A Framework for Handling Spatiotemporal Variations in Video Copy Detection

Chih-Yi Chiu, Chu-Song Chen, and Lee-Feng Chien

***Abstract*—An effective video copy detection framework should be robust against spatial and temporal variations, e.g., changes in brightness and speed. To this end, a content-based approach for video copy detection is proposed. We define the problem as a partial matching problem in a probabilistic model, and transform it into a shortest-path problem in a matching graph. To reduce the computation costs of the proposed framework, we introduce some methods that rapidly select key frames and candidate segments from a large amount of video data. The experiment results show that the proposed approach not only handles spatial and temporal variations well, also reduces the computation costs substantially.**

***Index Terms*—Video copy detection, probability modeling and matching, spatiotemporal analysis.**

I. Introduction

EFFICIENT and effective copy detection techniques are essential for content management and rights protection because they allow platform providers to check the integrity of user uploads, and thereby prevent plagiarism or illegal copying from other web sites. They also help content providers collect royalty payments and protect copyright by tracking the usage and distribution of particular videos. Generally, there are two techniques used for video copy detection: digital watermarking and content-based approach. In this study, we present a novel content-based approach for video copy detection.

A number of image copy detection approaches have been proposed [7][11][13]. In video copy detection, since much more data has to be processed than in image copy detection, the features employed are usually simple and easy to compute. The ordinal signature [3][5][8][14] has thus become a popular means of video copy detection. Hampapur et al. [3] examined several sequence-matching methods based on the motion, ordinal, and color features, and reported that the ordinal signature achieves the best video copy detection performance. Yuan et al. [14] employed a coarse-to-fine strategy in video sequential searching that uses the ordinal signature for coarse searching and the audio feature for fine matching. Hoad and Zobel [4] proposed two very compact features, namely, color-shift and centroid-based motion magnitudes. Approximated string matching, which can deal with slight frame-rate changes, is then employed for near duplicate detection. However, most studies in video copy detection focus on spatial variations (e.g., histogram equalization and frame resizing), while only limited efforts have been made to address temporal variations (e.g., slow motion and fast motion). In practice, temporal variations occur frequently in various video scenarios. For example, the slow motion operation is often used to recapture key moments in sports videos, while the fast motion operation is usually applied to generate condensed video clips for video skimming.

In addition, many methods use a fixed-length sliding window for matching. Searching by a fixed-length window is simple and can be accelerated by histogram pruning [6]. The coarse-to-fine approach proposed by Hua et al. [5] can handle some temporal variations in the fine matching stage. However, since the coarse matching stage employs a fixed-length sliding window, temporally varied video copies are apt to be filtered out in that stage. Kim and Vasudev [8] used adjacent-frame information to enhance the accuracy of video copy detection, but the method also lacks the ability to handle general temporal modifications. Hoad and Zobel [4] computed the color and motion magnitude differences between adjacent frames. However, for a video that is modified by serious temporal changes (e.g., altering the frame rate or video motion speed), the magnitudes are re-amortized in each frame, resulting in a possibly different signature pattern from that of the original video.

The proposed video copy detection approach is based on a general probability framework that can handle spatiotemporal variations. We treat video copy detection as a partial matching problem, and transform it into a shortest-path problem in a matching graph. To improve the efficiency of our framework, we introduce the key frame and candidate segment selection methods that can cull appropriate frames for matching. To our knowledge, this is the first approach that takes slow and fast motions into considerations. Experimental results confirm the effectiveness of the proposed framework.

The remainder of this paper is organized as follows. Section II formulates the video copy detection problem, and Section III provides the probability framework as our solution. In Section IV, we introduce some methods to speed up the detection process. The experimental results are presented in Section V. Then, in Section VI, we give some concluding remarks.

This work was supported in part by the National Digital Archives Program (NDAP, Taiwan) sponsored by the National Science Council of Taiwan under Grant NSC96-2422-H-001-001-.

The authors are with the Institute of Information Science, Academia Sinica, Taiwan. (e-mail: {cychiu, song, lfchien}@iis.sinica.edu.tw).

