WHO IS MY PARENT?
RECONSTRUCTING VIDEO SEQUENCES FROM PARTIALLY MATCHING SHOTS

S. Lameri[♯], *P. Bestagini*[♯], *A. Melloni*[♯], *S. Milani*[♯], *A. Rocha*[♭], *M. Tagliasacchi*[♯], *S. Tubaro*[♯]

[♯]Dipartimento di Elettronica, Informazione e Bioingegneria
Politecnico di Milano, Piazza Leonardo da Vinci 32, 20133, Milano, Italy
[♭] Institute of Computing, University of Campinas (UNICAMP)
Av. Albert Einstein, 1251, Cidade Universitária, 13083-852, Campinas, SP - Brazil

ABSTRACT

Nowadays, a significant fraction of the available video content is created by reusing already existing online videos. In these cases, the source video is seldom reused as is. Conversely, it is typically time clipped to extract only a subset of the original frames, and other transformations are commonly applied (e.g., cropping, logo insertion, etc.). In this paper, we analyze a pool of videos related to the same event or topic. We propose a method that aims at automatically reconstructing the content of the original source videos, i.e., the *parent* sequences, by splicing together sets of near-duplicate shots seemingly extracted from the same *parent* sequence. The result of the analysis shows how content is reused, thus revealing the intent of content creators, and enables us to reconstruct a *parent* sequence also when it is no longer available online. In doing so, we make use of a robust-hash algorithm that allows us to detect whether groups of frames are near-duplicates. Based on that, we developed an algorithm to automatically find near-duplicate matchings between multiple parts of multiple sequences. All the near-duplicate parts are finally temporally aligned to reconstruct the *parent* sequence. The proposed method is validated with both synthetic and real world datasets downloaded from YouTube.

Index Terms— Video forensics, video phylogeny, video alignment, near-duplicates detection

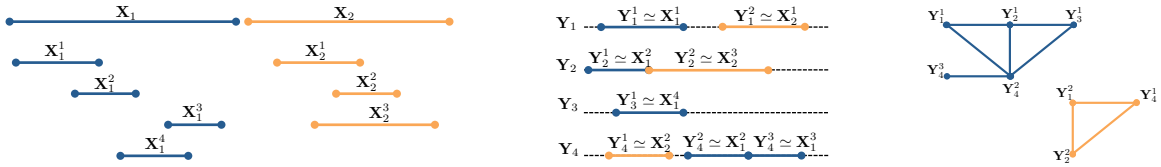
1. INTRODUCTION

With the rapid diffusion of inexpensive video recording devices (smartphones, action cameras, dashboard cameras, etc.), video content acquisition is at everyone's reach. For this reason, over the past years, the amount of user generated content available online has dramatically increased. At the same time, a significant fraction of such content is not originally acquired by content creators. Conversely, content creators can easily reuse video sequences already available online. In this process, the original source video sequences (hereinafter denoted as *parent* sequences) are typically time clipped, so as to extract the subset of video frames of interest, which consists of one or more shots of the original video sequence, possibly not adjacent in time. Then, the extracted shots might be edited further, by applying spatial cropping, resizing and temporal resampling to match the final video format, brightness/contrast adjustment, color correction, logo insertion, etc. Finally, the edited shots extracted from one or more parent sequence are spliced together to create the desired content, which is often compressed before being published online. This content creation pipeline is routinely adopted in the case of newscasts, documentaries and video compilations.

