames where the camera pans to the right whilst tilting up followed by a pan left with a tilt down returning to the origin. It can be seen where the latter frames start to overlap again with those earlier in the sequence. The shortest path in this graph is $p = \langle 0, 234, 336\rangle$ and $\Omega(p) = 0.570$. To be an efficient index into a video, the key frames must depict all of the content and convey the temporal order of events in the shot i.e. the camera motion. The key frames here (i.e. 0, 234, and 336) can be seen in Figure 5. In this example, the three selected key frames would not represent any of the video content when the camera pans far to the right. We therefore add a final constraint to forming an adjaceny matrix. In addition to the earlier condition that an edge only exists if $\omega(v_i, v_j) \geq T_{min}$, the edge (v_i, v_j) only exists if $\omega(v_i, v_k) \geq T_{min}$ for all $i \leq k \leq j$. Figure 4(b) shows the adjacency matrix representing this shot after this constraint has been added and the shortest path is now $p = \langle 0, 27, 57, 86, 203, 234, 336\rangle$ with $\Omega(p) = 1.430$ which represent all of the shot content, as shown in Figure 5.

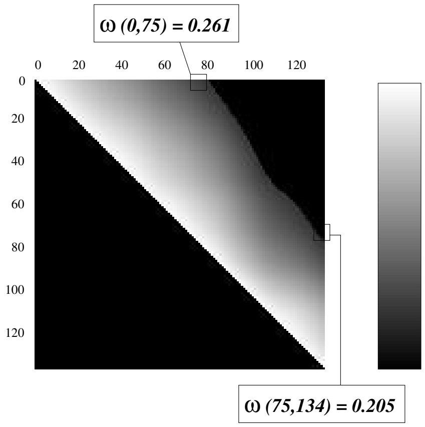


Figure 2. Adjaceny matrix representing a panning shot with $T_{min} = 0.2$

In summary, we form an adjacency matrix for each shot where the vertices represent each frame of the sequence and $\omega(v_i, v_j) = \psi(i, j)$. Table 1 outlines the conditions for which a directed edge (v_i, v_j) exists.

Condition 1	$j > i$
Condition 2	$\omega(v_i, v_j) \geq T_{min}$
Condition 3	$\omega(v_i, v_k) \geq T_{min}$ for all $i < k < j$

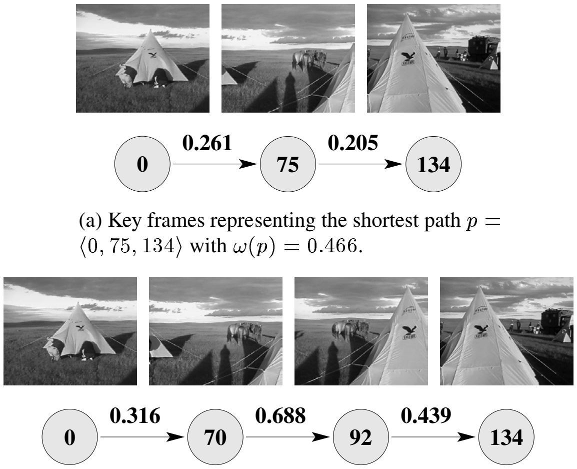
Table 1. Conditions for which a directed edge (v_i, v_j) exists.

6 Finding the Shortest Path

There are many search algorithms available for finding the shortest path from a starting vertex to a final vertex. We use the A^* graph search algorithm which guarantees the shortest path, provided a possible path exists. Central to the A^* algorithm is the use of an evaluation function for ordering the vertices in the search space:

$$f(v_i) = g(v_i) + h(v_i) \tag{8}$$

where $g(v_i)$ is the actual cost of reaching v_i from the starting vertex and $h(v_i)$ is a heuristic estimate of reaching the



(a) Key frames representing the shortest path $p = \langle 0, 75, 134 \rangle$ with $\omega(p) = 0.466$.