II. Problem Formulation

Given a query video Q and a target video T, our objective is to find a sequence of matching pairs of frames between them, i.e., each frame in the query video clip matches a frame in the target video clip. We say that the query video clip is a *near duplicate* of some video segments contained in the target video clip. Let Q_1, ... Q_n and T_1, ... T_m be the n and m frames in Q and T, respectively. To find a video segment that is a sub-segment of T, we denote the hypothesis space as follows:

$$\mathbf{H}=\{T_{\theta(1)} \ldots T_{\theta(n)} \mid 0\le\theta(i)\le m \text{ and } \theta(i)\le\theta(j)\}, \quad (1)$$

for $i < j$, $i, j = 1 \ldots n$. θ is a non-decreasing mapping from the integer set $\{1 \ldots n\}$ to another integer set $\{1 \ldots m\}$, the hypothesis space $\mathbf{H}$ contains all n-length sequences, and each frame in a sequence belongs to the target video T. Note that when $\theta(i) = k$, we do not assume that the next frame $\theta(i+1)$ is equal to $k+1$ because we want to handle general situations that contain spatiotemporal variations, such as frame rate changes and slow/fast motion.

Let h be a hypothesis with $h \in \mathbf{H}$. Given a query video clip Q, let us consider the *a posteriori* probability $P(h \mid Q)$. Our purpose is to find the *maximal a posteriori* (MAP) hypothesis that relates to the most probable solution segment in T:

$$h^* = \arg\max_{h\in\mathbf{H}} P(h \mid Q). \quad (2)$$

Since both h and Q contain n frames, the equation can be rewritten as:

$$\begin{aligned} h^* &= \arg\max_{h\in\mathbf{H}} P(h_1, \ldots h_n \mid Q_1, \ldots Q_n) \\ &= \arg\max_{h\in\mathbf{H}} P(Q_1, \ldots Q_n \mid h_1, \ldots h_n)\, P(h_1, \ldots h_n), \end{aligned} \quad (3)$$

where $h_i = T_{\theta(i)}$ is the i-th frame of the hypothesis in T. To simplify the computation of (3), we approximate it by assuming that:

(i) $P(Q_i \mid h_i)$, $i = 1 \ldots n$ are independent of each other, and
(ii) $P(h_1, \ldots h_n)$ can be modeled by a first-order Markov chain, i.e., $P(h_i \mid h_{i-1}, h_{i-2}, \ldots h_1) = P(h_i \mid h_{i-1})$ for all $i = 2 \ldots n$.

Although a video sequence actually contains higher-order redundancies, the first-order Markov chain assumption helps us make efficient use of temporal information for matching so that (3) can be rewritten as:

$$h^* = \arg\max_{h\in\mathbf{H}} P(h_1)P(Q_1 \mid h_1) \times \prod_{i=1\ldots n-1} P(h_{i+1} \mid h_i)P(Q_{i+1} \mid h_{i+1}) \quad (4)$$

which is a form of Hidden Markov Models (HMM) [12]. To evaluate (4), we consider the two probabilities $P(Q_i \mid h_i)$ and $P(h_{i+1} \mid h_i)$, where $P(Q_i \mid h_i)$ is a likelihood specifying the probability that the query frame Q_i is a copy of the hypothesis frame h_i, and $P(h_{i+1} \mid h_i)$ is the transition probability between frames h_i and h_{i+1}. We refer to $P(Q_i \mid h_i)$ as the *Probability caused by Frame Similarity* (PFS), and $P(h_{i+1} \mid h_i)$ as the *Probability caused by Temporal Continuity* (PTC). By specifying PFS and PTC, we can derive a probabilistic framework for video copy detection.

III. Probability Framework and Matching Graph

A. Probability Setting for PFS

To estimate the PFS, we use the ordinal signature and extract it as follows. A video frame is partitioned into $N_x \times N_y$ non-overlapping blocks and the average intensity of each block is computed. We then rank the blocks according to their average intensities. Consequently, the ordinal signature of the video frame is denoted as an $N_x \times N_y$ matrix of the ranking order. In this study, we set $N_x = N_y = 3$.

Let $dist(U, V)$ be the L_1 distance between any two frames U and V:

$$dist(U, V) = \sum_{x=1}^{3}\sum_{y=1}^{3} |\, F[U](x, y) - F[V](x, y)\,|, \quad (5)$$

where $F[U]$ is the ordinal signature of U, and $F[U](x, y)$ is the ordinal rank of the (x, y)-th block in U. The PFS of the query frame Q_i and the hypothesis frame h_i is then modeled as a Gaussian distribution:

$$P(Q_i \mid h_i) = \frac{1}{\sqrt{2\pi\sigma_f^{\,2}}} e^{-dist^2(Q_i, h_i)/2\sigma_f^{\,2}}, \quad (6)$$

where σ_f is the standard deviation, which remains the same for all i. Since σ_f will be eliminated in the matching cost derived in Section III-D, its estimation can be ignored in the implementation.