In this paper we analyze sets of video sequences related to the same event or topic. These sequences can be readily obtained by querying one of the many video sharing websites commonly used to post user generated content (e.g., YouTube, Vimeo, etc.). A pool of event- or topic-specific videos can be readily obtained by submitting text-based queries (e.g., "Boston marathon bombing", "Meteorite hits Russia", etc.). Within such a pool, the proposed method detects the reuse of video shots by leveraging a state-of-the-art video fingerprinting method for near-duplicate detection [1] applied to pairs of video sequences. Then, the relationships between all matching shots in the whole pool are further examined, so as to cluster together matching shots that were supposedly extracted from the same parent sequence. Finally, an estimate of each parent sequence is reconstructed, by splicing together the reused shots once they are properly aligned in time. The analysis performed by the proposed method enables to shed very interesting insights on the way content is reused. For example, it is possible to reveal how the original content was edited (e.g., by dropping some of the video frames) before being inserted in a new content, thus explaining the intent of the content creator. This is helpful to understand the semantics behind the processing of the content, which would be otherwise impossible by analyzing a video sequence alone. Moreover, it is possible to reconstruct a parent sequence also when it is no longer available online, e.g., because it has been deleted by the original content owner or it has never been publicly released in its totality.

The proposed method is related to previous work in the field of video fingerprinting / robust hashing [2, 1, 3, 4]. These methods aim at determining the presence of near-duplicate content to support, e.g., content-based video retrieval [5, 6] and content-based video authentication [7]. In the case of videos, retrieval is sometimes performed based on partial near-duplicates [8]. Our proposal exploits state-of-the-art robust hashing, but it goes beyond that, since it analyzes complex relationships within a pool of video sequences, rather than simple pairwise similarities between, e.g., a query video and a database video. In this respect, our work shares similar motivations with recent trends in multimedia forensics [9, 10], whereby pools of near-duplicate audio tracks [11], images [12, 13, 14] and videos [15] were analyzed so as to determine causal relationships between different versions of the same object, which are represented by means of phylogeny trees, somewhat extending early attempts to investigate relationships in groups of video sequences [16]. However, our work considers a much more challenging scenario, since the sequences un-

The project REWIND acknowledges the financial support of the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under FET-Open grant number:268478.



(a) Parent sequences (b) Observed sequences (c) Dependency graph

Fig. 1: (a) A set of possible child sequences is generated from two parent sequences $\mathbf{X}_1$ and $\mathbf{X}_2$. (b) In the real world, every child sequence is possibly edited generating $\mathbf{Y}_k^i$ sequences, which might be spliced together (also with other videos) to obtain other observed sequences $\mathbf{Y}_k$. The symbol $\simeq$ denotes the fact that the left hand side sequences may not be exact copies of the right hand side sequences, since editing operations may have been applied before or after splicing. (c) Our proposed method is applied to identify dependencies between pairs of $\mathbf{Y}_k^i$ sequences and possibly to recover the longer parent sequence.

der analysis are not simple near-duplicates of each other, but might share only part of the content. In addition, each of the available sequences might be the result of splicing together more than one *parent* sequence, thus representing a first step towards the problem of multiple parenting, which has not been addressed in [12, 13, 14, 15]. Note that the techniques developed in [15] can be applied to the output of the method proposed in this paper, when one is interested in determining the causal relationships between different versions of the same shot appearing in the parent sequence.

2. RECONSTRUCTING PARENT SEQUENCES

Let $\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_K$ denote a set of video sequences related to a given event or topic. Each video sequence contains both original content, previously unpublished, and content reused from other video sequences. Let $\mathbf{X}_1, \ldots, \mathbf{X}_P$ denote the parent sequences, i.e., the original source video sequences from which reusable content is extracted. Figure 1 illustrates an example, in which $K = 4$ video sequences were obtained by means of partial reuse of $P = 2$ parent sequences. Note that each available sequence $\mathbf{Y}_k$, $k = 1, \ldots, K$, contains content from one (e.g., $\mathbf{Y}_3$) or more (e.g., $\mathbf{Y}_1, \mathbf{Y}_2, \mathbf{Y}_4$) parent sequences. Let $\mathbf{Y}_k^i$, $i = 1, \ldots, M_k$, denote the i-th shot in $\mathbf{Y}_k$ extracted from one of the parent sequences. In the example in Figure 1, $\mathbf{Y}_1$ contains one shot from the parent sequence $\mathbf{X}_1$ ($\mathbf{Y}_1^1 \simeq \mathbf{X}_1^1$) and one from the parent sequence $\mathbf{X}_2$ ($\mathbf{Y}_1^2 \simeq \mathbf{X}_2^1$). Note that we use the symbol $\simeq$ to denote the fact that the left hand side sequence is a near-duplicate of the right hand side sequence, i.e., the original content might have undergone different kinds of content-preserving transformations (e.g., spatial cropping/resizing, contrast/brightness adjustment, compression, etc.).

