Multimedia Systems (2005) 10(5): 379–391
DOI 10.1007/s00530-005-0167-6

REGULAR PAPER

**Wei-Qi Yan · Jun Wang · Mohan S. Kankanhalli**

# Automatic video logo detection and removal

Received: 3 January 2004 / Revised version: 23 April 2004 / Published online: 7 July 2005
© Springer-Verlag 2005

**Abstract** Most commercial television channels use video logos, which can be considered a form of visible watermark, as a declaration of intellectual property ownership. They are also used as a symbol of authorization to rebroadcast when original logos are used in conjunction with newer logos. An unfortunate side effect of such logos is the concomitant decrease in viewing pleasure. In this paper, we use the temporal correlation of video frames to detect and remove video logos. In the video-logo-detection part, as an initial step, the logo boundary box is first located by using a distance threshold of video frames and is further refined by employing a comparison of edge lengths. Second, our proposed Bayesian classifier framework locates fragments of logos called *logolet*s. In this framework, we systematically integrate the prior knowledge about the location of the video logos and their intrinsic local features to achieve a robust detection result. In our logo-removal part, after the logo region is marked, a matching technique is used to find the best replacement patch for the marked region within that video shot. This technique is found to be useful for small logos. Furthermore, we extend the *image inpainting* technique to videos. Unlike the use of 2D gradients in the image inpainting technique, we inpaint the logo region of video frames by using 3D gradients exploiting the temporal correlations in video. The advantage of this algorithm is that the inpainted regions are consistent with the surrounding texture and hence the result is perceptually pleasing. We present the results of our implementation and demonstrate the utility of our method for logo removal.

**Keywords** Video logo detection · Video logo removal · Video inpainting · Visual watermark attack · Neural network

W.-Q. Yan · M. S. Kankanhalli (✉)
School of Computing, National University of Singapore, Singapore
E-mail: mohan@comp.nus.edu.sg.

J. Wang
Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, The Netherlands

## 1 Introduction

Video logos have been commonly employed in the television broadcasting industry. They serve as a means of digital rights management in the sense that they are a direct declaration of video content ownership. Almost all sitcoms, news programs, and sports broadcasts have a logo in a corner of the video frames. Even when the content has been bought by one channel from another, the logos are retained and rebroadcast. Therefore, videos often have multiple overlapped logos in the same place. While it serves the broadcasters' intention of announcing the ownership, it degrades the viewing experience because of the constant obscuring of part of the content. Thus, it would be interesting to see how such logos could be removed. Removal of logos can also be construed as a form of "attack" on visible watermarks [16, 18].

A video logo can be of two distinct styles: overlapped and blended. A directly overwritten logo overlaps a portion of the content of video frames so as to completely obscure the underlying contents of the video, such as the logo located in the bottom right of Fig. 1a, while a transparently overwritten logo blends itself with the host media, for example the logo at the bottom of Fig. 1b. The blended logo allows the content of the visual media to remain partially visible.

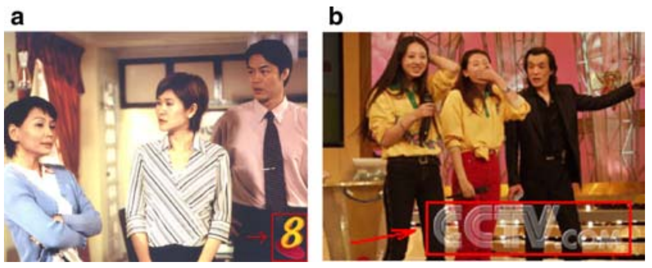


**Fig. 1** Video logos (*red rectangles*). **a** Overlapped logo. **b** Blended logo

## 2 Related work

The process of removing a video logo can be thought of as an attack on a visible watermark whose aim is to erase the embedded watermark information from its host media. Most invisible watermarking schemes can be defeated by attack software, such as Stirmark, Checkmark, and Optimark.[1] It is therefore not surprising that visible watermarks in the form of video logos are still popular worldwide and have proven to be a fairly robust yet inexpensive mechanism for digital rights protection. Unfortunately, video logos affect viewing pleasure precisely because of their robustness. Our motivation for this work was to develop a technique for removing such logos in order to restore the original viewing experience.

The easiest and most straightforward method of video logo removal is to crop out that portion of the frame. However, many video logos occupy a significant area of video frames so that cropped video frames lose a very large amount of visual information.

Image inpainting is an interesting technique that potentially could help in the removal of video logos [2, 3]. The term inpainting, an artistic synonym for image interpolation, refers to the technique of filling in gaps in photographs. The goal of inpainting is to restore the missing or damaged portions of photos. In [2], the authors borrow ideas from classical fluid dynamics to propagate isophote lines continuously from the exterior into the interior of the region to be inpainted. The technique allows users to fill in numerous regions containing completely different structures and surrounding background. This works very well for image inpainting, although it is an involved technique that is not easy to implement.

Several other works have been developed since image inpainting first came out. One is based on the extrapolation of neighboring pixels, recovery of edges, and curvature-driven diffusion [5]. A fast inpainting algorithm based on an isotropic diffusion model extended with the notion of user-provided diffusion barriers was proposed in [19]. A multiresolution image inpainting technique was provided in [27] based on inpainting at different levels, but the multiple-layer method does not appear to be very efficient. Chan et al. have investigated inpainting techniques to come up with general mathematical and variational descriptions [6, 7]. In [7] they present a theoretical analysis for a BV (bounded variation) image model that is used for image restoration; in [25] fundamental problems in image inpainting and image processing are discussed, and some theoretical contributions have been made toward solving this problem. Image inpainting has also been applied to attacking visually watermarked images in [10, 11]; image inpainting of multiple views is proposed in [3, 14], which present an algorithm for the simultaneous filling-in of texture and structure in regions of missing image information.

[1] `http://www.watermarkingworld.org`

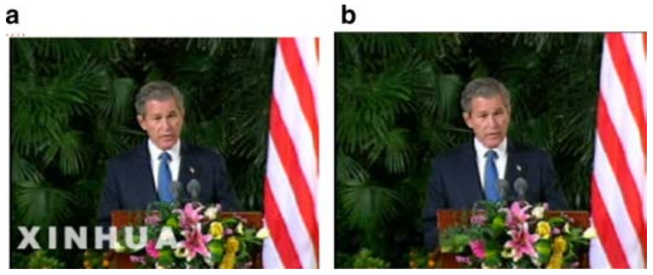


**Fig. 2** Image inpainting. **a** Image with a logo "XINHUA." **b** The inpainted image

Earlier we developed an algorithm based on image inpainting to erase video logos [31]. With this algorithm we manually select the logo region and choose the clearest logo from among all video frames. Since every frame has the logo, some have a clear appearance while others have the logo mixed with the background. We select the clearest logo in order to precisely mark the region. We then automatically erase the logo based on region filling by extrapolation. Figure 2 illustrates the removal of logos based on image inpainting. However, it is only a partial solution to the video-logo-removal problem because our inpainting technique cannot handle large gaps left due to overlapped logos. We therefore have developed a new technique of video inpainting based on texture synthesis. The proposed method works effectively on both overlapped logos and blended logos.

Bornard et al. [4] have also done relevant work on missing data correction in an image sequence, including logo removal. Although the algorithm provided in [4] has the additional benefit of avoiding the tricky step of motion vector repair and of smoothly adapting to the motion when processing video, it cannot automatically detect the logo and remove it from the motion pictures. Hence manual intervention is required.

The issue of logo detection has been considered in the domain of document understanding. In [24], a set of grayscale features is extracted to construct a suite of rules for classifying the segmentation of the logos. In [28], a method to match logos based on positive and negative shape features is proposed. In [26], contour trees and line adjacent graphs are adopted for logo detection; recursive neural networks have also been applied to classify logos. The use of grayscale features prevents these methods from exploiting the unique characteristics of video logos. In using color information, our method differs from previous methods. We extract features from both the luminance and chrominance components since our observations indicate that the distinctive features of video logos are their shapes and colors. Moreover, we use a neural network to detect the location of video logos. Our goal is to detect typical logos from video frames: overlapped logos and blended logos.