B. Probability Setting for PTC

To determine the probability caused by temporal continuity, PTC, we consider possible ways that the target video clip T could be altered in the temporal domain to produce near-duplicates. Without loss of generality, we consider the cases of slow and fast motions as temporal variations. Note that the slowest and fastest possible motions can be specified by the smallest and largest slopes of the mapping θ (defined in (1)), respectively. We introduce two non-negative parameters, σ_s and σ_l, to represent, respectively, the smallest and largest slopes allowed. Considering temporal modifications (e.g., slow motion and fast motion), the transition model should satisfy the following constraint:

$$\lfloor\sigma_s\rfloor \le \theta(i+1) - \theta(i) \le \lceil\sigma_l\rceil, \quad (7)$$

where $\lfloor\cdot\rfloor$ and $\lceil\cdot\rceil$ denote the floor and ceiling operations, respectively. Figure 1(a) shows a transition model comprised of three possible transitions from h_i (i.e., $T_{\theta(i)}$) to h_{i+1} (i.e., $T_{\theta(i+1)}$) when $\sigma_s = 0.5$ and $\sigma_l = 2$. We call h_{i+1} a *legal successor* of h_i if $\theta(i+1)$ and $\theta(i)$ satisfy (7).

Assume that there is no additional prior knowledge about the video speed; in other words, all video speeds within the slope bounds specified by σ_s and σ_l could occur with an equal probability. The PTC $P(h_{i+1} \mid h_i)$ can then be defined as a uniform distribution as follows:

$$P(h_{i+1} \mid h_i) = \begin{cases} \dfrac{1}{K} & \lfloor\sigma_s\rfloor \le \theta(i+1) - \theta(i) \le \lceil\sigma_l\rceil \\ 0 & \textit{otherwise} \end{cases}, \quad (8)$$

where $K = \lceil\sigma_l\rceil - \lfloor\sigma_s\rfloor + 1$ is the number of legal successors of h_i.

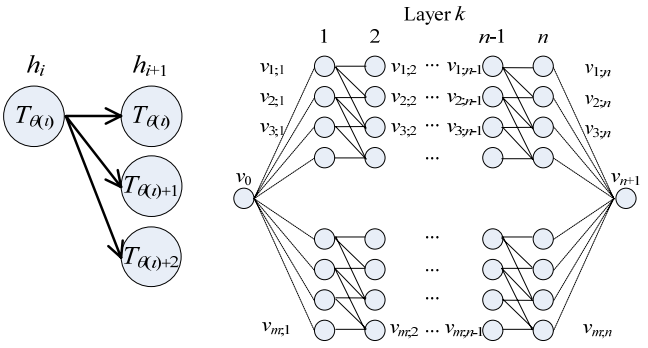


Figure 1. (a) A example of the transition model comprised of three transitions from h_i to h_{i+1}; **(b)** an example of a matching graph G.

C. Objective Function

To evaluate (4), we assume equal prior probabilities for all the hypotheses in **H**; i.e., $P(h_1) = 1/N_{total}$ for all $h_1 \in \mathbf{H}$, where N_{total} is the number of hypotheses contained in **H**. Taking a natural log of (4) yields the following formulation:

$$h^* = \arg\max_{h\in\mathbf{H}} P(h_1)P(Q_1 \mid h_1) \times \prod_{i=1\ldots n-1} P(h_{i+1} \mid h_i)P(Q_{i+1} \mid h_{i+1})$$
$$= \arg\min_{h\in\mathbf{H}} J(\mathbf{H}, Q), \quad (9)$$

where $J(\mathbf{H}, Q; \sigma_s, \sigma_l) =$

$$\begin{cases} \infty & \text{if } P(h_{i+1} \mid h_i) = 0 \text{ for some } i \in \{1 \ldots n-1\} \\ \dfrac{1}{2\sigma_f^{\,2}} \sum_{i=1}^{n} dist^2(Q_i, h_i) + \sum_{i=1}^{n-1} \ln(K) + n\ln(\sqrt{2\pi\sigma_f^{\,2}}) + \ln(N_{total}) & \text{else} \end{cases} \quad (10)$$

In (9), finding h^* that maximizes the a posteriori probability is equivalent to finding h^* that minimizes the objective function J defined in (10). Using a brute-force method to compute h^* is computationally intractable. In our work, we encode the PFS and PTC in a matching graph, and use a shortest-path algorithm to solve the minimization problem.