The proposed method starts from the analysis of the available sequences $\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_K$, and aims at reconstructing one, or more, parent sequences from which the content was reused. The key tenet is that the shots extracted from each parent sequence $\mathbf{X}_p$, i.e., $\mathbf{X}_p^1, \ldots, \mathbf{X}_p^{N_p}$, are partially overlapped in time. Therefore the comparison of pairs of available sequences reveals the temporal extent of the common sequence of frames. For example, the comparison between $\mathbf{Y}_1$ and $\mathbf{Y}_2$ is illustrated in Figure 2a, and shows that, in this case, two correspondences between common sequences of frames could be identified, namely $\tilde{\mathbf{Y}}_1^1 \leftrightarrow \tilde{\mathbf{Y}}_2^1$ and $\tilde{\mathbf{Y}}_1^2 \leftrightarrow \tilde{\mathbf{Y}}_2^2$. Then, matching subsequences are temporally extended up to the shot boundaries corresponding to scene cuts ($\tilde{\mathbf{Y}}_k^i \rightarrow \mathbf{Y}_k^i$), as shown in Figure 2b. Finally, each shot is assigned to one parent sequence as indicated in Figure 2c by analyzing the relationships between pairs of shots, and each parent is reconstructed from its constituent shots. In the following, we describe in detail the different algorithmic steps of the proposed method.

Near-duplicate matching: In order to detect whether two video sequences $\mathbf{Y}_{k_1}$ and $\mathbf{Y}_{k_2}$ share common frames (e.g., $\tilde{\mathbf{Y}}_1^1 \leftrightarrow \tilde{\mathbf{Y}}_2^1$ and $\tilde{\mathbf{Y}}_1^2 \leftrightarrow \tilde{\mathbf{Y}}_2^2$ as in Figure 2a), we make use of a robust hashing algorithm previously proposed in [1]. Specifically, each sequence $\mathbf{Y}_k$ is spatially downsampled to 32×32 pixels per frame and split into blocks of $F = 64$ frames each, overlapped by $F - 1$ frames. A 3D Discrete Cosine Transform (DCT) is applied to every block to obtain a set of $32 \times 32 \times 32$ DCT coefficients $c_{p,q,r}$, $p, q, r = 0, 1, \ldots, 31$. The 64 DCT coefficients $c_{p,q,r}$ such that $p, q, r \in [1, 4]$ are extracted (where $c_{0,0,0}$ is the DC coefficient). The binary hash h_k^n computed from the n-th block of sequence $\mathbf{Y}_k$ is composed by a 64-bit string obtained binarising the 64 DCT coefficients with respect to their median value $\bar{c}$. That is, if $c_{p,q,r} > \bar{c}$ then the corresponding hash bit is set to 1, otherwise it is set to 0. Once all the hashes of every block of $\mathbf{Y}_{k_1}$ and $\mathbf{Y}_{k_2}$ are computed, they are compared pair-wise using Hamming distance and stored in a distance matrix D_{k_1,k_2}, whose elements are defined as follows:

$$D_{k_1,k_2}(n_1, n_2) = \sum_{b=1}^{64} h_{k_1}^{n_1}(b) \oplus h_{k_2}^{n_2}(b), \tag{1}$$

where $\oplus$ is the Exclusive Or (XOR) operator. Low values of $D_{k_1,k_2}(n_1, n_2)$ indicate similar hashes, therefore a block of contiguous near-duplicate frames.

