Efficient Layout of Comic-Like Video Summaries

Janko Ćalić, David P. Gibson, and Neill W. Campbell

***Abstract*—In order to represent large amounts of information in the form of a video key-frame summary, this paper studies narrative grammar of comics, and using its universal and intuitive rules, lays out visual summaries in an efficient and user centered way. The system ranks importance of key-frame sizes in the final layout by balancing the dominant visual representability and discovery of unanticipated content utilizing a specific cost function and an unsupervised robust spectral clustering technique. A final layout is created using an optimization algorithm based on dynamic programming. Algorithm efficiency and robustness are demonstrated by comparing the results with the optimal panelling solutions.**

***Index Terms*—Reverse storyboarding, video representation, video summarization.**

I. Introduction

IN ORDER to enable intuitive access to large image and video archives, the main challenge of systems for video summarization and browsing is to achieve a good balance between removal of redundant sections of video and representative coverage of the video summary. Zhuang *et al.* [1] proposed an unsupervised clustering method based on HSV color features, where the frame closest to the cluster center is chosen as the key frame representative for a given video shot. Utilizing cluster-validity analysis, Hanjalic and Zhang [2] remove the visual content redundancy among video frames using an unsupervised procedure. An interesting approach introduced by DeMenthon *et al.* [3] represents the video sequence as a curve in a high dimensional space, and the summary is represented by the set of salient points on that curve. Recently, Wah *et al.* [4] exploited a normalized cut algorithm to globally and optimally partition the graph representation into video clusters and describe the evolution and perceptual importance of a video segment.

This work makes a shift towards more user centered summarization and browsing of large video collections by augmenting interaction rather than learning the way users create related semantics. In order to create an effortless and intuitive interaction with the overwhelming extent of information embedded in video archives, we propose a system that exploits the universally familiar narrative structure of comics to generate easily readable visual summaries. Being defined as "spatially juxtaposed images in deliberate sequence intended to convey information" [5], comics are the most prevalent medium that expresses meaning through a sequence of spatially structured images. Exploiting this concept, the proposed system follows the narrative structure of comics, linking the temporal flow of video sequence with the spatial position of panels in a comic strip. This approach differentiates our work from more typical reverse-storyboarding [6] or video summarization approaches.

There have been attempts to utilize the form of comics as a medium for visual summarization of videos. In [7], a layout algorithm that optimizes the ratio of white space left and approximation error of the frame importance function is proposed. Following a similar approach, the work presented in [8] introduces a number of heuristic rules to optimize the layout algorithm. However, due to an inherently difficult optimization, these attempts failed to develop a feasible layout algorithm. In order to uncover the underpinning structure of the key-frame data in an unsupervised manner, our work exploits K-way spectral clustering method [9] in perceptual grouping of extracted key-frames using locally scaled affinity matrix [10].

The work presented in this paper introduces a number of novel approaches to the algorithm pipeline, improving the processing efficiency and quality of layout optimization. In terms of efficiency, our approach brings real-time capability to video summarization by introducing a solution based on dynamic programming (DP) and showing that the adopted suboptimal approach achieves nearly optimal layout results. Not only does it improve the processing time of the summarization task, but it enables new functionalities of visualization for large-scale video archives, such as runtime interaction, scalability, and relevance feedback. In addition, the presented algorithm applies a new approach to the estimation of key-frame sizes in the final layout by exploiting a spectral clustering methodology coupled with a specific cost function that balances between good content representability and discovery of unanticipated content. In addition, a robust unsupervised estimation of number of clusters is introduced. The evaluation results compared to existing methods of video summarization [8], [7] showed substantial improvements in terms of algorithm efficiency, quality of optimization, and possibility of swiftly generating much larger summaries.