In this paper, we describe both video logo detection and its subsequent removal. For video logo detection, we present two approaches: one based on video motion and the other based on a Bayesian classifier. For video logo removal, we provide two solutions: one is to overlap a logo region by

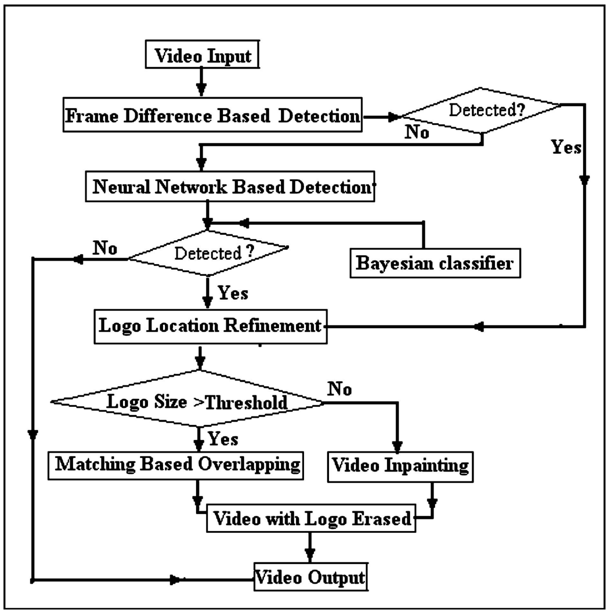


**Fig. 3** Flowchart for video logo detection and removal

seeking the best-matched frame region and the other is to remove the video logo by using inpainting.

As shown in Fig. 3, after a video clip is input, we detect a video logo by using a frame-differencing-based approach. If that fails, we use a Bayesian classifier framework in combination with a neural-network-based approach and prior knowledge to detect video logos with the aid of a logo database. After obtaining the rectangular region of the video logo, we refine it to obtain the logo outline. Once the precise logo region is obtained, we remove the logos by using either a matching-based overlapping technique or a video inpainting technique.

The reason for using two different algorithms is that distinct approaches are required based on logo background conditions. For instance, logo detection based on frame difference computation has the capability to detect logos on frames with motion, but it cannot detect them on static frames. To detect logos under realistic static conditions, we use a logo database to search for them using a Bayesian approach. To improve detection accuracy, we assume that the probability of the logo's appearing in the four corners of the video frames is higher than the probability of its appearing in the center. We combine this prior knowledge with a neural-network-based local feature classifier.

The primary reason for using two approaches to erase logos from video frames is that the result of the matching-based overlapping approach is not satisfactory if the motion of the logo region is insufficient to expose the region underneath. Another reason is that if the logo region is too large, the overlapping will result in observable edges for the video region. Hence the video inpainting approach provides an effective alternative to logo erasure.

The paper is structured as follows. We present our technique for logo detection in Sect. 3 and for removal in Sect. 4. Our results are described in Sect. 5, and conclusions and future work are presented in Sect. 6.

## 3 Video logo detection

If a video logo is to be removed, the first step is to locate the position of the video. In [31] we had to manually select the region of the video logo first and then select the best logo from among all the frames. However, this is not practical; therefore, we have developed a new method for automatically locating the video logo in a frame.

### 3.1 Initial location of the logo

A video logo, as a perceptual watermark for digital videos, has such features as distinctive shape and visible colors. It also has a stable position for a long duration within a video so that it can draw the attention of viewers and thus be identified as a trademark [9].

Suppose a video with a logo is denoted by $V = (v_{i,j,k})_{L\times W\times H}$, where $v_{i,j,k}$ is one of the color channels, $L$ is the total number of frames in the video, and $W$ and $H$ are the width and height of the video frames, respectively. We also need a Boolean array $A = (a_{m,n})_{W\times H}$, $a_{m,n} = True$ or $False$. During initialization, $a_{m,n} = True$, $m = 0, \ldots, W-1$; $n = 0, 1, \ldots, H-1$; and the color distance of corresponding pixels between two neighboring frames is given by the distance

$$d_{i,j,k} = |v_{i+1,j,k} - v_{i,j,k}|, \quad i = 0, 1, \ldots, L-1\,. \quad (1)$$

A threshold $\varepsilon$ (determined empirically) to measure the color distance is introduced to update all the Boolean values in the array: if $d_{i,j,k} > \varepsilon$, then $a_{j,k} = False$, else $a_{j,k} = True$. The advantage of this simple approach is its effectiveness, at the early stage, in filtering out a large portion of pixels that belong to the movable objects and background. It allows us to further develop a more precise logo classifier to refine the results.

### 3.2 Bayesian approach to logo detection

In this section, we propose a Bayesian classifier framework to perform the video logo detection task.

We use local features instead of global features due to the following observations on the features of the video logos:

1. *Color*: A good logo is designed such that the viewer is able to easily remember and identify it. The colors of a logo are distinctive even if the logo is mixed with a complex background. Thus primary colors, such as bright red, green, or blue, are often used. This observation suggests the use of full color features (RGB) to perform the detection task.
2. *Shape*: The shape of each logo is unique and distinctive, which obviously differentiates the logo from its surroundings. It is worth noting that while there is considerable variation in the overall shape of logos, the local

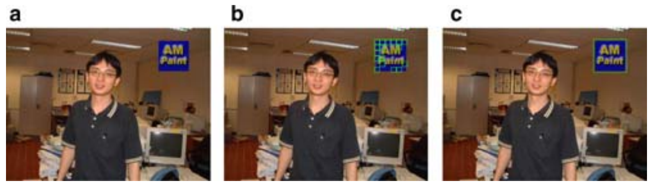


**Fig. 4** Region-based detection approach. **a** Sample frame with logo. **b** Sample frame with logo-lets detected. **c** Sample frame with logo detected

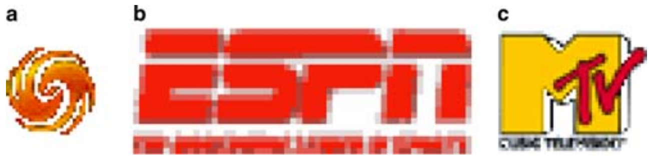


**Fig. 5** Logos with different global shapes and sizes but with the same local shapes. **a** Round. **b** Rectangular. **c** Square

shape (texture) appears to be similar. As shown in Fig. 5, although the three logos are very different (round, rectangular, and square), the local shapes are similar (with vertical, horizontal, and diagonal edges). For pattern classification, the good features of classification should have fewer variations within a class and more variations among classes. Thus the features to be selected should be related to the local shape information and not the global shape information. This suggests the use of local features that retain the local shapes while being tolerant to the variations of the global shapes. In addition, detection performed by using local features is tolerant to the variations of video logo size as well.

We therefore introduce the notion of a "logo-let." A logo-let is a fragment of a logo obtained by breaking the logo region into $12 \times 12$ pixel-sized fractions. We extract the RGB color information from the logo-lets. Since the features capture the local shape (texture information) in both luminance and chrominance components of the logos, they allow us to detect and segment the logo-lets from the video regardless of the distinct global shapes and sizes.

As shown in Fig. 6, for each pixel of the $12 \times 12$ pixel-sized fraction, RGB color values are extracted to construct the feature vector $F_R$ (a total of $12 \times 12 \times 3$ elements in one feature vector).

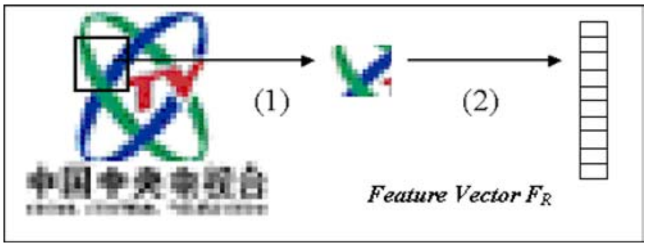


**Fig. 6** Features from the logo-lets, **1** logo-let extraction ($12 \times 12$ pixel-sized fraction), **2** feature vector $F_R$ extraction (RGB features from $12 \times 12$ pixel-sized fractions)

Again, instead of detecting the logo region, we look for the logo-lets first and then combine the found pieces to complete the entire logo detection. As shown in Fig. 4, there are two steps: first the $12 \times 12$ pixel-sized logo-lets are obtained, and then the logo is found by final combination of the detected logo-lets, as shown in Fig. 4c.

Compared to the global-features approach, there are two advantages in using local features: tolerance to both shape and size. In addition, the combination of the detected logo-lets could create a good approximation of the logo regions, as shown in Fig. 4c. This is quite important for the next step, logo removal, since it requires a good segmentation of the logo region rather than just an indication of presence/absence.