Figure 3 shows two distance matrices for the example illustrated in Figure 1. The matrix $D_{1,2}$ (left) refers to the case of matching $\mathbf{Y}_1$ and $\mathbf{Y}_2$, in which two sets of near-duplicate frames are found in both sequences. For each set of near-duplicate frames, low values in the $D_{1,2}$ matrix are found to be aligned along a segment. Note that, the higher the number of matching frames, the longer is the segment. Moreover, the angular coefficient of the line depends on the difference in frame rates between the two analyzed sequences and it corresponds to an angle of -45 degrees when the same frame rate is used in both sequences. Conversely, the matrix $D_{1,3}$ (right) refers to the case of two sequences that do not contain any near-duplicate frame, namely $\mathbf{Y}_1$ and $\mathbf{Y}_3$.

Given a matrix D_{k_1,k_2}, we automatically detect the presence of one or more matching sets of near-duplicate frame with the following steps:

1. Binarize the distance matrix D_{k_1,k_2}. That is, $\overline{D}_{k_1,k_2}(i, j) = 1$, if $D_{k_1,k_2}(i, j) \leq \tau$, and 0 otherwise, where τ is a threshold that is set to discriminate between near-duplicate and non-near-duplicate blocks of video frames (Figure 4a).
2. Apply a morphological opening to $\overline{D}_{k_1,k_2}$ to discard spurious areas of local low values and identify the connected components (Figure 4b).

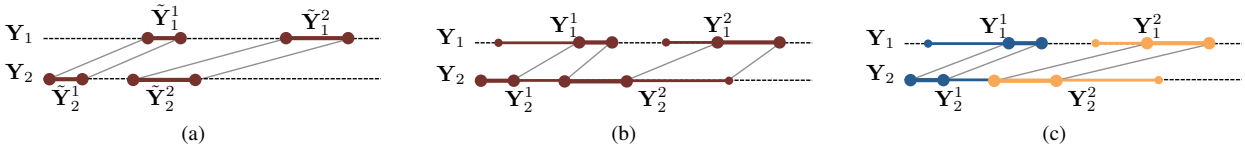


Fig. 2: An example showing the near-duplicate matching, alignment and shot extraction when comparing two sequences.

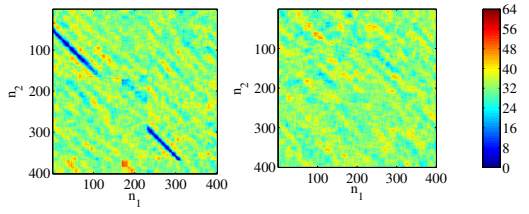


Fig. 3: Distance matrices $D_{k_1,k_2}(n_1, n_2)$. Pairs of sequences containing (left) or not containing (right) near-duplicate subsequences.

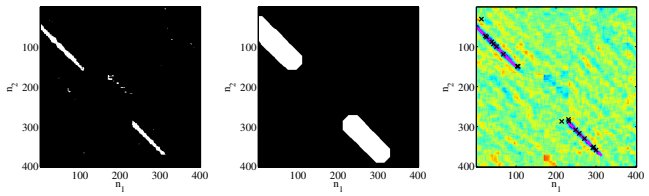


Fig. 4: Examples of $\overline{D}_{k_1,k_2}(n_1, n_2)$ before (a) and after (b) morphological opening. (c) Estimated segments representing matching subsequences.

3. For each connected component, analyze the underlying values of D_{k_1,k_2} and determine the coordinates of the minimum value along every row and column (indicated as black crosses in Figure 4c). Then, fit a line passing through these minima.

Note that, since the analysis is performed on every connected component in Step 2, the algorithm is able to find more than one segment in each matrix, as illustrated in Figure 4c for the pair $\mathbf{Y}_1$ and $\mathbf{Y}_2$.