In order to rank the importance of key-frames in the final visual layout, a specific cost function that relies on a novel robust image clustering method is presented in Section II. Two optimization techniques that generate a layout of panels in comic-like fashion are described in Section III. The first technique finds an optimal solution for a given cost function, while the second suboptimal method utilizes dynamic programming [11] to efficiently generate the summary. The results of the algorithms presented are evaluated by benchmarking the optimal against a suboptimal panelling solution.

II. Estimation of Frame Sizes

Our aim is to generate an intuitive and easily readable video summary by conveying the significance of a shot from analysed video sequences via the size of its key-frame representation. Any cost function that evaluates the significance is highly dependent upon the application. In our case, the objective is to

Manuscript received May 1, 2006; revised September 6, 2006 and November 22, 2006. This work was supported in part by the ICBR project within the 3C Research, Digital Media and Communications Innovation Centre. This paper was recommended by Associate Editor E. Izquierdo.

The authors are with the Department of Computer Science, University of Bristol, Bristol BS8 1RZ, U.K. (e-mail: janko@cs.bris.ac.uk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2007.897466

create a summary of archived video footage for production professionals. Therefore, the summary should clearly present visual content that is dominant throughout the analysed section of the video, as well as to highlight some cutaways and unanticipated content, essential for the creative process of production.

In the case of video summarization, the estimation of frame importance (in our case frame size) in the final video summary layout is dependant upon the underlying structure of available content. Thus, the algorithm needs to uncover the inherent structure of the dataset and by following the discovered relations evaluate the frame importance. By balancing the two opposing representability criteria, the overall experience of visual summary and the meaning conveyed will be significantly improved.

A. Frame Grouping

In order to generate the cost function that represents the desired frame size in the final layout: $C(i), i = 1, \ldots, N$ where $C(i) \in [0, 1]$ and N is the number of extracted key-frames for a given sequence, all key-frames are initially grouped into perceptually similar clusters. The feature vector of the ith frame $x_i, i = 1, \ldots, N$ used in the process of frame grouping is a $18 \times 3 \times 3$ HSV colour histogram appended with the pixel values of the DC sequence frame representation in order to maintain essential spatial information.

Large archives of raw video footage comprise of mainly repetitive video content inseparable from a random number of visual outliers such as establishing shots and cutaways. Centeroid-based methods like K-means fail to achieve acceptable results since the number of existing clusters has to be defined *a priori* and these algorithms break down in presence of nonlinear cluster shapes [12].

Being capable of analysing inherent characteristics of the data and coping very well with high nonlinearity of clusters, a *spectral clustering* approach was adopted as a method for robust frame grouping. The locally scaled affinity matrix $\mathbb{W}_{\text{loc}}^{N \times N}$, introduced by Lihi and Perona [10], is calculated as

$$\mathbb{W}_{\text{loc}}(i, j) = e^{\frac{-|x_i - x_j|}{2 \cdot \sigma_i \cdot \sigma_j}}. \tag{1}$$

Each element of the data set (i.e., a key-frame) has been assigned a local scale σ_i, calculated as median of κ neighbouring distances of element i. The selection of parameter value κ is independent of the scaling parameter σ and for high-dimensional data authors in [10] recommend that $\kappa = 7$.

The major drawback of K-way spectral clustering is that the number of clusters has to be known *a priori*. There have been a few algorithms proposed that estimate the number of groups by analysing eigenvalues of the affinity matrix. By analysing the ideal case of cluster separation, Ng *et al.* in [9] show that the eigenvalue of the Laplacian matrix $L = D - \mathbb{W}$ with the highest intensity (in the ideal case it is 1) is repeated exactly κ times, where κ is a number of well separated clusters in the data. However, in the presence of noise, when clusters are not clearly separated, the eigenvalues deviate from the extreme values of 1 and 0. Thus, counting the eigenvalues that are close to 1 becomes unreliable. Based on the same idea, Polito and Perona in [13] detect a location of a drop in the magnitude of the eigenvalues in order to estimate κ, but the algorithm still lacks the robustness that is required in our case.