D. Matching Graph Construction

We extend the transition model in Figure 1(a) to construct a matching graph G, as shown in Figure 1(b). G contains n layers, each of which contains m vertices (recall that n and m present the numbers of frames in the query and target videos, respectively). For the k-th layer, the m vertices are denoted as $v_{j;k}$, $j \in \{1 \ldots m\}$, where each vertex represents a frame in the target video. We also construct a source vertex, v_0, and a sink vertex, v_{n+1}. Then, some edges are constructed by connecting nodes in G as follows. The source vertex v_0 and sink vertex v_{n+1} are fully connected to vertices in layer 1 and layer n, respectively. For each adjacent pair of layers k and k+1, where $k = 1 \ldots n-1$, if $j \le j'$ and $\lfloor \sigma_s \rfloor \le j' - j \le \lceil \sigma_l \rceil$, there is an edge $e(v_{j;k}, v_{j';k+1})$ linking the vertex representing the j-th frame $v_{j;k}$ in the k-th layer to that representing the j'-th frame $v_{j';k+1}$ in the (k+1)-th layer. For each vertex $v_{j;k}$ (except the source and sink vertices), a vertex score is assigned based on (10) as $s_{j;k} = dist^2(Q_k, T_j)$, and the edge scores are all set to $2\sigma_f^2\ln(K)$. Note that the terms $n\ln(\sqrt{2\pi\sigma_f^{\,2}})$ and $\ln(N_{total})$ in (10) are omitted from the above score setting because they are constants. In addition, the scores of the source and sink vertices, as well as the edges linking to those vertices, are all set to zero.

Each path starting from the source vertex and ending at the sink vertex represents a matching sequence between the query and target videos. The cost of the path is the sum of the vertex scores along the path. Our goal is to find the minimal-cost (or shortest) path from v_0 to v_{n+1} in the matching graph. The frames associated with the matching path then constitute the sub-segment in T that is the most similar to the query video Q, based on the objective function $J(\mathbf{H}, Q)$ defined in (10). Since there are K edges connected to the vertex $v_{j;k}$, and all of them have equivalent edge scores $2\sigma_f^2\ln(K)$, the shortest path to $v_{j;k}$ is determined by aggregating the vertex scores only:

$$j^* = \arg\min_{j'} \{cost(v_{j';k-1}) \mid j' = j, j-1 \ldots j-K+2, j-K+1\}, \quad (11)$$
$$cost(v_{j;k}) = cost(v_{j^*;k-1}) + s_{j;k}, \quad (12)$$

where $cost(v_{j;k})$ is the cost of the shortest path from v_0 to $v_{j;k}$. Both the dynamic programming approach (or equivalently, the Viterbi algorithm [12]) and the single-source shortest-path algorithm in a directed acyclic graph [1] can be used to find the shortest path from v_0 to v_{n+1}. To solve the problem, we employ the dynamic programming approach based on (11) and (12). If the total cost of the shortest path is less than τn, we determine that **H** is a copy of Q, where τ is the parameter that can be used to control the recall-precision rates.

The detection algorithm has two parts: one part calculates the vertex scores, and the other finds the shortest path. In the first part, there are mn vertices in the matching graph; Thus, its time complexity is O(mn). In the second part, the dynamic programming takes O(mnK), where K is the branching factor (i.e., the number of legal successors) of each vertex.

Since the edge scores are all equal and not involved in the path-cost computation in (11) and (12), the time required to find the shortest path can be further reduced as follows. According to (11), when $cost(v_{j;k})$ is evaluated, we need to find the minimal cost in the set $\{cost(v_{j';k-1}) \mid j' = j, j-1 \ldots j-K+2, j-K+1\}$. Then, to compute the cost of the next vertex, $cost(v_{j+1;k})$, we need to find the minimal cost in another set $\{cost(v_{j';k-1}) \mid j' = j+1, j, j-1 \ldots j-K+2\}$. These two sets have K–1 overlapping elements $\{j, j-1 \ldots j-K+2\}$. Hence, if $j^* \in \{j, j-1 \ldots j-K+2\}$, we only need to compare $cost(v_{j+1;k-1})$ and $cost(v_{j^*;k-1})$ to get the required minimal element for finding the cost of the next vertex, $cost(v_{j+1;k})$. In most cases, this requires only one comparison of each vertex (unless the special case $j^* = j-K+1$). Thus, the time complexity can be further reduced to O(mn) if the special case does not happen frequently. Although it still takes O(mnK) for the worst situation (where the case $j^* = j-K+1$ always occurs for every vertex), the average complexity for the common cases can be reduced to O(mn).

