VIDEO CONTENT IDENTIFICATION USING THE VITERBI ALGORITHM

Sitaram Bhagavathy, Wen Chen, Dekun Zou, and Jeffrey Bloom

Dialogic Media Labs
12 Christopher way, Eatontown, NJ 07724, USA
{sitaram.bhagavathy, wen.chen, dekun.zou, jeffrey.bloom}@dialogic.com}

ABSTRACT

We propose a video content identification that uses the Viterbi algorithm for identifying unknown video content from a database of reference content. Both the source video in the database and exact frame index therein of each query frame are determined by our method. The Viterbi algorithm is effective for enforcing temporal consistency over frame-level hypotheses based on matching frame fingerprints. We have introduced an "unknown" state in the Viterbi state model to handle both missed detections and non-existent matches. Another novelty of our approach is the use of frame time stamps for computing transition probabilities between frames. This makes our method capable of handling even extreme frame rate changes between query and reference videos. We have provided experimental results demonstrating the effectiveness of the proposed method. This approach may conceivably be used for audio content identification as well.

Index Terms— Video content identification, video fingerprinting, video copy detection

1. INTRODUCTION

Video *fingerprints* are identifiers or "signatures" that are extracted from a piece of video content (usually a frame). Typically, in order to identify a piece of *query* content, fingerprints extracted from it are used to search against fingerprint information stored in a database of *reference* content. The content is then "identified" or not based on how well its fingerprints match with those in the database.

There are many video content identification (VCID) algorithms that use fingerprints (see [1] for a survey). Most of these use spatial fingerprints extracted from individual frames. However, VCID solely based on frame fingerprint matches leads to highly ambiguous and unreliable results due to content variability. Therefore, due consideration has to be given to the temporal consistency among frame-level matches between two pieces of content (see Fig. 1).

Some works try to enforce temporal consistency by considering a temporal neighborhood while matching spatial fingerprints. In [2], when matching a query frame with a reference frame, average L_1 distance is computed between all fingerprints in a temporal window around both frames. Lee et al. [3] use a two-stage matching approach wherein the first stage matches frame-wise query fingerprints against a database to find possible matching positions, and the second stage operates on this reduced search space and computes a distance measure involving a temporal window. Note that only short-term temporal consistency can be enforced by matching fingerprints in a temporal window. Other drawbacks include computational complexity and increased dimensionality of the search space (which makes indexing for efficient search a challenging problem).

A more effective approach for enforcing long-term temporal consistency is to first find potential matches for a set of query frames, and then temporally fuse these results. Hua et al. [4] first determine rough positions of matches by using a temporal fingerprint matching measure, and then apply a dynamic programming (Needleman-Wunsch) algorithm for more accurate alignment. Liu et al. [5] search a database to retrieve a set of matching key-frames and then compute a relevance score for each reference video by evaluating the temporal consistency between matched key-frame pairs. Hampapur et al. [6] accumulate the frame matches of the whole query video and then post-process the results to check for temporal ordering consistency with a rank order correlation coefficient. Joly et al. [7] also accumulate matches for a fixed number of key-frames and perform a robust parametric estimation to find a simple temporal relationship between query and reference key-frames. These methods however are not suitable for real-time applications since they apply temporal fusion only after the results for a batch of frames (and in some cases, the whole query video) are available. One approach that applies temporal fusion on a continuous basis (i.e. with each incoming query frame) is that of Gengembre et al. [8]. This method uses a probabilistic Markovian framework to fuse the results of individual key-frame searches.