Therefore, a novel algorithm to robustly estimate the number of clusters in the data is proposed. It follows the idea that if the clusters are well separated, there will be two groups of eigenvalues: one converging towards 1 (high values) and another towards 0 (low values). In the real case, convergence to those extreme values will deteriorate, but there will be two opposite tendencies and thus two groups in the eigenvalue set. In order to reliably separate these two groups, we have applied K-means clustering on sorted eigenvalues, where $K = 2$ and initial locations of cluster centers are set to 1 for high-value cluster and 0 to low-value cluster. After clustering, the size of a high-value cluster gives a reliable estimate of number of clusters k in analysed dataset.

Following the approach presented by Ng *et al.* in [9], a Laplacian matrix $L = D - \mathbb{W}_{\text{loc}}$ is initially generated with $\widehat{\mathbb{W}}_{\text{loc}}(i, i) = 0$, where D is the degree matrix. After solving the eigen-system for all eigenvectors eV of L, the number of clusters k is estimated following the aforementioned algorithm. The first k eigenvectors $eV(i), i = 1, \ldots, k$ form a matrix $X_{N \times k}(i, j)$. By treating each column of the row-normalized $\hat{X}$ as a point in $\mathbb{R}^k$, N vectors are clustered into k groups using the K-means algorithm. The original point i is assigned to cluster j if the vector $\hat{X}(i)$ was assigned to the cluster j.

B. Cost Function

To represent the dominant content in the selected section of video, the maximum cost function $C(i) = h_{\max}$ is assigned to the key-frame closest to the center of the corresponding cluster. If $d(i)$ is the ith frame's distance to the central frame and σ_i is the variance of the cluster, the cost function is calculated as follows:

$$C(i) = \alpha \cdot \left(1 - e^{\frac{-d(i)^2}{2\sigma_i^2}}\right) \cdot h_{\max}. \tag{2}$$

Normalizing $C(i)$ to the maximum row height $h_{\max}$, scales it to the interval of frame sizes used to approximate the cost function. The parameter α controls the balance between the importance of the cluster center and its outliers, and it is set empirically to 0.7. As a result, cluster outliers (i.e., cutaways, establishing shots, etc.) are presented as more important and attract more attention of the user than key-frames concentrated around the cluster centre. This grouping around the cluster centres is due to common repetitions of similar content in raw video rushes, often adjacent in time. To avoid the repetition of content in the final summary, a set of similar frames is represented by a larger representative, while the others are assigned a lower cost function value.

III. Panelling

The main task of the panelling module is to generate a frame layout that optimally follows the values of the cost function only using available panel templates. Each panel template generates a vector of frame sizes, that approximates the cost function values

of corresponding frames. Precision of this approximation depends upon the maximum size of a frame, defined by the maximum height of the panel $h_{\max}$ which gives granularity of the solution. For a given $h_{\max}$, a set of panel templates is generated, assigning a vector of frame sizes to each template.

$$f_0(p_1) = 0$$

$$\varepsilon = \varepsilon_1(p_1, p_2) + \varepsilon_2(p_2, p_3) + \cdots + \varepsilon_{n-1}(p_{n-1}, p_n) \quad (4)$$

$$\min \varepsilon(p_1, p_2, \ldots, p_n) = \min f_{n-1}(p_n) \quad (5)$$

$$f_{j-1}(p_j) = \min[f_{j-2}(p_{j-1}) + \varepsilon_{j-1}(p_{j-1}, p_j)]. \quad (6)$$