Near-duplicate extraction: The analysis of the segments identified at the previous step reveals which parts of the sequences $\mathbf{Y}_{k_1}$ and $\mathbf{Y}_{k_2}$ are near-duplicates. Let us denote with (s_1, s_2) the start point of a segment, and with (e_1, e_2) its end point. This means that the frames of $\mathbf{Y}_{k_1}$, $\{\mathbf{Y}_{k_1}(s_1), \ldots, \mathbf{Y}_{k_1}(e_1 + 63)\}$, are near-duplicates of the block of frames of $\mathbf{Y}_{k_2}$, $\{\mathbf{Y}_{k_2}(s_2), \ldots, \mathbf{Y}_{k_2}(e_2 + 63)\}$, where the constant value 63 considers the fact that hashes are computed on blocks of 64 frames. As indicated in Figure 2a, two correspondences are identified when considering the pair of sequences $\mathbf{Y}_1$ and $\mathbf{Y}_2$, namely $\tilde{\mathbf{Y}}_1^1 \leftrightarrow \tilde{\mathbf{Y}}_2^1$ and $\tilde{\mathbf{Y}}_1^2 \leftrightarrow \tilde{\mathbf{Y}}_2^2$. In general, $\tilde{\mathbf{Y}}_k^i$ denotes the i-th subsequence in sequence $\mathbf{Y}_k$, which is found to be matching with another subsequence. Without loss of generality, we assign the index i based on the index of the first frame in $\tilde{\mathbf{Y}}_k^i$.

Near-duplicate alignment: Once $\tilde{\mathbf{Y}}_{k_1}$ and $\tilde{\mathbf{Y}}_{k_2}$ are extracted, we perform an additional step of fine temporal alignment between the shots. Indeed, the estimation of the start and end points of a segment from the matrix D_{k_1,k_2} is affected by noise due to different sources: i) the limited temporal resolution of the hash, which groups together blocks of 64 frames; and ii) the uncertain localization of local minima in D_{k_1,k_2}. To this purpose, we resort to a monodimensional description of a video over time, inspired by the work in [17]. More specifically, given an arbitrary video sequence $\mathbf{Y}$, we compute the difference between the average luminance of adjacent frames as

$$l(n) = \text{avgluma}(\mathbf{Y}(n)) - \text{avgluma}(\mathbf{Y}(n-1)), \qquad (2)$$

where $\text{avgluma}(.)$ extracts the average of the luminance component of a frame. The alignment between two subsequences $\tilde{\mathbf{Y}}_{k_1}^{i_1}$ and $\tilde{\mathbf{Y}}_{k_2}^{i_2}$ containing a matching near-duplicate subsequence is performed by looking at the position of the highest peak of the phase-correlation between $l_{k_1}^{i_1}$ and $l_{k_2}^{i_2}$. That is,

$$n_{k_1,k_2}^{i_1,i_2} = \arg\max_n \mathcal{F}^{-1}\left[\frac{L_{k_1}^{i_1} \cdot L_{k_2}^{i_2 *}}{|L_{k_1}^{i_1} \cdot L_{k_2}^{i_2 *}|}\right](n), \qquad (3)$$

where $L_k^i = \mathcal{F}[l_k^i]$ denotes the Fourier transform, '$*$' indicates the complex conjugate operator, and '$\cdot$' the element-wise product.