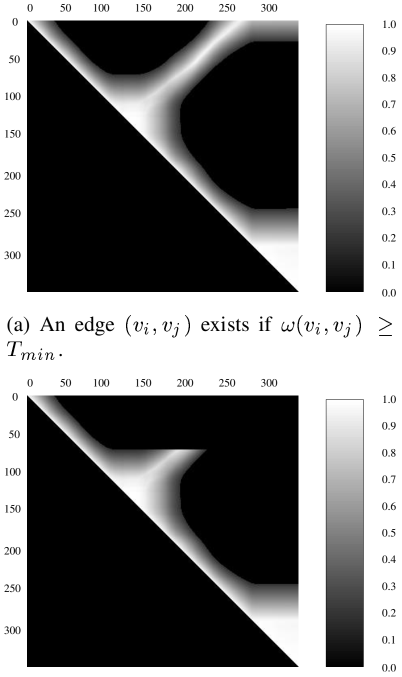
(b) Key frames representing the second shortest path $p' = \langle 0, 70, 92, 134 \rangle$ with $\omega(p') = 1.443$.

Figure 3. The shortest path results in less representational redundancy.

final vertex from vertex v_i which must always be an underestimate. It can be shown that an optimistic heuristic h always results in an optimal solution. In our algorithm, $h(v_i)$ is the minimum weight of all existing edges from v_i. Hence, the actual cost from v_i to reach the final vertex will always be greater than or equal to this estimate.

7 Extracting Key Frames

So far, we have used the frames corresponding to the vertices in the shortest path to represent the video. It follows, that we will always have a minimum of two key frames to represent each shot. However, when there is little or no camera motion a single key frame could potentially be sufficient. We introduce a second threshold T_{max} which defines the maximum amount of overlap between two key frames. If there are more than two vertices in the shortest path or there are only two vertices with $\Omega(p) < T_{max}$, then the path accurately summarises the video. However, if there are only two vertices v_i, v_j in p and $\Omega(p) \geq T_{max}$, we select a vertex that best represents the edge (v_i, v_j), i.e. we select the frame corresponding to the vertex v_k with the most overlap with all the other frames between and including frames $f(i)$ and $f(j)$. That is the vertex v_k with the maximum 'minimum edge weight' $\omega(v_k, v_l)$, $\forall\ i \leq k, l \leq j$, $k \neq l$. The values $T_{min} = 0.2$ and $T_{max} = 0.8$ were used to generate the example video abstracts shown in Figures 6 and 7 for two different video sequences. These thresholds have remained unchanged in all our experiments, but can be set according to user preference.



(a) An edge (v_i, v_j) exists if $\omega(v_i, v_j) \geq T_{min}$.

(b) An edge (v_i, v_j) only exists if $\omega(v_i, v_k) \geq T_{min}, \forall\, i < k < j$.

Figure 4. Adjaceny matrices representing a shot with a significant pan right and tilt up followed by a pan left and tilt down back to the origin.

8 Conclusions

We have presented a novel approach to key frame extraction by organising the similarity of frames within a shot as a graph structure. The frames corresponding to the vertices in the shortest path within it are then used to generate a content-based video index analogous to a storyboard. There is no absolute measure for the quality of an abstraction. Ultimately, the effectiveness of an approach can only be evaluated by users of a video library in which the system is implemented. Here, we have shown some example video abstracts to demonstrate our proposed method. Better subjective judgement can be made by viewing more video shots and key frames on-line[1]. The abstraction of video content is a very complex theme. In this work we have concentrated on using camera motion as one feature that changes scene content. Future work will be focused on characterising the camera motions and semantic changes in the scene due to object motion.

[1] http://www.cs.bris.ac.uk/home/porter/spath/video.html

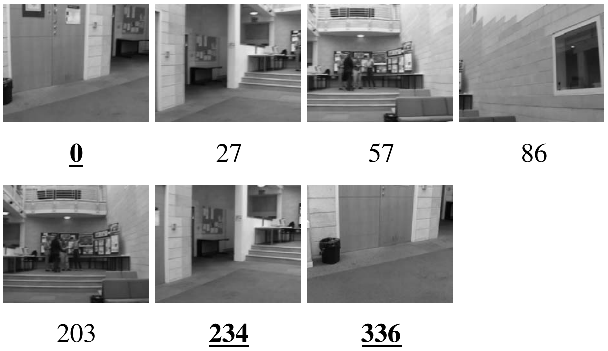


Figure 5. Shortest path key frames for the adjacency matrix in Figure 4(b). The three underlined frames denote the shortest path for the adjacency matrix in Figure 4(a).