The first algorithm searches for all possible combinations of page layout and finds an optimal solution for a given cost function. The layout needs to fit exactly into a predefined page width with a fixed number of images per page. Given the page height H, page width W, and number of images per page $\mathcal{N}$, distribution of frame sizes depends on the cost function $C(i), i = 1 \ldots \mathcal{N}$. The algorithm is divided into two stages: 1) distribution of row heights and 2) distribution of panels for each row. In both stages, the search space is generated by the partitioning of an integer (H or W) into its summands. Since the order of the summands is relevant, it is the case of *composition* of an integer n into all possible k parts, in the form $n = r_1 + r_2 + \cdots + r_k, r_i \geq 0, i = 1, \ldots, k$. Due to a large number of possible compositions $(n + k - 1)!/(n!(k - 1)!)$, an efficient iterative algorithm described in [14] is used to generate all possible solutions. In order to find an optimal composition of page height H into k rows with heights $h(i), i = 1, \ldots, k$, for every possible $k \in [H/h_{\max}, H]$, a number of frames per row $\eta(i), i = 1, \ldots, k$ is calculated to satisfy the condition of even spread of the cost function throughout the rows

$$\forall i, \sum_{j=1}^{\eta(i)} C(j) = \frac{1}{k} \sum_{l=1}^{\mathcal{N}} C(l). \quad (3)$$

For each distribution of rows $\eta(i)$ and a given page width W, each row is laid out to minimize the difference between the achieved vector of frame sizes and the corresponding part of the cost function $C(i)$. For each composition of $\eta(i)$ a set of possible combinations of panel templates is generated. The vector of template widths used to compose a row has to fit the given composition, as well as the total number of used frames has to be $\eta(i)$. For all layouts that fulfill these conditions, the one that generates a vector of frame sizes with minimal approximation error to the corresponding part of the cost function is used to generate the row layout. The final result is the complete page layout $\Omega(i)$ with the minimal overall approximation error $\Delta = \sum C(i) - \Omega(i) \forall i$.

There have been numerous attempts to solve the problem of discrete optimization for spatio-temporal resources. In our case, we need to optimally utilize the available two-dimensional space given required sizes of images. However, unlike many well studied problems like stock cutting or bin packing [15], there is a nonlinear transformation layer of panel templates between the error function and available resources. In addition, the majority of proposed algorithms are based on heuristics and do not offer an optimal solution.

TABLE I
APPROXIMATION ERROR Δ AS A FUNCTION OF MAXIMUM ROW HEIGHT $h_{\max}$ AND NUMBER OF FRAMES ON A PAGE $\mathcal{N}$, EXPRESSED IN [%]

$h_{max} \setminus \mathcal{N}$	40	80	120	160	200	240
1	6.40	3.92	3.42	2.81	2.58	2.34
2	2.16	1.83	1.65	1.61	1.39	1.46
3	2.24	2.02	1.81	1.53	1.32	1.43
4	2.67	2.17	1.68	1.65	1.31	1.28

TABLE II
APPROXIMATION ERROR Δ USING OPTIMAL ALGORITHM FOR GIVEN $h_{\max}$ AND $\mathcal{N}$, EXPRESSED IN [%]

	Δ_{optimal}			$\lvert\Delta_{\text{DP}} - \Delta_{\text{optimal}}\rvert$		
$h_{max} \setminus \mathcal{N}$	40	80	120	40	80	120
1	6.40	3.92	3.42	0.00	0.00	0.00
2	1.87	1.57	1.45	0.29	0.26	0.20
3	2.05	1.34	1.81	0.19	0.68	0.00
4	2.21	1.62	1.60	0.39	0.55	0.08

TABLE III
COMPARISON OF LAYOUT ALGORITHM SPEEDS, DEPENDING UPON NUMBER OF FRAMES ON A PAGE $\mathcal{N}$, PAGE WIDTH W AND HEIGHT H

$\mathcal{N}$	W	H	$W \cdot H$	T_{ORIG}	T_{FAST}	T_{DPLY}
25	12	10	120	0.03	0.03	0.127
75	16	14	224	0.57	0.16	0.241
125	20	18	360	200	1.8	0.382
150	19	27	513	×	×	0.547
1000	42	59	2478	×	×	1.907
2400	64	90	5760	×	×	4.672