Shot extraction: Given a near-duplicate matching shot $\tilde{\mathbf{Y}}_k^i$, we want to identify the shot in $\mathbf{Y}_k$ that includes $\tilde{\mathbf{Y}}_k^i$. To this end, we extend $\tilde{\mathbf{Y}}_k^i$ to $\mathbf{Y}_k^i$, so that the latter encompasses a contiguous set of frames including $\tilde{\mathbf{Y}}_k^i$ and delimited by scene changes. This is illustrated in Figure 2b for the shots extracted from $\mathbf{Y}_1$ and $\mathbf{Y}_2$ in our example. In our work, we identify scene changes when the average luminance suddenly changes from a frame to another. This is achieved by thresholding $l(n)$. Note that this step is necessary if we want to reconstruct the parent sequence starting from partially overlapping shots. For example, shots $\mathbf{X}_1^2$ and $\mathbf{X}_1^3$ would be considered to belong to different parent sequences if we neglected this simple, yet important step. Indeed, it is the reuse of shot $\mathbf{X}_1^4$ that provides the necessary link between $\mathbf{X}_1^2$ and $\mathbf{X}_1^3$.

Parent reconstruction: Once all the reused shots $\mathbf{Y}_k^i$ have been identified and temporally aligned with respect to each other, we need to cluster together shots belonging to the same parent sequence. To this purpose, we build a graph where every node represents a shot $\mathbf{Y}_k^i$, and we link together pairs of nodes whose corresponding shots share a common subsequence. Then, the topology of the resulting graph is analyzed so that each connected component is assigned to a parent sequence. For example, Figure 1 (right) shows the graph built considering all the shots in $\mathbf{Y}_1$, $\mathbf{Y}_2$, $\mathbf{Y}_3$ and $\mathbf{Y}_4$. Two connected components are identified, one for each parent sequence.

In order to reconstruct each parent sequence, for each connected component of the graph, we select as root the node with the highest degree, and run a Depth-First Search (DFS) algorithm to find an acyclic path traversing the graph. Shots are then re-aligned with the root according to the selected path.

3. EXPERIMENTS AND RESULTS

In order to validate the proposed method, we considered three datasets. The first dataset was used to evaluate individually the different steps of the method and to determine the values of the configuration parameters. This dataset consists of a pool of near-duplicates generated starting from eight sequences of 300 frames each at CIF spatial resolution, namely: *city*, *crew*, *foreman*, *hall*,

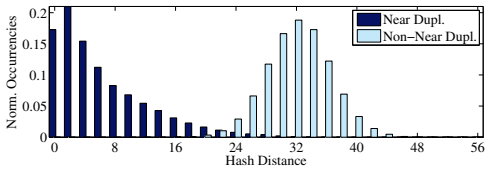


Fig. 5: Histogram of hash distances between near-duplicate (dark blue) and non-near-duplicate (light blue) blocks of video frames.

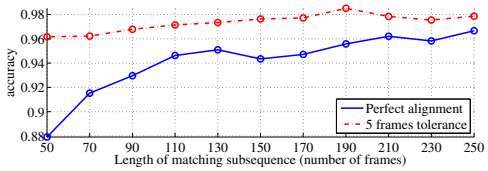


Fig. 6: Near-duplicate alignment accuracy when varying the length of the matching subsequence.

mother, *news*, *paris*, and *soccer*. We applied one of the following transformations to each sequence: blurring, brightness adjustment, contrast enhancement, spatial cropping, AVC/H.264 coding, logo insertion, and rotation. By varying the parameters used for these transformations, we obtained a total number of 424 test sequences[1]. In order to test the temporal alignment, we also trimmed the sequences by dropping frames from the leading and trailing part.

The second dataset was used to test the accuracy of the whole reconstruction algorithm. The dataset consists of four parent sequences, namely *sign irene*, *highway*, *mother* (long version) and *students*, with more than 540 frames each. Then, we created $K = 6$ sequences $\mathbf{Y}_1, \ldots, \mathbf{Y}_6$ obtained by splicing together (in random order) six shots: one shot from each parent sequence (processed with the same kind of transformations used in the first dataset), and two shots taken from an external set of sequences. Each $\mathbf{Y}_k$ had a total length between 800 and 1700 frames. The set of shots extracted from each parent sequence is such that, for each pair of shots, they overlap for at least 90 frames (3 seconds at 30fps). We repeated this process 100 times so as to compute averaged results over several realizations.