IV. Key Frame and Candidate Segment Selection

We have described a probabilistic framework for video copy detection with an average complexity of O(*mn*). Although this is the same as the complexity of the fixed-length window approach, our detection speed is slower since more computational overheads are included. In this section, we introduce two pre-processing steps, *key frame selection* and *candidate segment selection*, to further improve the matching efficiency.

A. Key Frame Selection

Inspired by the Scaled Invariant Feature Transform (SIFT) [10], we use the scale-space to identify stable key frames under different temporal scales. For a target video *T*, let *T*(*x*, *y*, *t*) be the ordinal signature of the (*x*, *y*)-th block of the *t*-th frame in *T*, and $G_{\sigma_e}(x, y, t)$ be a 3×3×3 Gaussian kernel with standard deviation σ_e:

$$G_{\sigma_e}(x, y, t) = \frac{1}{\sqrt{2\pi\sigma_e^2}} e^{-(x^2+y^2+t^2)/2\sigma_e^2}. \quad (13)$$

A 3×3×3 Difference-of-Gaussian (DoG) kernel is derived by computing the difference between two Gaussian kernels:

$$DoG_s(x, y, t) = G_{k^{s+1}\cdot\sigma_e}(x, y, t) - G_{k^s\cdot\sigma_e}(x, y, t), \quad (14)$$

where $k > 1$ is a multiplicative factor, and $s = 1, 2 \ldots$ is the scale of the DoG kernel. Then, we use the DoG kernel sliding over *T* to generate a vector L_s by the convolution operation:

$$L_s(t) = \sum_{t'=t-1}^{t+1} \sum_{x=1}^{3} \sum_{y=1}^{3} T(x, y, t') \cdot DoG_s(x, y, t'), \quad (15)$$

for $t = 1 \ldots m$. If the *t*-th element in L_s is a local extreme, it is considered the key frame in *T*. In this study, we follow Lowe's suggestion [10] and set $\sigma_e = 1.8$, $k = \sqrt{2}$, and $s = \{1, 2, 3\}$.

We conducted an experiment to validate the reliability of the key-frame selection method. The target video was transformed by spatial and temporal modifications (e.g., brightness enhancement and slow/fast motion) to obtain several test videos. We then applied the proposed method to the target and test videos to observe the *repeatability*, i.e., the percentage of the same key frames selected in both target and test videos. On average, the repeatability of the spatial and temporal modifications was 90.93% and 83.51%, respectively. For comparison, we also implemented the triangle-model method proposed by Liu *et al*. [9]. Its repeatability for the spatial and temporal modifications was 82.30% and 76.89%, respectively.

B. Candidate Segment Selection

The number of frames in the target video *T* is usually much larger than that in the query video *Q*. Since directly matching *T* and *Q* through the proposed matching-graph algorithm is time-consuming, we try to find a set of candidate segments in *T*, and then match these candidate segments with *Q*. The candidate selection method filters out many unnecessary frames; hence, the computation time is reduced. Besides, *T* may contain multiple copies, which can be located by the candidate segment selection method and verified by the matching-graph algorithm. Thus, our method can detect multiple copies contained in *T*.

As a partial matching problem, video copies can also be detected by considering pair-wise frame matching errors. We can thus use some frames in the query video for probing, and select candidate video segments from the target video by finding frames that are similar to the probing frames. Since only a few probing frames are used, the candidate selection method is efficient. Without loss of generality, we use two snippets in the query video *Q* for probing. One is selected from the head part of *Q*, $Q_{head} = \{Q_i \mid i = 1, 2 \ldots w\}$, and the other is selected from the tail part of *Q*, $Q_{tail} = \{Q_i \mid i = n–w+1, n–w+2 \ldots n\}$, where *w* is the snippet length, i.e., the number of frames in the snippet. We compute the Average Ordinal Feature (AOF) of a snippet as follows:

$$H_{Q_{head}} = \frac{1}{w}\sum_{i=1}^{w} F[Q_i] \text{ and } H_{Q_{tail}} = \frac{1}{w}\sum_{i=n-w+1}^{n} F[Q_i]. \quad (16)$$

The AOF of the *j*-th frame in the target video *T*, denoted as H_{T_j}, is computed similarly:

$$H_{T_j} = \frac{1}{w}\sum_{i=0}^{w-1} F[T_{j+i}], \; j = 1,2 \ldots m - w + 1. \quad (17)$$

The AOF serves as the signature of a snippet. A head candidate C_{head} is a set of frames selected from *T* based on the following similarity measure:

$$C_{head} = \{T_j \mid dist(H_{T_j}, H_{Q_{head}}) < \varepsilon, \; j = 1,2 \ldots m\}, \quad (18)$$

where $dist(\cdot)$ is defined in (5) and ε is the threshold. In our empirical test procedure, we set $\varepsilon = 9$ to achieve a balance between robustness and efficiency. Likewise, the tail candidate C_{tail} is selected from *T* by computing H_{T_j} and $H_{Q_{tail}}$.