We propose a reliable and dynamic VCID method suitable for real-time applications. It is based on the well-known Viterbi algorithm [9] which is effective for tracking temporal relationship between observations. The approach is to first search for candidate matches (from a database) to each query frame based on frame fingerprints. Viterbi is then applied to determine the best sequence of matches among the can-

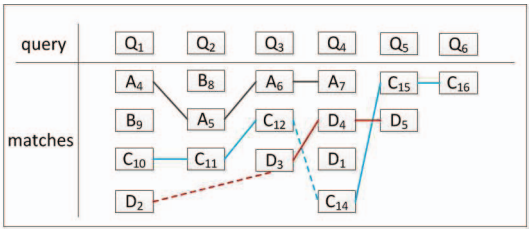


Fig. 1. Video content identification from frame-level fingerprints.

didates taking their temporal consistency into consideration. Note that the algorithm in [8] bears a resemblance to Viterbi. Our approach has several functional differences from that of [8], including the way we compute transition probabilities in order to be robust to frame rate changes between query and reference videos.

2. VIDEO CONTENT IDENTIFICATION

Suppose we have a database of various reference videos along with fingerprints extracted from each frame. When an unknown (and possibly degraded) query video is encountered, the video content identification (VCID) task is to determine which reference video (identified by a *refId*) it corresponds to and also which frames (each identified by a *frame index*) therein. If there is no match, this fact must also be inferred.

The information we are provided for this task is a set of fingerprint vectors, one for each frame of the query video and all the reference videos in the database. For each frame of the query video Q_t, we may determine a set of matching frames from the database based on fingerprint distances, as illustrated by Fig. 1. Note that A_4 refers to frame index 4 from a reference video with *refId* = A. The problem is to determine for each query frame the corresponding *refId* and frame index from the database.

Let $(refId_t, FI_t)$ be the "identified" reference video and frame index for a query frame Q_t at time t. Given a set of query frames $\{Q_t|t = 1, \ldots, T\}$, we need to determine the set $\{(refId_t, FI_t)|1, \ldots, T\}$. The Viterbi algorithm [9] is applied to this task. A brief description of Viterbi is as follows.

2.1. The Viterbi algorithm

Consider a Hidden Markov Model (HMM) with states S, initial probabilities π_i for each state i, and the transition probabilities $a_{i,j}$ of the system transitioning from state i to state j. Suppose we observe a sequence of outputs $\{x_1, \ldots, x_T\}$. The state sequence $\{s_1, \ldots, s_T\}$ that best "explains" the observations may be derived by the following recurrence relations:

$$\begin{array}{rcl} V_{1,k} & = & \pi_k P(x_1|k) \\ V_{t,k} & = & P(x_t|k) \max_{s \in S} (a_{s,k} V_{t-1,s}) \end{array} \quad (1)$$

These are the Viterbi *update* equations. $V_{t,k}$ is the probability of the most probable sequence (or "path") of states ending with state k and taking into account the first t observations.

In order to later reconstruct this best path (known as the *Viterbi path*), back pointers are used to remember which state s was used in the second equation during each update step. Let $prev(k, t)$ be a function that returns the value of s used to compute $V_{t,k}$ if $t > 1$, or k if $t = 1$. Then the Viterbi *back-tracking* equations are:

$$\begin{array}{rcl} s_T & = & \arg\max_{s \in S} V_{T,s} \\ s_{t-1} & = & prev(s_t, t) \end{array} \quad (2)$$

2.2. Video content identification using Viterbi

In order to apply Viterbi, we first need to define the set of states S, their prior probabilities π_i, the observation probabilities $P(x_t|\text{state } k)$, and the state transition probabilities $a_{i,j}$ of transitioning from state i at time t to state j at time $t + 1$. Let us define each frame index f from each reference video, say *refId* = M, as a possible state M_f. For example, in Fig. 1, each match $\{A_4, B_9, C_{10}, D_2\}$ for query frame Q_1 is one state. In addition to all possible matches, an "unknown" state Y_u is also considered as a "match" for each Q_t. This is necessary to robustly handle non-existent or missing matches and ambiguity. A miss occurs when a correct match exists and is not found. Non-existence of a match means the matching frame is not present in the database. Ambiguity is present when there are multiple matches that are similar to each other.