Finally, the third dataset includes real-world sequences downloaded from YouTube representing videos related to different events. More specifically we considered $K = 9$ sequences related to the "Boston marathon bombing" event, and $K = 8$ related to the "Meteorite hits Russia" events. Manual inspection revealed the presence of one and three parent sequences, respectively.

The robust hashing algorithm adopted in our work was evaluated on the first dataset. Figure 5 shows the histogram of hash distances computed between all the near-duplicate (dark blue) and non-near-duplicate (light blue) blocks of the synthetically created sequences. From this study, we set $\tau = 16$ and tested the algorithm adopted to extract segments corresponding to matching subsequences from the analysis of matrices D_{k_1,k_2}. In this controlled dataset, the correct alignment is known and represented as a ground truth segment that can be compared with the segment estimated by our algorithm. A point in the (n_1, n_2)-space is considered to be a true positive, if it belongs to the ground truth segment and lies at a distance equal to at most three frames from the estimated segment in both directions. The true positive rate is obtained by dividing this number by the number of points that belong to the ground truth segment. The false positive rate is defined similarly, with the role of ground truth and estimated segments reversed. In our experiments, we measured a false positive rate equal to 0.0001, that is, virtually all the non-near-duplicate blocks were correctly identified as such. The true positive rate was equal to 0.85, indicating that the vast majority of near-duplicate blocks were correctly identified. We observed that this was due to the fact that the estimated segment was slightly shorter, on average, than the ground truth segment. However, this is not a major issue in the overall method, since the start and end points of the estimated segment are then extended up to the nearest scene change.

[1]Additional material available at: `http://tinyurl.com/oomvaxt`

The near-duplicate alignment algorithm was evaluated by investigating its accuracy when considering matching subsequences of variable length. This is illustrated in Figure 6, which expresses the accuracy as a function of the length of the matching subsequence. The accuracy indicates the fraction of times two sequences are exactly aligned. Figure 6 also shows the case in which an error of 5 frames is tolerated. The results confirm the intuition that the longer the matching subsequence, the more accurate the alignment. In the two cases, when two sequences share more than 90 frames, the alignment was correctly estimated in more than, respectively, 93% (perfect alignment) and 96% (5% tolerance alignment) of the tested cases.

In order to evaluate the overall parent reconstruction method, we considered each of the $K = 6$ sequences $\mathbf{Y}_k$ and checked whether the extracted shots were correctly assigned to the corresponding parent sequences. For each of the 100 realizations, we had six shots belonging to each one of the $P = 4$ parent sequences. We evaluated the percentage of times we were able to perform the correct assignment between shots and parents. It turned out that we were able to assign all the six shots to their parent sequence in 85% of the cases. On the other hand, in more than the 90% of the cases we were able to assign at least five out of six shots to each parent.

Finally, we also tested the parent reconstruction method on the real-world dataset of sequences downloaded from YouTube. In the "Boston marathon bombing" event we extracted two parent sequences, one of which of 1005 frames by automatically aggregating four shots, each having between 111 and 600 frames. Similarly, in the "Meteorite hits Russia" event, three parent sequences were identified from eight observed sequences. To visually illustrate the output produced by the proposed method, we invite the reader to watch the reconstructed parent sequences that we made available online[1].

The possibility of reconstructing parent sequences in a real-world scenario allows us to increase the temporal coverage of an event under analysis. This can be useful to provide additional information related to an event, e.g., in case of criminal investigations. Moreover, this gives an interesting insight on the way content is reused, e.g., to understand whether newscasts decided to broadcast only partial news related to important events.

4. CONCLUSIONS

We presented a method for the reconstruction of one or more parent sequences, given a set of partially overlapped near-duplicate video shots reused in other sequences. The use of the proposed method allows one to study how content is reused and to recover original content when it is no longer available. Future work will be devoted to the study causal relationships between reused shots, thus extending video phylogeny to address multiple parenting.