Next, we scan the two candidates C_{head} and C_{tail}. An *m'*-length consecutive-frame segment $CS = \{T_s, T_{s+1}, \ldots, T_{s+m'-1}\}$ in *T* is chosen as a candidate segment in the scanning process if the following conditions are satisfied:

$$T_s \in C_{head} \text{ and } T_{s+m'-1} \in C_{tail}, \quad (19)$$

$$\sigma_s \times n < m' < \sigma_l \times n, \quad (20)$$

$$dist(H_{CS}, H_Q) < \varepsilon, \quad (21)$$

where H_{CS} and H_Q are the AOFs of the candidate segment *CS* and query *Q*, respectively:

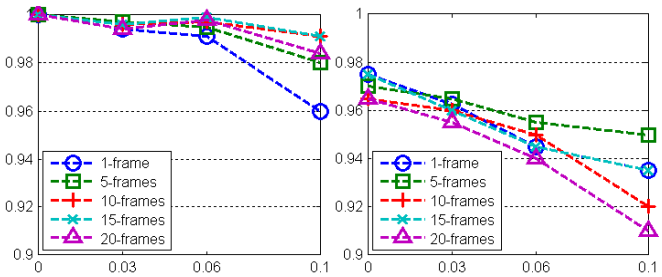
$$H_{CS} = \frac{1}{m'}\sum_{i=1}^{m'} F[T_{s+i-1}] \text{ and } H_Q = \frac{1}{n}\sum_{i=1}^{n} F[Q_i]. \quad (22)$$

The purpose of (20) is to delete candidate segments that are too long or too short, and that of (21) is to remove segments whose AOFs are not close to that of the query video.

We designed a simulation to investigate the performance of the proposed method under frame corruption and different video speeds. A collection of 712060 video frames, each comprised of 320×240 pixels, was used. We randomly selected 1000 queries from the collection. The query video was further perturbed by randomly selecting $CR \times n$ frames as corrupt frames, where $CR \in [0, 1]$ was the ratio of corrupt frames and *n* was the length of the query video. The ordinal signature of a corrupt frame was set to zero, or the frame was removed from the query video. We then submitted each query to the video

collection to obtain a set of candidate segments. In the set, if there was a candidate segment whose region overlaps with the region where the query was extracted from, we said the query achieved a *correct selection*. The correct selection ratio was defined as the number of correct selections divided by the number of queries.

Figures 2 shows the corrupt ratio (X-axis) versus the correct selection ratio (Y-axis) under different snippet lengths (w = 1, 5, 10, 15 and 20). The corrupt ratios used in the test ranged from 0% to 10%. Figure 2(a) shows the result of submitting the query videos to a normal-speed video collection, while Figure 2(b) shows the result of submitting them to a video collection consisting of 0.5× or 2× speeds. The correct selection ratios of the normal-speed case (Figure 2(a)) are higher than those of the variable-speed case (Figure 2(b)). This is reasonable since the latter includes both corrupt frames and temporal variations. However, the correct-selection ratio remains high (over 93% in average), even when 10% of the frames are corrupt. In our implementation, we selected the snippet length w = 10 to achieve a balance between the information contained in the snippet and temporal variations it can tolerate.



(a) **(b)**

Figure 2. The corrupt frame ratio (X-axis) versus the correct selection ratio (Y-axis) under different snippet lengths: **(a)** the normal speed case; **(b)** the variable speed case.

V. Experiment Results

We collected more than six hours (712060 frames) of video data from the Open Video Project and the MPEG-7 collection as our original dataset. The video content included sports, news, documentaries, and landscapes. The format of the original video data was MPEG-1 with 320×240 pixels and 30 fps. Eight copies were produced by applying spatial and temporal modifications to the original data. These modifications included brightness enhancement, histogram equalization, 176×120 pixels, 10 fps, 15 fps, 0.5× speed, 2× speed, and a hybrid case comprised of 176×120 pixels, 10 fps, and 2× speed. They were designated as the target videos. Then, 200 video clips were selected randomly from the original data and designated as the query videos, each of which contained 1000 frames. All query clips were compared with eight target videos; a total of 1600 (i.e., 200 × 8) query videos were compared.