The prior probabilities of all states are considered to be the same, i.e. all possible matches are equally probable to begin with. In this case, the actual probability value does not influence the final result. Therefore, in a practical implementation, all prior probabilities may be set to 1. The observation probabilities are defined in terms of fingerprint distances. Let $x_t = F(Q_t)$ be the observed fingerprint (or feature vector) of Q_t and k be one of its matches. Then, one way to define the observation probability $P(x_t|k)$ is based the distance between the fingerprints, as follows:

$$P(x_t|k) = \begin{cases} \frac{1}{K} e^{-\alpha d(x_t, F(k))}, & k = M_f \\ \frac{1}{K} e^{-\alpha D}, & k = Y_u \end{cases} \quad (3)$$

where K is a normalization factor, $d(.,.)$ is defined to be the Euclidean distance between the two vectors, and D is some large distance. α is a parameter that controls the rate at which the probability drops with the distance. The actual value of K does not affect the final result and may be set to 1.

The most tricky part for our application of Viterbi is the definition of state transition probabilities. Let i be a match at time t and j be a match at time $t+1$. Then, the state transition probabilities $a_{i,j}$ are defined as follows.

1. If $i = Y_u$ and $j = Y_u$, $a_{i,j} = p_{uu}$ (transition from unknown to unknown),
2. If $i = Y_u$ and $j = M_g$, $a_{i,j} = p_{uk}$ (transition from unknown to known),

3. If $i = M_f$ and $j = Y_u$, $a_{i,j} = p_{ku}$ (transition from known to unknown),
4. If $i = M_f$ and $j = N_g$ and $M \neq N$, $a_{i,j} = p_{MN}$ (transition from one *refId* to another), and
5. If $i = M_f$ and $j = M_g$, $a_{i,j} = trans(f, g)$ (transition from one frame index to another).

Since, in most practical applications, video content has a degree of continuity (even with editing), the *refId* does not change very often from one frame to the next. Therefore, p_{MN} has to be a relatively small value. A table summarizing the transition probabilities is given below.

i \ j	$N_g(M \neq N)$	M_g	Y_u
M_f	$p_{MN}(=0)$	$trans(f,g)$	$p_{ku}(=0.01)$
Y_u	$p_{uk}(=0.01)$	$p_{uk}(=0.01)$	$p_{uu}(=0.5)$

Table 1. Definition of transition probabilities with typical values in parantheses.

If the frame indices are defined as *presentation time stamps*[1] of the query and reference frames, this information could be used to define transition probabilities agnostic of frame rate differences. The time stamp difference between the matches to two query frames have to be approximately the same as the time stamp difference between the query frames. Thus, the transition probability may be computed as follows:

$$trans(f,g) = \begin{cases} p_{\text{trans}}, & \text{if } |g - f - \Delta ts| < \epsilon \\ 0, & \text{otherwise} \end{cases} \quad (4)$$

where ϵ and p_{trans} are pre-specified values, with the latter being a fairly high probability. When the frame indices correspond to *frame numbers*, if the frame rate and time stamp clock frequency are known, frame numbers can be converted to time stamps, otherwise, the transition probability may be computed belows:

$$trans(f,g) = \begin{cases} e^{-\alpha(g-f)}, & 0 < (g-f) \leq N \\ p_{\text{far}}, & (g-f) > N \end{cases} \quad (5)$$

in which p_{far} and *N* are predetermined values.

Having all the pieces in place, the Viterbi algorithm may be applied to obtain the optimal sequence of matches $\{s_1, \ldots, s_T\}$ given the fingerprints $\{x_1, \ldots, x_T\}$ for the query frames $\{Q_1, \ldots, Q_T\}$. A measure of confidence regarding the optimal match sequence obtained after applying Viterbi may be computed based on (2) as $C_T = \max_s V_{T,s}$.