Based on this feature, we propose a novel logo detection approach. Let $F_R$ in $\Re^d$ ($d = 432$ in our case from RGB features in $12 \times 12$ pixel-sized logo-lets; each dimension has been normalized to the range [0, 1]) be a random variable denoting the local feature vector extracted from the $12 \times 12$ pixel-sized region $R$. We denote $r_R$ as the region label. It has two underlying class labels 1, 0: either $r_R = 1$ if it is a *logo-let* region or $r_R = 0$ if it is a *non-logo-let* region. Given the feature $F_R$, we want to find a classification $C$ that judges whether region $R$ belongs to a *non-logo-let region* ($r_R = 0$) or a *logo-let region* ($r_R = 1$) class based on the feature $F_R$, $C$: $F_R \to r_R$. We introduce a Bayesian decision framework to embed our prior knowledge into the above classification task.

### 3.2.1 Bayesian classifier

In contrast to other objects (e.g., faces, cars) in video scenes, logos have a clear positional preference. They normally have a higher probability of occurrence at the four corners than at the center of frames. Therefore, to use this attribute, besides using the feature $F_R$, we also add the location variable into our framework. We denote by $l_R = (x, y)$ the location of region $R$, represented by its centroid. The classification problem becomes: $C : (F_R, l_R) \to r_R$, which is to classify region $R$ into a *non-logo-let region* ($r_R = 0$) or a *logo-let region* ($r_R = 1$) by knowing both $l_R$ and $F_R$. By applying the Bayes theorem [8, 17], the classifier can be represented by the posterior probability $P(r_R = 1|F_R, l_R)$. That is, given $l_R$, the location of region $R$, and its feature $F_R$, which is the probability that region $R$ is a logo-let, we can define the classification as:

$$C : r_R = \begin{cases} 1 & P(r_R = 1 \mid F_R, l_R) > 0.5 \\ 0 & \text{otherwise} \end{cases}. \qquad (2)$$

Applying the Bayesian rule, it is easy to obtain the following equation:

$$P(r_R = 1 \mid l_R, F_R) = \frac{P(l_R \mid r_R = 1, F_R)P(r_R = 1 \mid F_R)}{P(l_R \mid F_R)}.$$

We assume that $F_R$ and $l_R$ are independent (which is normally true). The equation becomes

$$\begin{aligned}\frac{P(l_R \mid r_R = 1)P(r_R = 1 \mid F_R)}{P(l_R)} \\ = kP(r_R = 1 \mid F_R)P(l_R \mid r_R = 1)\,,\end{aligned} \tag{3}$$

where $k = 1/P(l_R)$ is a constant. Here the second factor is the probability $P(r_R = 1 \mid F_R)$, which is the posterior probability that region $R$ is a logo-let, given its feature $F_R$. The third factor, $P(l_R \mid r_R = 1)$, is the likelihood of the region regarding its location $l_R$; it reveals the positional variation of video logos in a video frame. Since it normally has a higher probability at the four corners of the frames than at the center portion, we divide each frame into four blocks and approximate the probability by a mixed Gaussian distribution. We use four bivariate Gaussian distributions to represent this probability:

$$P(l_R \mid r_R = 1) = \sum_{i=1}^{4} \frac{1}{2\pi\sqrt{\Sigma_i \mid}} e^{-\frac{1}{2}(l_R-\mu_i)(\Sigma_i)^{-1}(l_R-\mu_i)^T}, \tag{4}$$

where $\sum_i$ is the covariance matrix at corner $i$ and $\mu_i = E[l_R]$ is the expectation that the logo will be in corner $i$ ($i = 1, 2, 3, 4$ represents the four corners: top left, top right, bottom right, and bottom left, respectively). Those parameters are obtained by training from the ground truth of the logo-let location. During the training, each frame is divided into $2 \times 2$ blocks for four corners and all four of the Gaussian components $N_i(\mu_i, \Sigma_i)$, $i = 1, 2, 3, 4$ are trained according to their respective corners.

The posterior probability $P(r_R = 1 \mid F_R)$ is related to the local feature $F_R$. In our framework, it is expressed by the output of a trained neural network, which will be explained in the following section.

### *3.2.2 Posterior probability using neural networks*

Neural networks have the ability to implicitly model nonlinear relationships. Recently, it has been successfully applied to the detection problem both in the pixel domain [22, 23, 26] and the compressed domain [30]. This has motivated us to use a neural network to represent the above-mentioned posterior probability. The output of a neural network [8, 17, 21] in the range [0, 1] can be practically considered to be proportional to the classification confidence. Thus we have developed a fully connected three-layer feedforward neural-network-based classifier [29] (432 input units, 20 hidden units, and 1 output unit) trained by the feature vector $F_R$ (432 elements from a $12 \times 12$ pixel-sized region). The output of the neural network representing the posterior probability, $P(r_R = 1 \mid F_R)$, is shown below:

$$P(r_R = 1 \mid F_R) \propto O \in [0, 1], \tag{5}$$

where $O$ is the output of the neural network. We use the backpropagation algorithm to train the neural network and

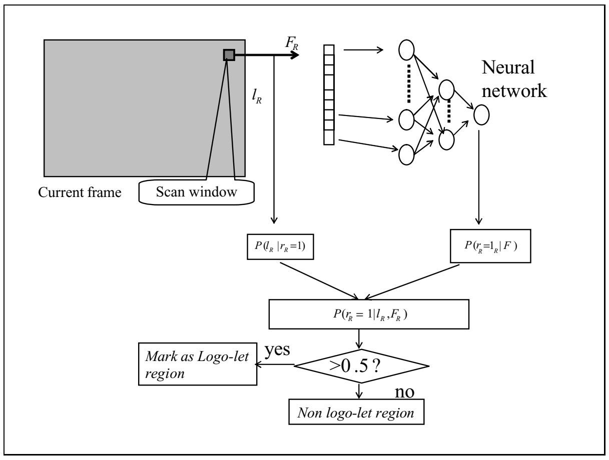


**Fig. 7** Detection routine for logo-let regions

construct a database (our data were collected from the Web), which contains 379 logos. The feature vectors of a logo-let region are extracted from the database based on the extraction scheme shown in Fig. 6. These feature vectors are used to build the logo-let region model and are stored in the form of the parameters in the trained neural network; these parameters in turn represent the distribution of the logo patterns.

For the non-logo-let features, 48 video frames picked from several movies and advertisements are employed. In total, 8,573 logo-let region features and 28,831 non-logo-let region features are used for this supervised learning.

Figure 7 illustrates the flowchart of our logo-let-region-detection approach. In Fig. 7, a rectangular window with a fixed size of $12 \times 12$ pixels is used to scan the whole area of a video frame. The feature vector $F_R$ (RGB values of $12 \times 12$ pixels, in total 432 elements) in each window is extracted. The extracted feature vector $F_R$ is input into the trained neural network, and the output is regarded as $P(r_R = 1 \mid F_R)$. On the other hand, based on the current location in frame $l_R$, the probability $P(l_R \mid r_R = 1)$ is obtained from Eq. 4. Consequently, by using these probabilities the posterior probability $P(r_R = 1 \mid l_R, F_R)$ is calculated from Eq. 3 and the current region is accordingly classified as a logo region or nonlogo region based on Eq. 2. Note that the output of the flowchart of Fig. 7 is the detection procedure of the logo-let regions. However, after the logo-let regions are detected, these regions need to be amalgamated in order to achieve final output. This postprocessing procedure has two steps: (1) merge all the neighboring log-let regions together, then (2) remove any of the isolated log-let regions that are not bigger than $12 \times 12$ pixels in size. After this postprocessing, the final results are obtained.

## 3.3 Refining the video logo location

We call the logo before it is refined a "coarse" logo. This refers to a region that may not have a clear background. Our task is to find the logo with a clear background.

After marking the position of the video logo, the region perhaps includes some details that are really not part of the logo information. A detected logo with a lot of background clutter is of low quality. We need to refine this initially detected logo by calculating the logo edge lengths.