We set σ_s = 0.5 and σ_l = 2 to detect video speeds between 0.5× and 2× speed. The threshold ε in (18) and (21) was set to 9. For key frame selection, the scale-space parameters, discussed in Section IV-A, were set to σ_e = 1.8, $k = \sqrt{2}$, and s = {1, 2, 3}. With this configuration, a key frame was extracted from the video database about every 6.43 frames on average.

A. Retrieval Accuracy

We compare the results of our approach with those of Hua et al. [5], and Kim and Vasudev [8]. Hua et al.'s approach uniformly samples a video and generates a 3×3 ordinal signature for each sampled frame, and one key frame is uniformly extracted every 6 frames. In Kim and Vasudev's framework, a 2×2 ordinal signature and a 2×2 temporal signature are used, and the two signatures are linearly combined with an equal weight. Both approaches employ a fixed-length sliding window to search for video copies.

To evaluate the performance, we often use Precision-Recall (PR) curves and Receiver Operator Characteristic (ROC) curves. Let True Positives (TP) be positive examples correctly labeled as positives, False Positives (FP) be negative examples incorrectly labeled as positive, True Negatives (TN) be negative examples correctly labeled as negative, and False Negatives (FN) be positive examples incorrectly labeled as negative. Then three metrics are given as follows:

$$\begin{aligned} &\textit{recall} = \mathrm{TP} / (\mathrm{TP} + \mathrm{FN}) \\ &\textit{precision} = \mathrm{TP} / (\mathrm{TP} + \mathrm{FP}) \\ &\textit{false positive} = \mathrm{FP} / (\mathrm{FP} + \mathrm{TN}). \end{aligned} \tag{23}$$

The PR-curve plots the recall versus the precision, while the ROC curve plots the false positive versus the recall. In this experiment, we use PR curves for evaluations because, in a large skew class distribution, a PR curve is more capable of capturing changes in the number of negative examples than an ROC curve [2]. In the following, we set different τ (defined in Section III-D) to generate PR-curves, where the X-axis denotes the recall and the Y-axis denotes the precision.

Figure 3(a) shows the results of three spatial variations: brightness enhancement, histogram equalization, and 176×120 pixels. The three approaches yield good results under these spatial modifications. Kim and Vasudev's approach produces slightly better results than our approach or that of Hua et al. because it uses every frame, instead of only key frames, for video matching. Basically, our approach achieves higher detection accuracy than that of Hua et al. Although both approaches use the same ordinal signature and close sampling rate, it is clear that our key-frame selection method can preserve more significant frames than the uniform sampling method adopted by Hua et al.

Figure 3(b) shows the results of two temporal variations: 15 fps and 10 fps. Kim and Vasudev use a fixed-length sliding window to compare 30 fps query videos with the 15 fps or 10 fps target videos. Since the sliding window can not deal with temporal discrepancies between the query and target videos, the method does not perform as well as the other two approaches. Unlike Kim and Vasudev's approach, Hua et al. use the same frame rate to re-sample query and target clips before matching; hence, the performance is less affected by frame-rate changes. Compared with the other two approaches, our approach yields relatively accurate and stable results, and can

effectively compensate for temporal discrepancies caused by different frame rates.

Figure 3(c) shows the results of the other temporal variations: 0.5× speed and 2× speed. We find that the performances of both Hua et al.'s and Kim and Vasudev's approaches are severely degraded in these cases. Their PR curves, located in the bottom right part of Figure 3(c), represent a high recall rate and a low precision rate, respectively. In other words, there are a lot of false positives in the retrieved set. Although the motion speeds are different, the slow- and fast-motion videos were generated under identical frame rates. Hence, in Hua et al.'s approach, even though the videos are re-sampled with the same frame rate, the results are still not good enough because a fixed number of frames are used in the sliding window. In contrast, the results show that our approach can handle slow and fast speeds well.

Figure 3(d) shows the result of the hybrid case that combines three variations: 176×120 pixels, 10 fps, and 2× speed. These evaluations demonstrate that the proposed approach is robust under spatial and temporal variations, and a mixture of them.

B. *Retrieval Efficiency*

The detection speed of our approach highly depends to a large extent on the length-reduction ratio r in the candidate-selection stage, where r is the sum of the lengths of the candidate segments to the length of the target video T. The smaller the ratio r, the faster the detection speed will be. In Table 1, we compare the number of query frames, n, versus r. We observe that r is relatively stable (close to 1%) as n increases. Therefore, the computation cost does not increase too much as the number of query frames n grows. For n = 1000, the detection time of our approach is only 1.28 seconds for the 6.5-hour videos on a 2.8GHz and 1GB ram computer, while the sliding window approach [8] requires 30.15 seconds.