3. EXPERIMENTAL RESULTS

In order to test the performance of the proposed VCID method, we created a reference database of 100 video sequences (around 57,000 frames, CIF and 1280 $\times$ 720 resolutions) that are H.264 encoded with a high quality (i.e. QP $\leq$ 20 and frame rate of 30 fps). For each frame, the database stores the *refId*, time stamp, and a fingerprint extracted from it. We use the ordinal fingerprint proposed in [2]. In order to realize a dynamic real-time system, Viterbi back-tracking (see (2)) is made optional. At any time T, the result based on the current most likely state (s_T in (2)) is retained. Alternatively, a look-ahead window (of duration t_W) may be used and the result for a frame at time t may be queried after performing the update step for frames upto $t + t_W$ and back-tracking from there. The Locality-Sensitive Hashing (LSH) scheme [10] is used to index frame fingerprints for efficient search in the database.

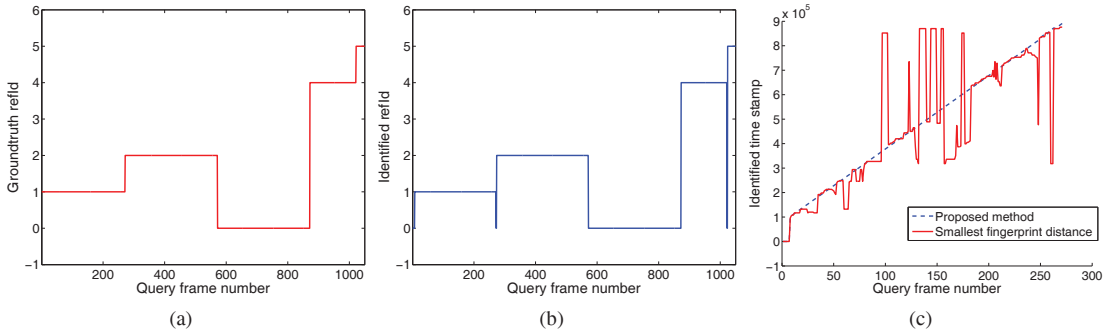
The *query video* is created by concatenating 113 videos, including lower quality versions of all the reference videos (QP $\geq$ 30, frame rates of 30/15 fps), and 13 interspersed videos that do *not* exist in the database. In order to evaluate our VCID performance, we compute the following metrics:

1. *False positive rate* (*FPR*): Percentage of query frames that are falsely "identified" even though they do *not* exist in the database (i.e. reported *refId* > 0 and actual *refId* = 0).
2. *False negative rate* (*FNR*): Percentage of query frames that are classified as "unknown" even though they exist in the database (reported *refId* = 0, actual *refId* > 0).
3. *Sequence-level precision rate* (*SLPR*): Percentage of query frames that are correctly identified with regard to the reference video (*refId*) they correspond to.
4. *Frame-level precision rate* (*FLPR*): Percentage of query frames that are correctly identified with regard to the exact frame index (time stamp or frame number) in the reference video, among those whose *refId* is correctly identified.

Fig. 2 shows some illustrative results of the proposed approach. Fig. 2(b) shows the identified *refId*s for a range of frames in the query video. Note that these match the expected (*groundtruth*) *refId*s shown in Fig. 2(a), except for a few *transitional* frames (i.e. when the content changes from one reference video to another) where *refId* defaults to 0. By setting p_{MN} to 0 (see Table 1), the algorithm errs on the side of caution and defaults to "unknown" in transitional regions until it can return a confident match. Fig. 2(c) shows the advantage of our approach in enforcing temporal consistency of matched frames. Compare the estimated time stamps for a range of query frames using the match with the smallest fingerprint distance (in red) and those using the proposed method (in blue). The latter is monotonically increasing (as it should be) and is consistent with the groundtruth time stamps.

Table 2 shows the overall performance of the proposed method. Note that our experiments cover cases wherein the query videos are of lower encoded quality, resolution and frame rate as compared to the corresponding reference videos.

[1]Most compressed video formats contain a presentation time stamp for each frame specifying the actual time instant when that frame is to be displayed relative to other frames. Its unit is defined based on a given time stamp clock frequency.