Suppose a coarse video logo is $f(x, y), x = 0, 1, \ldots, m-1; y = 0, 1, \ldots, n-1$; the Robert edge detector is used to extract the edges in the region [29]:

$$R(f(x,y)) = \sqrt{(f(x+1,y) - f(x,y))^2} + \sqrt{(f(x,y+1) - f(x,y))^2}\,. \quad (6)$$

A threshold $E_T$ is used to decide whether a pixel is on the edge or not. Therefore, the length of the edges in the logo becomes a distinguishing feature. Given that the coarse logo is detected in all frames of a given video, we want to pick the frame with the clearest logo. We select the logo region with the shortest edge lengths as the best logo frame. This is because a bad logo is always accompanied by a complex background texture; on the other hand, a good-quality video logo displays a clear background with distinct contrast without additional texture obscuring the logo area. Relying on this observed feature, we refine the obtained logo. The logo refined by this method is very close to the real logo if the noise edges are very short. The use of this algorithm will be made clear with an example.

Figure 8a,b illustrates two logos of different quality in two frames. The logo in Fig. 8a is clear; the logo background is green grass in a soccer field and has few textures and edges. The logo in Fig. 8b has a complex background; the logo is blended with an advertisement board of "adidas" such that the logo content cannot be distinguished clearly. We cannot easily determine whether "adidas" is part of the logo or not.

## 4 Removal of the video logo

We could have considered using the image inpainting technique for logo removal. However, since this method cannot erase logos having a large area, we adopt a different approach in this paper. We remove the logo by obtaining the most similar region in a video shot to fill the region. It is akin to the idea of motion compensation used in compression albeit with the aim of performing region restoration. Although many distance measures can be selected for matching, for simplicity's sake we adopt herein the matching method based on the Hausdorff distance. The advantage of our proposal is that it is automatic without human intervention. Moreover, we develop the idea of *video inpainting*. In this paper, we imagine the logo in a frame sequence to be a "cylinder," and our job is to fill the logo "cylinder" with the help of neighboring frames and boundary information. This will be illustrated in Fig. 10.

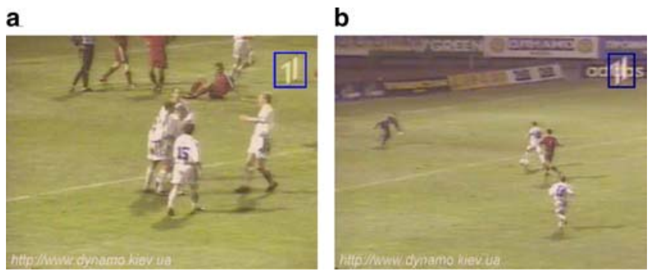


**Fig. 8** Logos of differing quality. **a** Good-quality logo. **b** Bad-quality logo

### 4.1 Matching-based logo removal

#### *4.1.1 Matching the logo region based on the Hausdorff distance*

Given two finite point sets $xA = \{a_1, a_{2,} \ldots, a_m\}$ and $B = b_1, b_{2,} \ldots, b_m\}, m > 0$ is an integer, and the Hausdorff distance is defined as: $h(A, B) = \max\{h(A, B), h(B, A)\}$, $h(A, B) = \max_{a \in A} \min_{b \in B} \|a_i - b_j\|, a_i \in A, b_j \in B, i, j = 1, \ldots, m$; $\| \bullet \|$ is a norm on the points $A$ and $B$.

The key advantage to using the Hausdorff distance to compare the difference in images is its low sensitivity to small perturbations of the image. The computation of Hausdorff distance differs from many other shape-comparison-based methods in which there is no correspondence between the model and the image. The method is quite tolerant of small position errors such as those that occur with edge detectors and other feature extraction methods [1, 12, 13, 20]. Although we do not use the pixel position information in the computation of the Hausdorff distance, each pixel has its fixed position in the blocks. Here we are mainly concerned with the position of each search block instead of the position of each pixel.

Assume that a video logo occupies a portion of video frames $\Omega_w(x, y)$; we need to find the most similar region $\Omega_s(x, y)$ among all video frames $(v_{i,j,k})_L$ under the definition of Hausdorff distance. The content of this region is most similar to the content of the region that the logo occupies $\Omega_o(x, y)$. That is:

$$d_c = h(\Omega_o, \Omega_c), \quad d_s = \min\{d_c\}, \quad \Omega_s = \Omega_c|_{d_c = d_s}. \quad (7)$$

If we can find this region, it can be used to replace the logo. But this method has a problem: since the logo overlaps the region, we cannot find the best match for it. Because we do not know what lies underneath the logo, adjacent regions are considered. The regions to the right, left, top, and bottom of the video logo region are considered for matching with other frames in the shot. For the current frame, we match these four regions with all the other video frames. We find the most similar pair among all the candidate pairs in the video frames. The corresponding position of the video logo in the best-matched frame will be found. For instance, if the most similar pair among the candidate regions is at the top of the logo region in a video, the region under the most similar

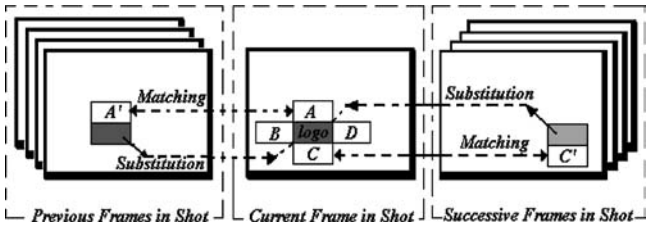


**Fig. 9** Matching-based algorithm for region overlapping

region of the matched frame will replace the logo region. Subsequently the region with the logo will be replaced, and thus the video logo will be removed.

### *4.1.2 Matching-based logo region overlapping*

When this algorithm is used to remove a video logo, at least three frames are considered: the current frame, the next frame, and the previous frame in the same shot, if available.

In Fig. 9, suppose the middle frame is a frame of the input video with a logo. Around the video logo, four regions are marked with **A**, **B**, **C**, and **D**. In order to obtain the substitution region of the logo, the regions in the related frames (previous frames and next frames) of the current frame are considered, with the most similar pair among all the candidate pairs being selected. The corresponding region is then used to replace the watermarked region. If the most similar pair is at the top of the logo, the region under matching is used to overlap the visible watermark, shown in the left-hand side of Fig. 9; if the most similar pair is at the bottom of the logo, the area above the matched region is used to overlap the logo, as shown in the right-hand side of Fig. 9.

Since objects in a video are moving, the region occupied by the video logo in the current frame may not be obscured by the logo forever. Thus we have the opportunity to locate that exposed region that can be subsequently used for removing the logo from the current frame.

The drawback of this technique is similar to that of locating the logo region. If most objects in the frames of the target video are stationary, then we perhaps could not find a better substitution region to replace the region occupied by the video logo. However, we are surely able to find some regions to replace the logo even if it is not the optimal one. Another shortcoming of the proposal is that sometimes the found blocks will bring extra visible edges after overlapping and destroy the coherence of this region. Therefore, we would like to inpaint the video frames using video inpainting. In short, overlapping-based logo removal is a supplementary measure to handle logos with motion in the background (for at least one logo region position). If the motion is not significant, video inpainting can be used to fill the logo region.

## 4.2 Video-inpainting-based logo removal

Unlike image inpainting and block-based overlapping algorithms for video logo removal, video inpainting considers a frame sequence to fill a logo region. The logo region to be filled in the frame sequence is treated as a "cylinder." Video logo removal processing can be considered as the filling of the logo "cylinder" in the video volume layer from the outermost to the innermost layers.

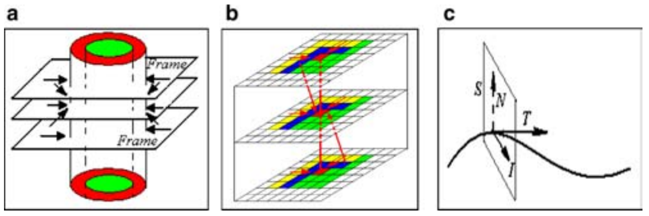


**Fig. 10** Video-inpainting-based logo removal. **a** "Cylinder"-layer-filling-based video logo removal. **b** Each pixel on the layer of the "cylinder" is inpainted using adjacent layers. **c** The vectors relevant to logo edges

Our scheme is illustrated in Fig. 10. Three example video frames are shown in the diagram. The rectangles represent pixels in frames, with the red rectangles indicating the current pixels to be inpainted. In Fig. 10b, the green rectangles depict pixels to be inpainted later, and the information of the blue pixels is used to inpaint the current pixels. The yellow pixels are also used in the rendering of the current pixels.