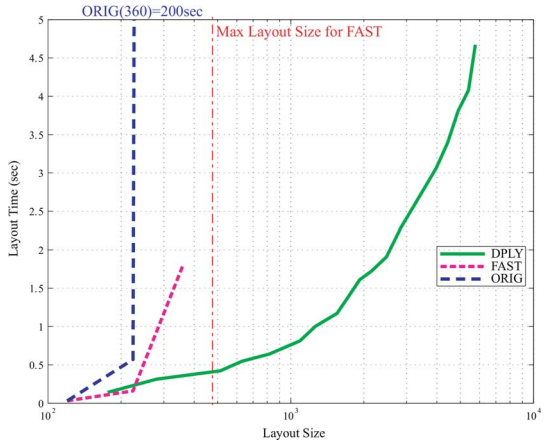

Fig. 1. Comparison of the layout algorithm speed for methods presented in [16] [ORIG], [8] [FAST] to our method [DPLY]. Linear complexity of the proposed layout algorithm is observable.

Therefore, we propose a suboptimal solution using dynamic programming and we will show that the deviation of achieved results from the optimal solution can be practically disregarded. Dynamic programming finds an optimal solution to an optimization problem $\min \varepsilon(p_1, p_2, \ldots, p_n)$ when not all variables in the evaluation function are interrelated simultaneously, as given in (4). In this case, solution to the problem can be found as an iterative optimization defined in (5) and (6), with initialization The adopted model claims that optimization of the overall page

Fig. 2. News sequence from the TRECVID 2006 search corpus, summarized using layout parameters $\mathcal{N} = 1$ and $H/W = 3/5$. Repetitive content is always presented by the smallest frames in the layout. On the other hand, outliers are presented as big (e.g., a commercial break within a newscast, row 2, frame 11) which is very helpful for the user to swiftly uncover the structure of the presented sequence.

layout error Δ is equivalent to optimization of the sum of independent error functions of two adjacent panels p_{j-1} and p_j, where

$$\varepsilon_{j-1}(p_{j-1}, p_j) = \sum_{i \in \{p_{j-1} \cup p_j\}} (C(i) - \Omega(i))^2. \tag{7}$$

Although the dependency between nonadjacent panels is precisely and uniquely defined through the hierarchy of the DP solution tree, strictly speaking the claim about the independency of sums from (4) is incorrect. The reason for that is a limiting factor that each row layout has to fit to required page width W, and therefore, width of the last panel in a row is directly dependent upon the sum of widths of previously used panels. If the task would have been to layout a single row until we run out of frames, regardless of its final width, the proposed solution would be optimal. Nevertheless, by introducing specific corrections to the error function $\varepsilon_{j-1}(p_{j-1}, p_j)$ the suboptimal solution often achieves optimal results.

The proposed suboptimal panelling algorithm comprises following procedural steps.

1) Load all available panel templates p_i
2) For each pair of adjacent panels: penalize, if panel heights are not equal, determine corresponding cost function values $C(i)$, form the error function table $\varepsilon_{j-1}(p_{j-1}, p_j)$, find optimal $f_{j-1}(p_j)$ and save it (penalizing means assigning the biggest possible error value to $\varepsilon_{j-1}(p_{j-1}, p_j)$.)
3) If all branches reached row width W, roll back through optimal $f_{j-1}(p_j)$ and save the row solution
4) If page height reached, display the page. Else, go to the beginning

In a specific case when the current width W_{curr} reaches the desired page width W, the following corrections to $\varepsilon_{j-1}(p_{j-1}, p_j)$ are introduced:

- if $W_{\text{curr}} > W$, penalize all but empty panels;
- if $W_{\text{curr}} = W$, return standard error function, but set it to 0 if the panel is empty;
- if $W_{\text{curr}} < W$, empty frames are penalized and error function is recalculated for the row resized to fit required width W, as

$$\varepsilon_{j-1}(p_{j-1}, p_j) = \sum_i \left(C(i) - \frac{W_{\text{curr}}}{W} \cdot \Theta(i) \right)^2. \tag{8}$$