(a) (b) (c)

Fig. 2. Illustrative results of the proposed method: (a) Groundtruth reference videos (shown by their *refIds*), (b) Identified reference videos, and (c) Comparison between time stamps identified by the proposed method and by using the smallest fingerprint distance.

Query video	*FPR*	*FNR*	*SLPR*	*FLPR*
CIF, QP 30, 30 fps	0	0.35%	99.68%	82.60%
CIF, QP 35, 15 fps	0	0.99%	99.10%	82.67%
QCIF, QP 30, 30 fps	0	0.98%	99.15%	81.18%

Table 2. Video content identification results. *FPR*: False positive rate, *FNR*: False negative rate, *SLPR*: Sequence-level precision rate, *FLPR*: Frame-level precision rate. All values are percentages.

Despite these degradations, false negatives are kept below 1% and false positives are entirely avoided. The reference videos corresponding to the query frames are correctly identified with over 99% accuracy (*SLPR*). Even the precision of identification at the level of frame indices (*FLPR*) is quite high (over 80%) despite the ambiguity of frame-level identification in static or low-motion scenes wherein fingerprints change very little (or not at all). Note that frame-level identification may be relatively less important for some common applications, e.g. copy detection.

4. CONCLUSION

In this paper, we have proposed a video content identification method that uses the Viterbi algorithm for identifying unknown video content from a database of reference content. Both the source video in the database and exact frame index therein of each query frame are determined by our method. The Viterbi algorithm is shown to be very effective in enforcing temporal consistency of frame-level matches based on fingerprint distances. We have introduced an "unknown" state in the Viterbi state model to handle both missed detections and non-existent matches. Another novelty of our approach is the use of frame time stamps for computing transition probabilities between frames. This makes our method capable of handling even extreme frame rate changes between query and reference videos, and also affords the ability to process query frames arriving out of order without any buffering.

The experimental results have demonstrated the effectiveness of the proposed method. This method is currently integrated into a dynamic video content identification system that runs significantly faster than real-time. The method may be improved in several ways (in terms of both performance and complexity), including a) predicting match candidates based on previous results, and b) processing only a subset of sampled query frames, either randomly or based on other parameters (e.g. video quality). Finally, note that the proposed method may conceivably be used for audio content identification as well (and indeed for any sequential media).

References

[1] J. Lu, "Video fingerprinting for copy identification: from research to industry applications," in *Proc. SPIE, Media Forensics and Security*, 2009, vol. 7254.

[2] R. Mohan, "Video sequence matching," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1998, vol. 6, pp. 3697–3700.

[3] S. Lee and C. D. Yoo, "Robust video fingerprinting for content-based video identification," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 7, pp. 983–988, July 2008.

[4] X.-S. Hua, X. Chen, and H.-J. Zhang, "Robust video signature based on ordinal measure," in *Proc. IEEE International Conference on Image Processing (ICIP)*, 2004, vol. 1, pp. 685–688.

[5] Z. Liu, T. Liu, and B. Shahraray, "AT&T research at TRECVID 2009: content-based copy detection," in *TRECVID Workshop*, 2009.

[6] A. Hampapur and R. M. Bolle, "Videogrep: Video copy detection using inverted file indices," in *Technical report, IBM research division*, 2002.

[7] A. Joly, O. Buisson, and C. Frelicot, "Statistical similarity search applied to content-based video copy detection," in *Proc. International Workshop on Managing Data for Emerging Multimedia Applications*, April 2005.

[8] N. Gengembre and S. Berrani, "A probabilistic framework for fusing frame-based searches within a video copy detection system," in *Proc. International Conference on Content-based Image and Video Retrieval (CIVR)*, 2008.

[9] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. Inf. Theory*, vol. 13, no. 2, pp. 260–269, 1967.

[10] M. Datar, N. Immorlica, P. Indyk, and V. Morrokni, "Locality-sensitive hashing scheme based on p-stable distributions," in *DIMACS Workshop on Streaming Data Analysis and Mining*, 2003.