In Fig. 10, we overlap the logo region by using the 3D gradient vectors instead of the 2D gradient vectors normally used in image inpainting to obtain a region to replace the logo. We will compute the contributions of the gradient vectors of adjacent frames and predict the color value of the current pixels. We also use motion vectors of the logo region to find the pixels on the outside rings. The idea is to fill the marked region more precisely than was done by the methods proposed earlier.

Suppose $\Omega$ is the logo region with $\partial\Omega$ its boundary. A pixel on the boundary $\partial\Omega$ that needs to be inpainted is denoted as $z = f(t, x, y)$, where $(x, y)$ is the pixel position and $t$ is the frame in which it is located. The tangent vector is $\mathbf{T}$ $(f_t, f_x, f_y)$, and the normal vector $\mathbf{N}(n_t, n_x, n_y)$ lies on a plane, where $\mathbf{T} \cdot \mathbf{N} = 0$. The gradient vector $\mathbf{G}(-n_t, -n_x, -n_y)$ is also on this plane. To inpaint the logo "cylinder," we use only those normal vectors $\mathbf{I}(I_t, I_x, I_y)$ whose direction points toward the axis of the "cylinder," namely: $\mathbf{G} \times (1, 0, 0) \cdot \mathbf{T} > 0$, as shown in Fig. 10c. We estimate the inpainted pixels by using the first-order difference:

$$\Delta f_t = f(t+1, x, y) - f(t, x, y), \tag{8}$$

$$\Delta f_x = f(t, x+1, y) - f(t, x, y), \tag{9}$$

$$\Delta f_y = f(t, x, y+1) - f(t, x, y), \tag{10}$$

$$\begin{aligned} f(t, x, y) = {} & w_t \cdot (2 \cdot f(t-1, x, y) - f(t-2, x, y)) \\ & + w_x \cdot (2 \cdot f(t, x-1, y) - f(t, x-2, y)) \\ & + w_y \cdot (2 \cdot f(t, x, y-1) - f(t, x, y-2)), \end{aligned} \tag{11}$$

where $w_t \in [0, 1]$, $w_x \in [0, 1]$, and $w_y \in [0, 1]$ are weights that indicate the contribution of the various gradients to the current pixels, and $w_t + w_x + w_y = 1$. In the temporal domain, we exploit video frame coherence and use the correlation between these frames. Thus we need to compute the 3D gradients from different directions. Not only do we use the 2D gradients to compute the influence on the current pixels, but we also use the temporal domain information from adjacent frames. We compute these gradients to interpolate the pixels in the current frame.

The video-inpainting-based method fills the concentric "cylinder" step by step in both the temporal and spatial directions by propagating the region information. Video inpainting processing can be visualized as the dissolving surface of a cylindrical "ice block" in a tank of water. One challenge in this 3D dissolution method is that we need to obtain the motion estimation, which we use to infer the optical flows and isohyets.

## 4.3 Issues in video inpainting

In this section, we will discuss some theoretical issues. One open issue is how large a logo area can be inpainted in videos, and another open issue is how to patch the cracks that appear due to the overlapping processing to maintain coherence between adjacent video frames. Here we briefly introduce the outline of the approach.

### *4.3.1 Area of video inpainting*

Basically, a region of any size that a video logo occupies can be inpainted using our proposed method. However, the quality of the inpainted video and its correlation with the video scene will be a function of the logo size and shape. Thus, in this section we will discuss the relationship between the region size and the quality of the inpainted video.

The fidelity of video inpainting is subject to the distribution and density of marked pixels in the inpainted region. Figure 11a provides an ideal distribution, where the black blocks represent the marked pixels and the white blocks represent the original ones. In this case, for a four-connected region, there are four white pixels around each black pixel and four black pixels around each white pixel. Thus, each black pixel to be inpainted is able to infer the pixel value from its adjacent pixels. Consequently, the area to be inpainted with high fidelity can be as large as possible, that is, the inpainted area achieves the maximum quality value regardless of the size of the marked region, because there is hardly any error in the interpolation.

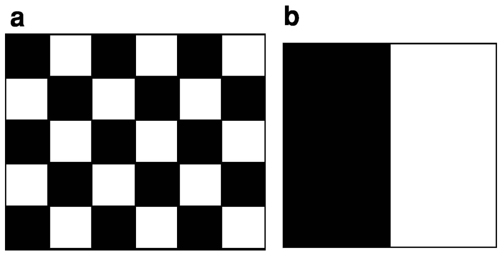


**Fig. 11** Distribution and density of marked pixels in video inpainting. **a** Best-case distribution of marked pixels. **b** Worst-case distribution

Thus the fidelity of inpainted video will be better if the marked pixels are less than in the case mentioned in Fig. 11a while the distribution remains the same. Otherwise, if the density of the marked pixels is high while retaining the distribution, the restored quality will be degraded. Figure 11b presents the worst case for video inpainting fidelity in relation to inpainting density and distribution. Certainly, if the marked region is entirely filled by the marked pixels, the fidelity of the restored region is the lowest. Errors will be propagated over a large distance in this case.

The logo is a symbol that covers a portion of video frames. Video inpainting is the approximation of the covered region by using the surrounding information and their differences. Thus the region to be filled is a "hole" or a "cylinder" that has temporal and spatial coherence. For logo removal, the contributions of each ring surrounding the logo region (four-connected region) should satisfy Eq. 11. We now estimate the contributions of each ring pixel to the boundary region:

$$\begin{aligned} f'(t,x,y) = {} & w_{t-1}\cdot(f(t-1,x,y)+\Delta f_{t-1}) \\ & + w_{x-1}\cdot(f(t,x-1,y)+\Delta f_{x-1}) \\ & + w_{y-1}\cdot(f(t,x,y-1)+\Delta f_{y-1}). \end{aligned} \tag{12}$$

The error is

$$\begin{aligned} \delta(t,x,y) = {} & |f(t,x,y)-f'(t,x,y)| = |w_{t-1} \\ & \cdot(\Delta f_t - \Delta f_{t-1}) + w_{x-1}\cdot(\Delta f_x - \Delta f_{x-1}) \\ & + w_{y-1}\cdot(\Delta f_y - \Delta f_{y-1})| = |V_1 - V_2|, \end{aligned} \tag{13}$$

where $\mathbf{V}_1 = (w_{t-1}\cdot\Delta f_t, w_{x-1}\cdot\Delta f_x, w_{y-1}\cdot\Delta f_y)^T$ and $\mathbf{V}_2 = (w_{t-1}\cdot\Delta f_{t-1}, w_{x-1}\cdot\Delta f_{x-1}, w_{y-1}\cdot\Delta f_{y-1})^T$. Equation 13 suggests that the inpainted pixels are closely related to the gradient vectors of neighboring rings. If we use the gradient of only the pixels near the edges to estimate the pixels far from the edges, this will lead to error. If a pixel to be inpainted is far from the edge, the pixel value is less accurate. Thus it is impossible to inpaint an infinite extent of a region.

Actually, the above situation can be described better by taking a continuous function $f(t,x,y)$ into account instead of the discrete expression. Then its Taylor series expansion at $(t_0+\delta_t, x_0+\delta_x, y_0+\delta_y)$, $\delta_t > 0$, $\delta_x > 0$, $\delta_y > 0$ is:

$$\begin{aligned} f(t_0+\delta_t, x_0+\delta_x, y_0+\delta_y) = {} & f(t_0,x_0,y_0) \\ & + f_x(t_0,x_0,y_0)\delta_x + f_y(t_0,x_0,y_0)\delta_y \\ & + f_t(t_0,x_0,y_0)\delta_t + o(\rho), \end{aligned} \tag{14}$$

where

$$\rho = \sqrt{\delta_t^2 + \delta_x^2 + \delta_y^2}\,. \tag{15}$$

In general, if the video frame is represented by a continuous function $f(t,x,y)$ with $n+1$ order partial derivatives in the region $\Omega = \{(t_0+\delta_t, x_0+\delta_x, y_0+\delta_y), \delta_t > 0, \delta_x > 0,$