IV. Results

The experiments were conducted on the TRECVID 2006 evaluation content, provided by NIST as the benchmarking material for evaluation of video retrieval systems. In order to evaluate the results of the DP suboptimal panelling algorithm, results are compared against the optimal solution, as described in Section III. Results in Table I show the dependency of approximation error defined in (9) for two main algorithm parameters: maximum row height $h_{\max}$ and number of frames on a page $\mathcal{N}$

$$\Delta = \frac{1}{\mathcal{N} \cdot \langle_{\max}} \sqrt{\sum_{i=1}^{\mathcal{N}} (C(i) - \Theta(i))^2}. \tag{9}$$

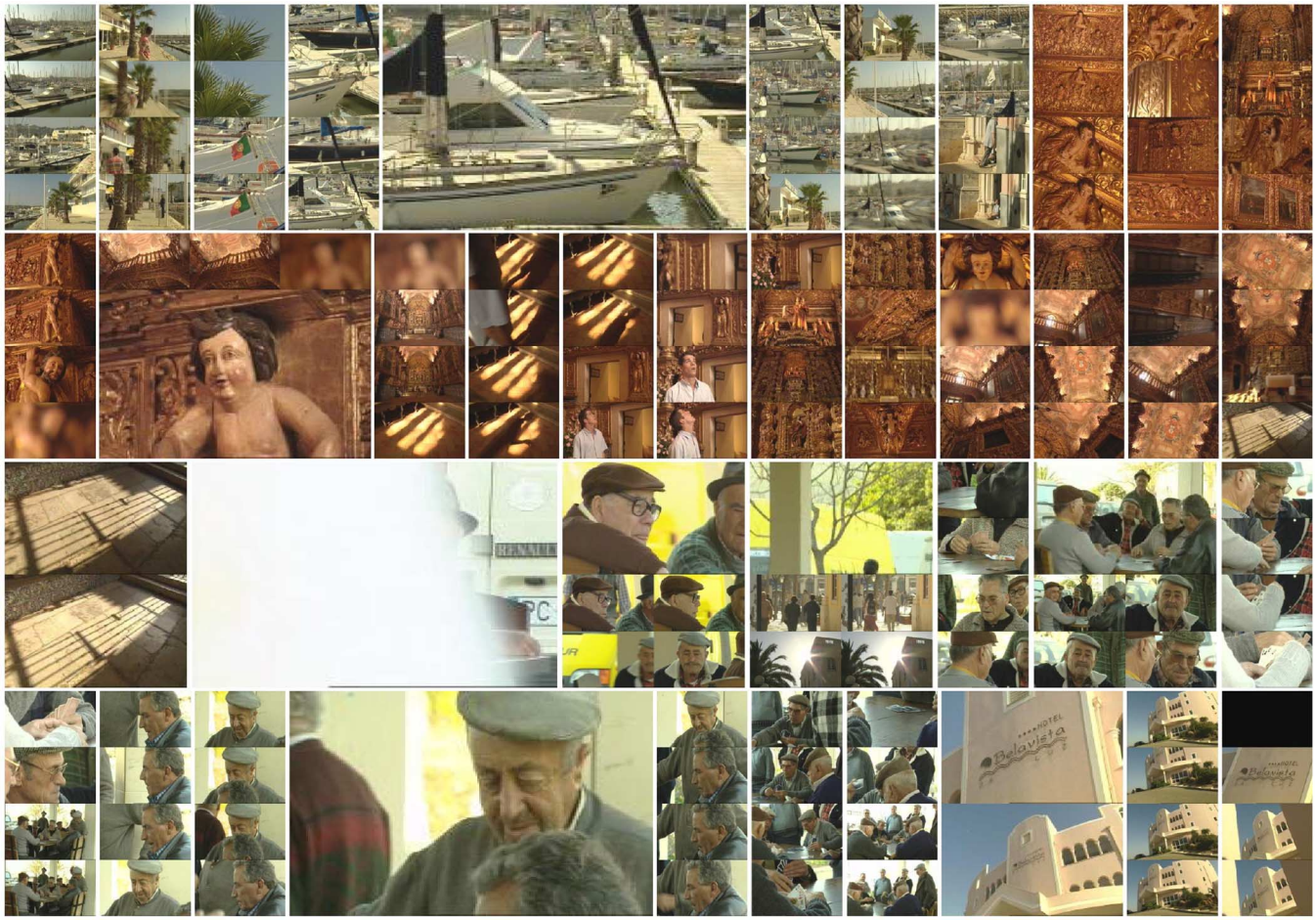

Fig. 3. Sequence from the TRECVID 2006 rushes corpus, summarized using layout parameters $\mathcal{N}$ =150 and $H/W = 1$. Since there is a lot of repetition of the content, this type of data fully exploits functionality of the presented system: the largest frames represent the most frequent content and in some cases extreme outliers (e.g., a capture error due to an obstacle in row 3, frame 3); middle sized frames represent similar, but a bit different content to the group represented by the largest frames; the smallest frames are simple repetitions of the the content represented by the largest frames.

As expected, error generally drops as both $h_{\max}$ and $\mathcal{N}$ rise. By having more choice of combinations for panel templates with bigger $h_{\max}$, the cost function can be approximated more accurately. In addition, the effect of higher approximation error has less impact as number of frames per page $\mathcal{N}$ rises. As we described in Section III, the reason behind this phenomenon is the finite page width, that results in suboptimal solution of the DP algorithm. On the other hand, the approximation error rises with $h_{\max}$ for lower values of $\mathcal{N}$, due to a strong boundary effect of our suboptimal solution for small values of W.