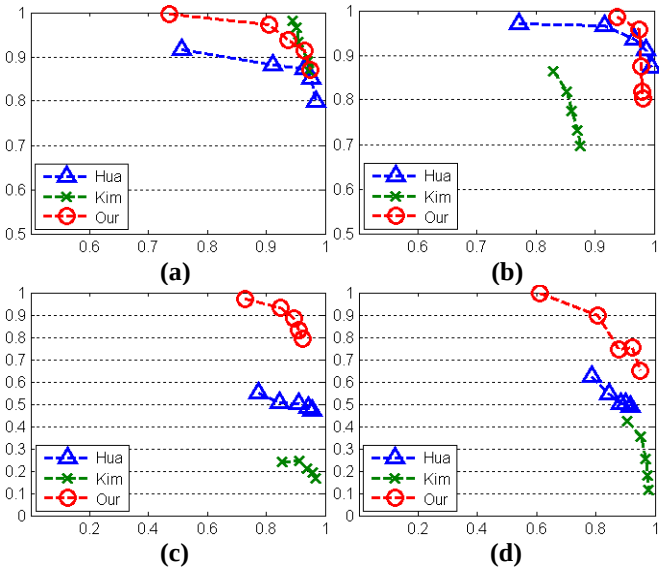


Figure 3. The PR graph for spatial and temporal variations: **(a)** brightness enhancement, histogram equalization, and 176×120 pixels; **(b)** 10 and 15 fps; **(c)** 0.5× and 2× speeds; **(d)** hybrid.

Table 1. The number of query frames n versus the length-reduction ratio r in the target video.

n	200	400	600	800	1000
r	1.02 %	0.88 %	0.89 %	1.11 %	1.07 %

VI. Conclusions

We have proposed a probabilistic framework that can handle spatiotemporal variations in video copy detection. Since approaches that use the fixed-length sliding window can not deal with general temporal variations, we treat the video copy detection problem as a partial matching problem and transform it into a shortest-path problem. To solve the problem more efficiently, we introduce the key frame and candidate segment selection, which are used to extract appropriate frames for matching. The experiment results show that our method can handle both spatial and temporal variations in video copy detection effectively at a low computational cost. In our future work, we will try to employ more powerful features as video descriptors so that image-cropping attacks can be detected.

References

[1] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*, The MIT Press, 1996.

[2] J. Davis and M. Goadrich, "The relationship between precision-recall and ROC curves," *International Conference on Machine Learning*, 2006.

[3] A. Hampapur, K.-H. Hyun, and R. M. Bolle, "Comparison of sequence matching techniques for video copy detection," *SPIE Conference on Storage and Retrieval for Media Databases*, 2002.

[4] T. C. Hoad and J. Zobel, "Detection of video sequence using compact signatures," *ACM Transactions on Information System*, Vol. 24, No. 1, pp. 1-50, 2006.

[5] X. S. Hua, X. Chen, and H. J. Zhang, "Robust video signature based on ordinal measure," *IEEE International Conference on Image Processing*, Singapore, Oct. 2004.

[6] K. Kashino, T. Kurozumi, and H. Murase, "A quick search method for audio and video signals based on histogram pruning, " *IEEE Transactions on Multimedia*, Vol. 5, No. 3, pp. 348-357, 2003.

[7] C. Kim, "Content-based image copy detection," *Signal Processing: Image Communication*, Vol. 18, No. 3, pp. 169-184, 2003.

[8] C. Kim and B. Vasudev, "Spatiotemporal sequence matching for efficient video copy detection," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 15, No. 1, pp. 127-132, 2005.

[9] T. Liu, H. J. Zhang, and F. Qi, "A novel video key-frame-extraction algorithm based on perceived motion energy model," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 13, No. 10, pp. 1006-1013, 2003.

[10] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, Vol. 60, No. 2, pp. 91-110, 2004.

[11] A. Qamra, Y. Meng, and E. Y. Chang, "Enhanced perceptual distance functions and indexing for image replica recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 27, No. 3, pp. 379-391, 2005.

[12] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, Vol. 77, No. 2, pp. 257-286, 1989.

[13] K. Yan, R. Sukthankar, and L. Huston, "Efficient near-duplicate detection and subimage retrieval," *ACM International Conference on Multimedia*, NY, USA, Oct. 2004.

[14] J. Yuan, Q. Tian, and S. Ranganath, "Fast and robust search method for short video clips from large video collection," *International Conference on Pattern Recognition*, Cambridge, UK, Aug. 2004.