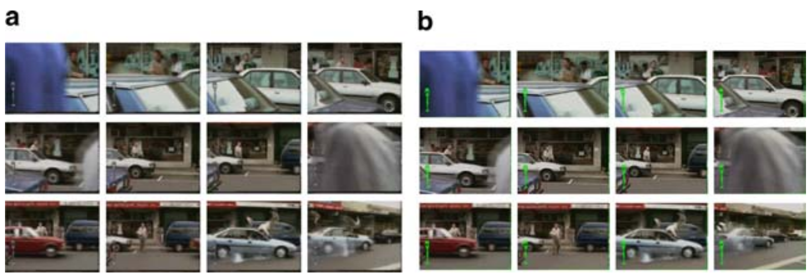
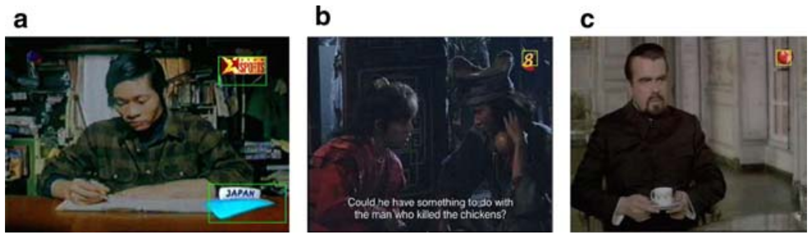


**Fig. 12** Video logo detection based on frame difference. **a** Original video. **b** The video logo is detected


Could he have something to do with the man who killed the chickens?
**Fig. 13** Detected results of overlapped logos. **a** TV logo 1. **b** TV logo 2. **c** TV logo 3

$\delta_y > 0\}$, $(t_0, x_0, y_0) \in \partial\Omega$, then the Taylor expansion of the function at $(t_0+\delta_t, x_0+\delta_x, y_0+\delta_y)$ is given by

$$\begin{aligned}f(t_0+\delta_t, x_0+\delta_x, y_0+\delta_y) &= f(t_0,x_0,y_0)\\ &+\left(\delta_t\frac{\partial}{\partial t}+\delta_x\frac{\partial}{\partial x}+\delta_y\frac{\partial}{\partial y}\right) f(t_0,x_0,y_0)\\ &+\frac{1}{2!}\left(\delta_t\frac{\partial}{\partial t}+\delta_x\frac{\partial}{\partial x}+\delta_y\frac{\partial}{\partial y}\right)^2 f(t_0,x_0,y_0)+\cdots\\ &+\frac{1}{n!}\left(\delta_t\frac{\partial}{\partial t}+\delta_x\frac{\partial}{\partial x}+\delta_y\frac{\partial}{\partial y}\right)^n f(t_0,x_0,y_0)+\varepsilon_n, \quad (16)\end{aligned}$$

where

$$\left(\delta_t\frac{\partial}{\partial t}+\delta_x\frac{\partial}{\partial x}+\delta_y\frac{\partial}{\partial y}\right)^m f(t_0,x_0,y_0) = \sum_{r=0}^{m} C_m^r \delta_t^a \delta_x^b \delta_y^c \frac{\partial^m f(t,x,y)}{\partial t^a \partial x^b \partial y^c}, \quad a+b+c=m \qquad (17)$$

and $\varepsilon_n$ is the remainder:

$$\varepsilon_n = \frac{1}{(n+1)!}\left(\delta_t\frac{\partial}{\partial t}+\delta_s\frac{\partial}{\partial x}+\delta_y\frac{\partial}{\partial x}\right)^{n+1} \times f(t_0+\theta\delta_t, x_0+\theta\delta_x, y_0+\theta\delta_y), \theta\in(0,1). \qquad (18)$$

$\varepsilon_n$ in Eq. 18 gives the estimate of the error for a continuous function in image inpainting.

### *4.3.2 Stitching of cracks in texture overlapping*

In general, a suitable block can always be found to overlap a logo region no matter how large the size of the logo region and how inferior the quality of the overlapped image. However, the region sometimes leaves a trace of overlapping and causes perceptual quality degradation. To properly remove the obvious edges arising due to overlapping, we can use the idea related to texture cuts [15]. In [15], video textures can be easily cut by an intelligent scissor; blending automatically sews the seams of two overlapping frames. Unless there are unsuitable, significant, or obvious lighting and color variations, the overlapping edges are undetectable. Thus a reasonable solution would be to find a similar block and blend the edges [15] so that the edges are not visible.

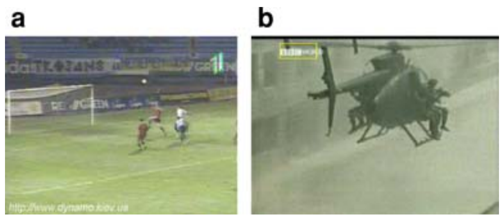


**Fig. 14** Detected logos of blended logos. **a** Blended logo 1. **b** Blended logo 2

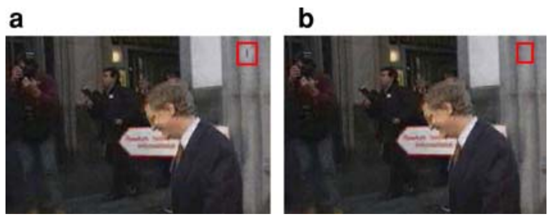


**Fig. 15** Logo removal. **a** Original frame of a video with logo. **b** Corresponding frame in video with removed logo (PSNR = 25.70 dB)

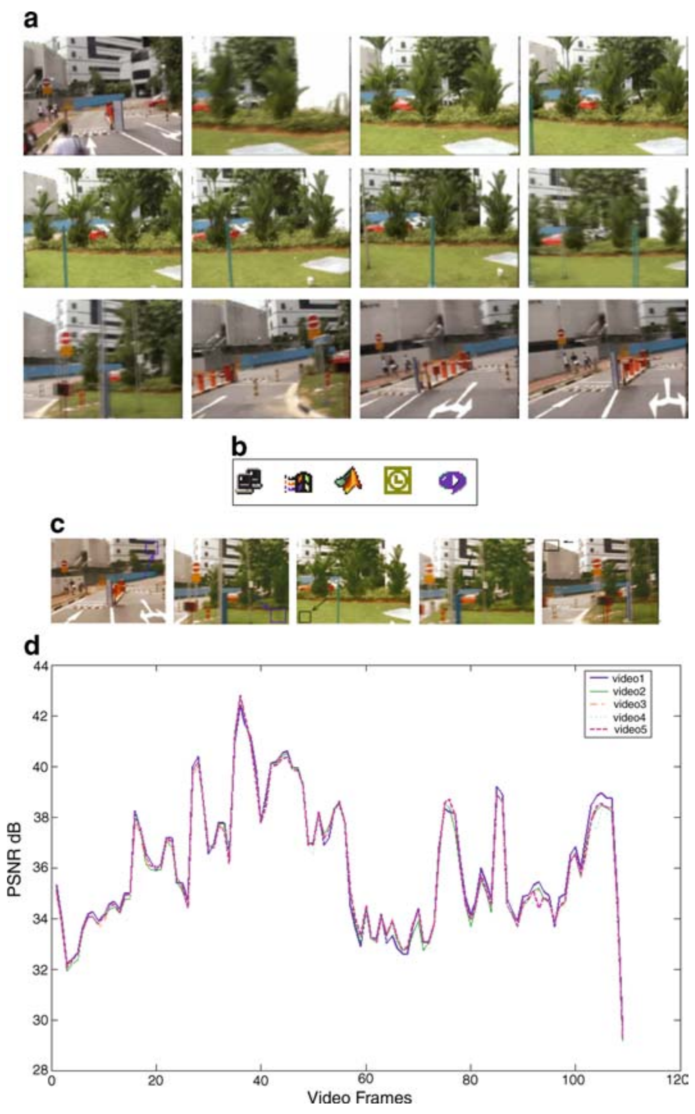


**Fig. 16** Video logo removal based on overlapping. **a** Original video without logo. **b** Five logos for experimental results. **c** Some good overlapping frames. **d** PSNR comparison in five different logo-erased videos

## 5 Results

In this section, we describe some of the experimental results. The online version of these experimental results can be found on our Web site.[2]