The first three columns of Table II show the approximation error of the optimal method, while the other three columns show absolute difference between errors of the optimal and suboptimal solutions. Due to the high complexity of the optimal algorithm, only page layouts with up to 120 frames per page have been calculated. The overall error due to the suboptimal model is on average smaller than 0.5% of the value of cost function. Therefore, the error can be disregarded and this result shows that the much faster suboptimal solution achieves practically the same results with the optimal method. The optimal algorithm lays out 120 frames on a page in approximately 30 min, while the suboptimal algorithm does it in a fraction of a second (see Table III).

The page layout optimization algorithm is an NP-hard problem. Therefore, the approach presented in [16], as well as our optimal solution, regardless the speedup achieved by various heuristics [8], is not feasible for larger layouts. In [8], the authors limit the size of the final layout to $484(22 \times 22)$. The layout times for the sequence TRECVIDnews.mpg of the algorithms presented in [16] (T_{ORIG}) and [8] (T_{FAST}), compared to the proposed method (T_{DPLY}) are depicted in Fig. 1 and numerically given in Table III.

Examples of two contrasting content types, news broadcast and rushes, from the TRECVID corpus are presented in Figs. 2 and 3. respectively. The news sequence is summarized using layout parameters $\mathcal{N}$ =70 and $H/W = 3/5$. It can be observed that the repetitive content is always presented by the smallest frames in the layout. On the other hand, outliers are presented as big (e.g., a commercial break within a newscast, row 2, frame 11) which is very helpful for the user to swiftly uncover the structure of the presented sequence. Finally, a sequence from the TRECVID 2006 rushes corpus is summarized using layout parameters $\mathcal{N}$ =150 and $H/W = 1$. Since there is a lot of repetition of the content, this type of data fully exploits the functionality of the presented system. The largest frames represent the most frequent content and in some cases extreme outliers (e.g.,

a capture error due to an obstacle in row 3, frame 3); middle sized frames represent slightly different content to the the largest frames, while the smallest frames are simple repetitions of the the content represented by the largest frames.