Acknowledgements

The first author would like to thank UBQT Media Plc. for funding received during this work.

References

[1] J. Boreczky and L. Rowe. Comparison of video shot boundary detection techniques. In *SPIE volume 2670*, pages 170–179, 1996.

[2] M. Christel, A. Hauptmann, A. Warmack, and S. Crosby. Adjustable filmstrips and skims as abstractions for a digital video library. In *IEEE Advances in Digital Libraries Conference*, pages 98–104, 1999.

[3] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. MIT Press, 1990.

[4] D. DeMenthon, V. Kobla, and D. Doermann. Video summarization by curve simplification. In *ACM Multimedia*, pages 211–218, 1998.

[5] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, June 1981.

[6] D. Gibson, N. Campbell, and B. Thomas. Visual abstraction of wildlife footage using GMM and MDL criterion. In *ICPR*, pages 814–817, 2002.

[7] R. Lienhart. Comparison of automatic shot boundary detection algorithms. In *SPIE volume 3656*, pages 290–301, 1999.

[8] R. Lienhart, S. Pfeiffer, and W. Effelsberg. Video abstracting. *Comm. of the ACM*, 40(12):54–62, 1997.

[9] S. Porter, M. Mirmehdi, and B. Thomas. Detection and classification of shot transitions. In *12th BMVC*, pages 73–82. BMVA Press, 2001.

[10] P. H. S. Torr and A. Zisserman. MLESAC: A new robust estimator with application to estimating image geometry. *CVIU*, 78(1):138–156, 2000.

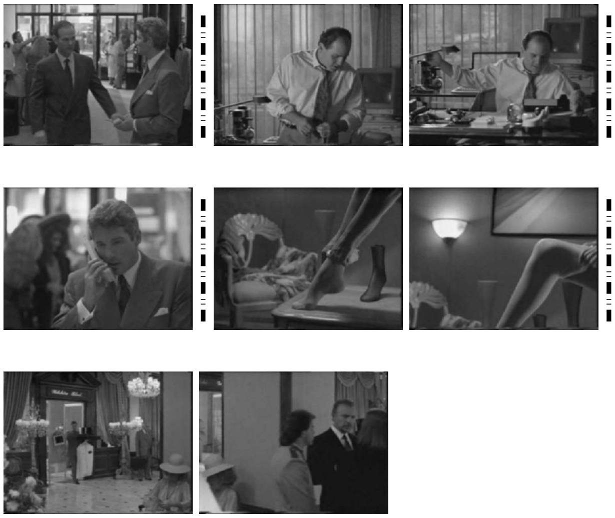


Figure 6. Pretty Woman Video Abstract - Shot change delineations are also shown.

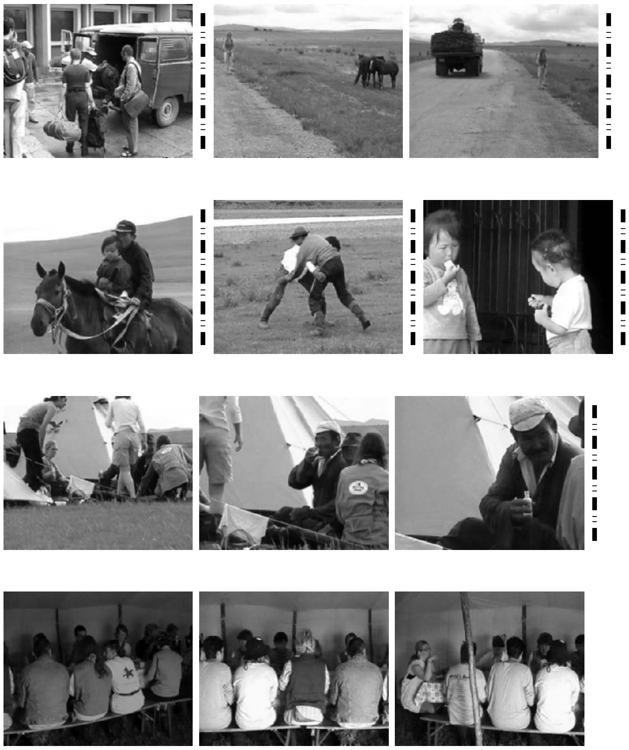


Figure 7. Mongolia Video Abstract - Shot change delineations are also shown.