### 5.1 Logo detection

#### *5.1.1 Logo detection based on frame difference*

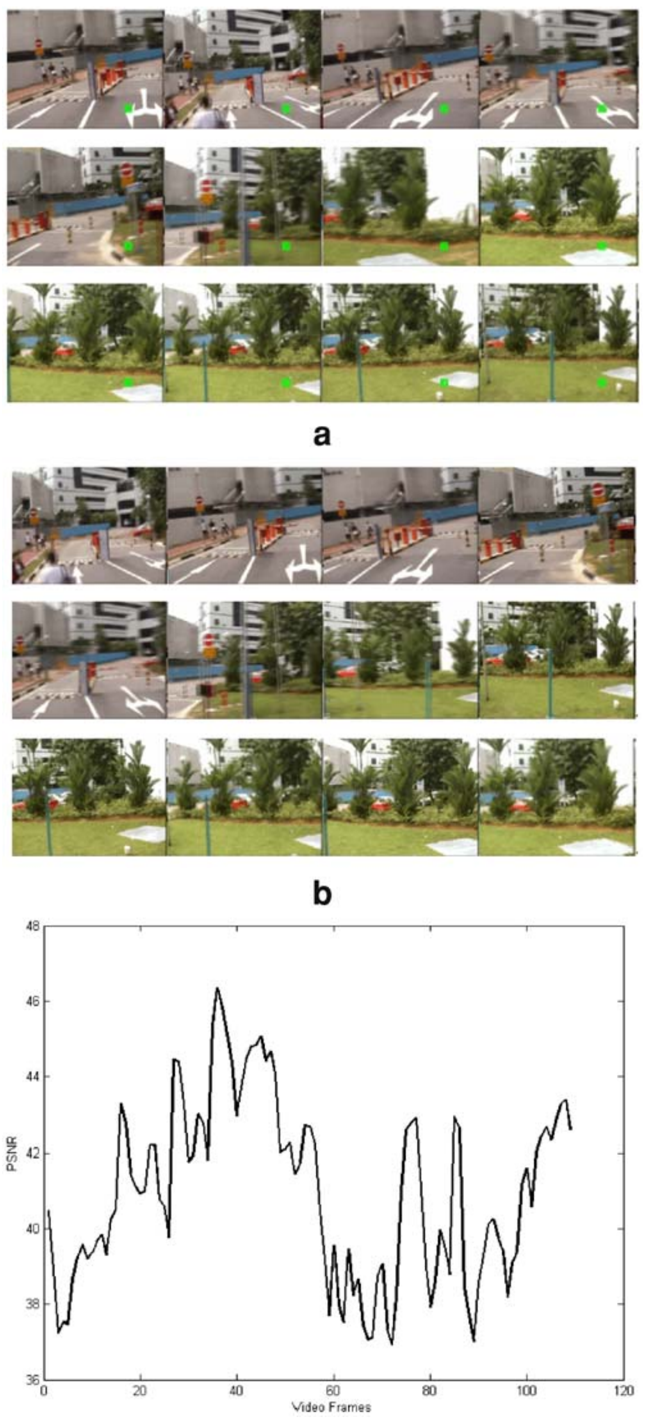


**Fig. 17** Logo removal based on video inpainting. **a** Video with marked logo. **b** Video with erased logo. **c** Comparison between original video and video with erased logo

Figure 12 lists the results of video logo detection. Figure 12a shows the original video frames with a logo, and Fig. 12b is the video frames with detected video logos. From the results we can clearly see that the video logo is successfully detected. In this experiment, the threshold for logo detection is 64. If the sum of the difference between two frames is greater than 64, we assume that motion exists.

[2] `http://www.comp.nus.edu.sg/~yanwq/ACMMMSYS/logo/index.htm`

### 5.1.2 *Logo detection based on neural network*

To demonstrate our proposal, we tested several video clips collected from various Web sites. The detection rate obtained is over 82% on 236 frames (10 nonlogo frames, 226 frames with logos extracted from 23 different video clips containing 23 different video logos) listed in Table 1.

**Table 1** Video logo detection results

| Total logos | 226 |
|---|---|
| Detected logos | 187 |
| Missed logos | 39 |
| False alarms | 36 |
| Detection rate (%) | 82.7 |

Our experiments show that our approach to video logo detection is quite robust across various types of logos with respect to shape and size. Figure 13 shows some stacked TV logos, and Fig. 14 illustrates some blended logos from various Web sites. It indicates that the local features we used are tolerant in shape and size due to the fact that they are extracted from small (12 × 12) regions.

## 5.2 Logo removal

### 5.2.1 *Video logo removal based on overlapping*

Figure 15a shows a frame of video with a logo in its top-right corner, and Fig. 15b is the same frame with the logo removed. We remove the logo by overlapping a similar region found in the video shot. The rectangle on the frames indicates the positions of the logo.

We take a video without any logo (NUS campus) shown in Fig. 16a and add five different logos at different positions of the video frames. The logos used are shown in Fig. 16b. We remove these logos using the matching-based overlapping approach and obtain five restored videos without logos. Some resultant frames are listed in Fig. 16c, the logo-removed regions are marked by rectangles, such as top-right, bottom-right, bottom-left, top-left corners and center, etc. We then compare the original video and the restored videos without logos by computing the difference between each pair and show the PSNR (peak-signal-to-noise ratio) values (in dB) obtained as results. The PSNR values are shown in Fig. 16d.

From Fig. 16 we see that the experimental results are quite good. The average PSNR values are around 35.5 dB. It is well known that if the PSNR values are greater than 30 dB, the two compared images are perceptually similar.

Matching has been used to overlap the logo region for dynamic videos. Since it does not work well for a static video, we use video inpainting for such cases.

### 5.2.2 *Video logo removal based on video inpainting*

In case the direct overlapping method does not work, we attempt to remove the logo using video inpainting. Figure 17 shows the experimental results for this case. We erase the green logo in the video shown in Fig. 17a and obtain the video shown in Fig. 17b. Figure 17c is the comparison

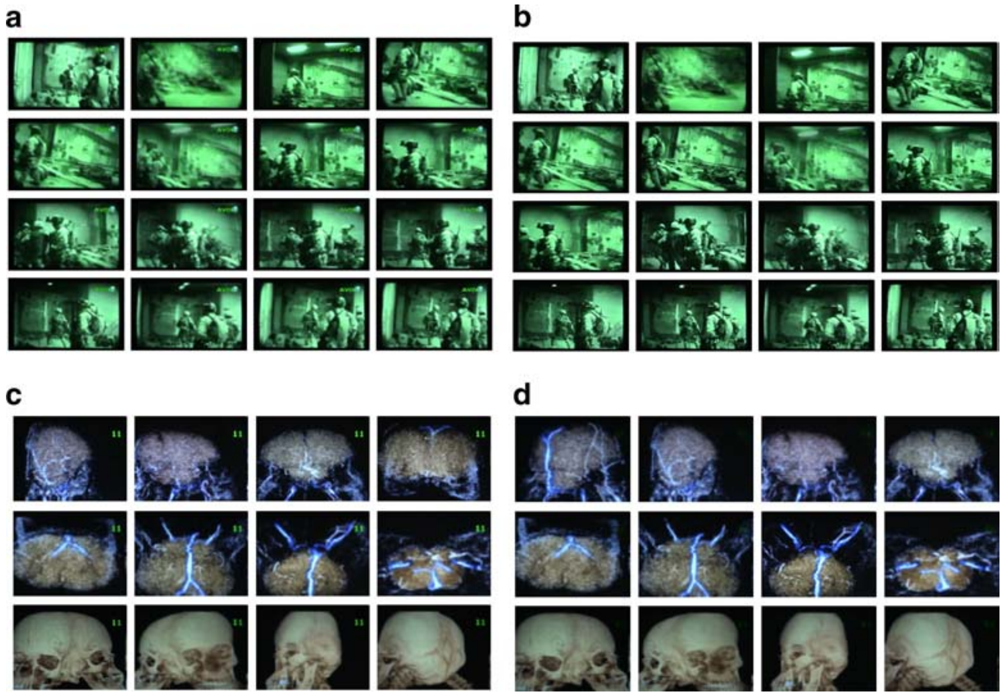


**Fig. 18** Logo removal in various video clips. **a** Iraq war video with a logo. **b** The video in **a** after removal of logo. **c** Iranian twin sisters' connected skull with logo. **d** The video in **c** after removal of logo

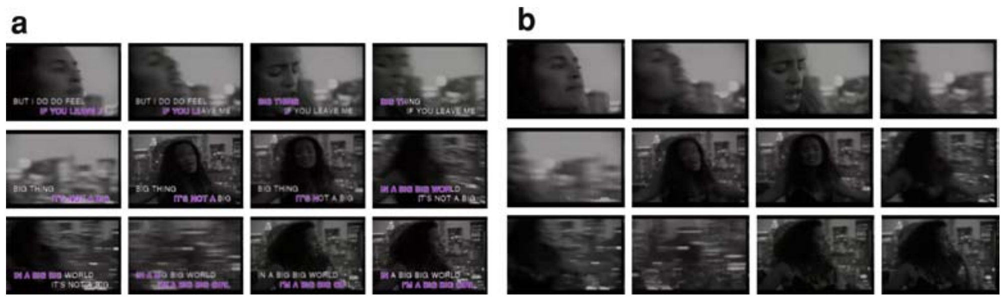


**Fig. 19** Removal of captions on a karaoke video. **a** A karaoke clip with captions. **b** The clip with removed captions

between the original video and video with erased logo. We can see the PSNR value in Fig. 17c is above 36 dB, which indicates that we can find hardly any perceptual difference between the two. Although the video is the same as in Fig. 16, the result is different since we use a different method to inpaint the video logo.