V. Conclusion

This paper presents a video summarization and browsing algorithm that produces a comic-like representation of analyzed videos. The algorithm exploits the narrative structure of comics and using its well-known intuitive rules, creates visual summaries in an efficient and user centered way.

From the results presented, one can observe that the approximation error introduced by the suboptimal solution is insignificant, whilst the processing is faster, enabling real-time interaction with a long video sequence. The results show that a summary of an hour long video, comprising 250 shots, can be browsed swiftly and easily. Furthermore, the creative process of finding interesting or representative content is significantly augmented using the comic-like layout.

Future work will be directed towards an extension of the summarization algorithm towards interactive representation of visual content. Having the potential to create layouts on various types of displays and a fast system response, this algorithm could be used for interactive search and browsing of large video and image collections.

In addition, frame cropping driven by the saliency of visual content will be investigated, enabling arbitrary shape of frame panels and thus setting more challenging optimization problems. Finally, a set of high-level rules of comics grammar [5] will be learned and exploited to improve representation of time in such a space constrained environment.

References

[1] Z. Yueting, R. Yong, T. S. Huang, and S. Mehrotra, "Adaptive key frame extraction using unsupervised clustering," in *Proc. ICIP*, 1998, vol. 1, pp. 866–870.

[2] A. Hanjalic and Z. HongJiang, "An integrated scheme for automated video abstraction based on unsupervised cluster-validity analysis," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, no. 8, pp. 1280–1289, Dec. 1999.

[3] D. DeMenthon, V. Kobla, and D. Doermann, "Video summarization by curve simplification," in *Proc. MULTIMEDIA*, New York, 1998, pp. 211–218, ACM Press.

[4] N. Chong-Wah, M. Yu-Fei, and Z. Hong-Jiang, "Video summarization and scene detection by graph modeling," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 2, pp. 296–305, Feb. 2005.

[5] S. Mccloud, *Understanding Comics*. New York: HarperPerennial, 1994.

[6] R. Dony, J. Mateer, and J. Robinson, "Techniques for automated reverse storyboarding," in *Proc. IEE Vis., Image Signal Process.*, 2005, vol. 152, no. 4, pp. 425–436.

[7] S. Uchihashi, J. Foote, A. Girgensohn, and J. Boreczky, "Video manga: Generating semantically meaningful video summaries," in *Proc. MULTIMEDIA*, New York, NY, USA, 1999, pp. 383–392, ACM Press.

[8] A. Girgensohn, "A fast layout algorithm for visual video summaries," in *Proc. ICME*, 2003, vol. 2, pp. 77–80.

[9] A. Y. Ng, M. I. Jordan, and Y. Weiss, *On Spectral Clustering: Analysis of An Algorithm*, pp. 849–856, 2002.

[10] L. Zelnik-Manor and P. Perona, *Self-Tuning Spectral Clustering*, pp. 1601–1608, 2005.

[11] R. E. Bellman and S. E. Dreyfus, *Applied Dynamic Programming*. Princeton, NJ: Princeton Univ. Press, 1962.

[12] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: A review," *ACM Comput. Surv.*, vol. 31, no. 3, pp. 264–323, 1999.

[13] M. Polito and P. Perona, *Grouping Dimensionality Reduction Locally Linear Embedding*, pp. 1255–1262, 2002.

[14] A. Nijenhuis and H. S. Wilf, "Combinatorial algorithms: For computers and calculators," in *Computer Science and Applied Mathematics*, 2nd ed. New York: Academic Press, 1978.

[15] A. Lodi, S. Martello, and M. Monaci, "Two-dimensional packing problems: A survey," *Eur. J. Oper. Res.*, vol. 141, no. 2, pp. 241–252, 2002.

[16] S. Uchihashi and J. Foote, "Summarizing video using a shot importance measure and a frame-packing algorithm," in *Proc. ICASSP*, 1999, vol. 6, pp. 3041–3044.