Future work will focus on studying causal relationships between reused videos and extending upon our current solution to incorporate video phylogeny analysis as well as multiple parenting relationship refinements.

5. REFERENCES

[1] B. Coskun, B. Sankur, and N. Memon, "Spatio-temporal transform based video hashing," *IEEE Transactions on Multimedia*, vol. 8, pp. 1190–1208, 2006.

[2] J. C. Oostveen, T. Kalker, and J. Haitsma, "Visual hashing of digital video: applications and techniques," in *SPIE workshop on Applications of Digital Image Processing (ADIP)*, 2001.

[3] J. Song, Y. Yang, Z. Huang, H. T. Shen, and R. Hong, "Multiple feature hashing for real-time large scale near-duplicate video retrieval," in *ACM international conference on Multimedia (MM)*, 2011.

[4] H.-S. Min, J.-Y. Choi, W. De Neve, and Y.-M. Ro, "Near-duplicate video clip detection using model-free semantic concept detection and adaptive semantic distance measurement," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, pp. 1174–1187, 2012.

[5] J.R. Zhang, J.Y. Ren, Fangzhe Chang, T.L. Wood, and J.R. Kender, "Fast near-duplicate video retrieval via motion time series matching," in *IEEE International Conference on Multimedia and Expo (ICME)*, 2012.

[6] J. Song, Y. Yang, Z. Huang, H. T. Shen, and J. Luo, "Effective multiple feature hashing for large-scale near-duplicate video retrieval," *IEEE Transactions on Multimedia*, vol. 15, pp. 1997–2008, 2013.

[7] S. Lee, C.D. Yoo, and T. Kalker, "Robust video fingerprinting based on symmetric pairwise boosting," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, pp. 1379–1388, 2009.

[8] H-K. Tan, C. W. Ngo, and T.-S. Chua, "Efficient mining of multiple partial near-duplicate alignments by temporal network," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 20, pp. 1486–1498, 2010.

[9] A. Piva, "An overview on image forensics," *ISRN Signal Processing*, vol. 2013, pp. 22, 2013.

[10] S. Milani, P. Bestagini, M. Tagliasacchi, and S. Tubaro, "Multiple compression detection for video sequences," in *IEEE International Workshop on Multimedia Signal Processing (MMSP)*, 2012.

[11] L. Nucci, M. Tagliasacchi, and S. Tubaro, "A phylogenetic analysis of near-duplicate audio tracks," in *IEEE International Workshop on Multimedia Signal Processing (MMSP)*, 2013.

[12] Z. Dias, A. Rocha, and S. Goldenstein, "First steps toward image phylogeny," in *IEEE International Workshop on Information Forensics and Security (WIFS)*, 2010.

[13] A. De Rosa, F. Uccheddu, A. Piva, M. Barni, and A. Costanzo, "Exploring image dependencies: A new challenge in image forensics," in *SPIE Conference on Media Forensics and Security (MFS)*, 2010.

[14] Z. Dias, A. Rocha, and S. Goldenstein, "Image phylogeny by minimal spanning trees," *IEEE Transactions on Information Forensics and Security*, vol. 7, pp. 774–788, 2012.

[15] Z. Dias, A. Rocha, and S. Goldenstein, "Video phylogeny: Recovering near-duplicate video relationships," in *IEEE International Workshop on Information Forensics and Security (WIFS)*, 2011.

[16] J. R. Kender, M. L. Hill, A. P. Natsev, J. R. Smith, and L. Xie, "Video genetics: A case study from youtube," in *ACM International Conference on Multimedia (MM)*, 2010.

[17] Q. Xie, Z. Huang, H. T. Shen, X. Zhou, and C. Pang, "Efficient and continuous near-duplicate video detection," in *International Asia-Pacific Web Conference (APWEB)*, 2010.