Figure 18 lists one of the results when the techniques are used in real video logo handling. The figure shows a piece of news video from the Iraq war. Some frames with the logo of U-Channel in Singapore are provided in Fig. 18a, and some frames with the logo removed are shown in Fig. 18b. Figure 18c,d shows the results of another experiment: Fig. 18c is the animation of the Iranian twin sisters' connected skull with the logo "11" on the frames, and Fig. 18d is the animation with the logo removed.

Figure 19 illustrates a result when the techniques are used to remove the captions on a karaoke video. Figure 19a provides a karaoke clip with captions, while Fig. 19b presents the same clip with removed captions.

We have shown the results of logo detection and removal as well as for real logo handling in a news video clip and a karaoke clip. From these results we can see that the proposed techniques are quite effective.

## 6 Conclusions

Video logo removal is a hard problem despite its apparent ease. We have developed a new technique based on frame difference and neural networks to locate the video logo position. Moreover, a logo-refining algorithm has been provided for obtaining a more precise logo region. If a good match cannot be obtained, the technique of video inpainting can be used. The experimental results indicate the feasibility and effectiveness of our technique. For future work, we will consider attention-based and context-based logo detection on fabric. We will also attempt online logo removal from live video streams. Another challenge would be to erase logos directly in compressed-domain MPEG videos.

**Acknowledgements** Wei-Qi Yan's work on this paper is sponsored by a fellowship from the Singapore Millennium Foundation (SMF). We deeply appreciate the constructive suggestions from the anonymous reviewers.

## References

1. Belogay, E., Cabrelli, C., Molter, U., Shonkwiler, R.: Calculating the Hausdorff distance between curves. Inf. Process. Lett. **64**, 17–22 (1997)
2. Bertalmio, M., Sapiro, G., Caselles, V., Ballester, C.: Image inpainting. In: Proceedings of ACM Siggraph, pp. 417–424 (2000)
3. Bertalmio, M., Vese, L., Sapiro, G., Osher, S.: Simultaneous texture and structure image inpainting. IEEE Trans. Image Process. **12**(8), 882–889 (2003)
4. Bornard, R., Lecan, E., Laborelli, L., Chenot, J.: Missing data correction in still images and image sequences. In: Proceedings of ACM Multimedia. Juan Les Pins, France, pp. 355–361 (2002)
5. Chan, T.F., Shen, J.H.: Non-texture inpainting by curvature-driven diffusions. J. Vis. Commun. Image Represent. **12**(4), 436–449 (2001)
6. Chan, T.F., Shen, J.H.: Mathematical models for local deterministic inpaintings. SIAM J. Appl. Math. **62**(3), 1019–1043 (2001)
7. Chan, T.F., Shen, J.H.: On the role of the BV image model in image restoration. Am. Math. Soc. Contemp. Math. **330**, 25–41 (2003)
8. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern-classification. Wiley, New York 2000
9. Holland, R.J., Hanjalic, A.: Logo recognition in video stills by string matching. In: Proceedings of the IEEE International Conference on Image Processing (ICIP 2003), Barcelona (2003)
10. Huang, C.H., Wu, J.L.: Inpainting attacks against visible watermarking schemes. Proc. SPIE **4314**, 376–384 (2001)
11. Huang, C.H., Wu, J.L.: Attacking visible watermarking schemes. IEEE Trans. Multimedia **6**(1), 16–30 (2004)
12. Huttenlocher, D., Klanderman, G., Rucklidge, W.: Comparing image using the Hausdorff distance. IEEE Trans. Pattern Anal. Mach. Intell. **15**(9), 850–863 (1998)
13. Huttenlocher, P., Lilien, H., Olsen, F.: Approximate Hausdorff matching using eigenspaces. In: Proceedings of the ARPA Image Understanding Workshop, pp. 1181–1186 (1996)
14. Kang, S.H., Chan, T.F., Soatto, S.: Landmark based inpainting from multiple views. In: Proceedings of the International Symposium on 3D Data Processing Visualization and Transmission (3DPVT) (2002)
15. Kwatra, V., Schödl, A., Essa, I., Turk, G., Bobick, A.: Graphcut textures: image and video synthesis using graph cuts. ACM Trans. Graph. **22**(3), 277–286 (2003)
16. Meng, J., Chang, S.F.: Embedding visible video watermarking in the compressed domain. In: Proceedings of the IEEE International Conference on Image Processing (ICIP 1998), Chicago, pp. 474–477 (1998)
17. Mitchell, T.M.: Machine Learning. New York: McGraw-Hill 1997
18. Mohanty, S.P., Kankanhalli, M.S., Ramakrishnan, K.R.: A DCT domain visible watermarking technique for images. In: Proceedings of the IEEE International Conference on Multimedia and Expo (ICME 2000), New York, pp. 1029–1032 (2000)

19. Oliveira, M.M., Bowen, B., McKenna, R., Chang, Y.S.: Fast digital image inpainting. In: Proceedings of the International Conference on Visualization, Imaging and Image Processing (VIIP 2001), pp. 261–266 (2001)
20. Olsen, F.: A probabilistic formulation for Hausdorff matching. In: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR 1998), pp. 150–156 (1998)
21. Papageorgiou, P., Oren, M., Poggio, T.: A general framework for object detection. In: Proceedings of the International Conference on Computer Vision (ICCV 1998), Mumbai, India, pp. 555–562 (1998)
22. Rowley, H.A., Baluja, S., Kanade, T.: Neural network-based face detection. IEEE Trans. Pattern Anal. Mach. Intell. **20**(1), 23–38 (1998)
23. Rowley, H.A., Baluja, S., Kanade, T.: Rotation invariant neural network-based face detection. In: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR 1998), pp. 38–44 (1998)
24. Seiden, S., Dillencourt, M., Irani, S.: Logo detection in document images. In: Proceedings of the International Conference on Imaging Science, Systems, and Technology, pp. 446–449 (1997)
25. Shen, J.H.: Inpainting and the fundamental problem of image processing. SIAM News **36**(5) (2003)
26. Sheng, J.: Graphical item recognition using neural networks. Ph.D Thesis, Universita degli Studi Di Firenze (1998)
27. Shih, T.K., Lu, L.C., Chang, R.C.: Multi-resolution image inpainting. In: Proceedings of the IEEE International Conference on Multimedia and Expo (ICME 2003), Baltimore, MD (2003)
28. Soffer, A., Samet, H.: Using negative shape features for logo similarity matching. In: Proceedings of the 14th International Conference on Pattern Recognition (ICPR 1998), Brisbane, Australia, pp. 571–573 (1998)
29. Sonka, M., Hlavac, V., Boyle, R.: Image processing, analysis, and machine vision, 2nd edn. London: Brooks/Cole, Chapman & Hall 1999
30. Wang, J., Kankanhalli, M.S., Mulhem, P., Abdulredha, H.H.: Face detection using DCT coefficients in MPEG video. In: Proceedings of the International Workshop on Advanced Imaging Technology (IWAIT 2002), pp. 66–70 (2002)
31. Yan, W.Q., Kankanhalli, M.S.: Erasing video logos based on image inpainting. In: Proceedings of the IEEE International Conference on Multimedia and Expo (ICME 2002), Lausanne, Switzerland pp. 521–524 (2002)