# Video Logo Removal Using Iterative Subsequent Matching

Maryam Dashti
Computer Engineering and Information Technology
Amirkabir University of Technology
Tehran, Iran
m.dashti@aut.ac.ir

Reza Safabakhsh
Faculty member of Computer Engineering and Information Technology
Amirkabir University of Technology
Tehran, Iran
safa@aut.ac.ir

Mohammadreza Pourfard
Electrical Engineering Department
Amirkabir University of Technology
Tehran, Iran
pourfardm@aut.ac.ir

Mohammadjavad Abdollahifard
Faculty member of Electrical Engineering Department
Tafresh University
Tehran, Iran
mj.abdollahi@tafreshu.ac.ir

***Abstract*— Video inpainting methods has a large number of applications and some of these algorithms are specialized for specific applications such as logo removal. There are only a few general video inpainting algorithms most of which are very time-consuming. This problem makes these algorithms unsuitable for fast video inpainting. In this paper, a fast simple logo removal algorithm has been proposed which uses frames of each video shot for logo removal and removes logo from video after a few iterations. A more accurate non-casual version of our algorithm is also proposed which uses both the information of previous and next frames. The quality of the inpainted video is also comparable with well-known video inpainting algorithms.**

***Index Terms*— Iterative, Logo Removal, Subsequent Matching, Video Inpainting, Video Texture.**

## I. Introduction

Video inpainting is a method which tries to fulfill the degraded places of a video through the healthy region of a video. Video inpainting has a widespread application [1-5] including Repairing the degraded video [3] and logo removal [6-7]. After the pioneering work of Efros et al. [8] in texture synthesis, the idea was extended and used in lots of applications such as video textures [1] and dynamic textures [9]. This idea has also been extended to arbitrarily non-textural images. Criminisi [10] has added some priority factors and mixed them with the idea of Efros et al. [8] to fill a large hole in an image. On the basis of similarity between image and video frame's, Criminisi et al.'s [10] idea has been used in videos. Kokaram et al. [3] have used this idea to fill the image hole for every frame. In fact, they have used the information of the previous and next frames to modify a specific frame of a video [11]. While this non-casual algorithm has solved some problems of video inpainting, some deficiency will remain in the video [12]. In fact, these approaches have inpainted every frame very well but they are not totally compatible in time and some abrupt changes are observable in time. This problem has been solved reasonably by Wexler et al. [12]. He has used both space and time blocks to judge about the value of each pixel of each frame. He has also defined an optimization cost function to consider both the local and temporal consistency. This idea was the last general idea for video inpainting and up to now some customized algorithms for specific applications have been introduced. For example, a human is a non-rigid object and has lots of variation in a video. Some ideas have focused on human motion to solve this problem [5, 13-15]. Video inpainting has also been used in a 3D TV [16]. Xlet approaches have also found their way in video inpainting and an approach is proposed based on Bandlet transform [17]. All of these approaches are very time consuming and a general-purpose method like Wexler [12] needs a few hours' time to remove a logo of a 60 frame video with a Quad-core 4G RAM computer.

In some applications such as logo removal from a soccer game it is highly necessary to have an online algorithm. This algorithm must be a fast non-casual one with good quality.

In this paper, a fast algorithm for logo removal is proposed. Our algorithm uses an iterative subsequent matching method for logo removal. Our algorithm has two steps. First it divide the video into some homogeneous shots. Then in every video shot it uses the information of the two subsequent frames to fill the video hole. It uses the information which is in frame $x_i$ and not in frame $x_{i+1}$ to fill the hole in frame $x_{i+1}$. It also uses the information which is in frame $x_{i+1}$ and not in frame $x_i$ to fill the hole in frame $x_i$;

The accuracy of our algorithm is increased every frame in a video shot and the logo place will be filled after a few iterations.

This paper is organized as follows. In section 2, our video shots extraction will be explained. Then the algorithm of our subsequent matching method will be clarified. In section 3, the result of our algorithm on a crowded soccer game will be shown.

## II. Proposed Method

### A. Video Shots

The general-purpose video inpainting algorithms use the information of the next frames to have a better judgment about a pixel value in an image so they are non-casual and cannot be used for online application. They also have a high computational cost. In this paper, an idea based on video shots is introduced. Video shot is a sequence of the frames captured from a specific scene. For example, assuming in a soccer game, camera is showing the crowd (Fig. 1(a)), it is a frame of a video shot and when it changes its view and shows the referee (Fig. 1(b)) it is a frame of another video shot. Frames of every video shot have structural and statistical similarity and often show the same type of textures.

Video shots can be defined through total average of difference of each two subsequent frames. If the value of this total average is sufficiently larger than a specified threshold, the second frame is detected as the first frame of a new shot.

(a) (b)

Figure 1. Frame (a) and (b) are the subsequent frames. These are from two video shots of a soccer game

*B. Subsequent Matching*

To fill a logo in a frame, a preliminary algorithm is to consider a rectangle around the logo and then find a best match in the current and multiple previous frames with respect to the pixel values in the rectangle around the logo. But this global judgment may be inaccurate and some deficiency will be seen in the filled logo place.

In our algorithm we will fill the logo place very smoothly and then we will update it in every subsequent frame. To this goal, consider two subsequent frames, frame no.1 (Fig. 2(a)) and frame no. 2 (Fig. 2(b)). To fill the logo place in frame no.1, we consider a bigger rectangle around the logo in frame no. and one around logo in frame no.2. Each rectangle has some valid pixels and some invalid pixels which are placed in the logo place.

We want to find the best value for invalid pixels with the help of valid pixels around logo, so we move the rectangle in frame no.2 some pixels up, down, right and left around its place to find the best match for rectangle in frame no.1. The best match has the least mean square error (LMSE) with rectangle in frame no.1 for valid pixels (Fig. 2(c)). So only the valid pixels can fill the logo place. Therefore, if the best match is found in a place outside the boundary of the logo place (it is possible for the last frames of each shot), all the pixels of the logo place will be filled.

Fig 2.e and Fig 2.f shows frame no.2 and no.15 respectively. But to describe our algorithm more visually suppose that they are two subsequent frames. The best match in Fig 2.d has found in red rectangle of Fig 2.f. This match can fill degraded part of the neck of referee of Fig 2.e.

To clarify the proposed method, consider Fig.3. In this figure pixels with value 1 are valid pixels and other pixels are invalid pixels. Fig.3 (a) is the first frame 1 and Fig.3 (b) is the second frame with the best match. After finding the best match, invalid pixels in frame 1 will be replaced with valid pixels in the best match and vice versa. Fig.3(c) shows the result of this replacement for frame no.1 and Fig.3 (d) shows the result for frame no.2. Then filled invalid pixels of the logo, are presented with (*). For the next frame, the third frame, new frame no.2 will be used for filling the logo place. Therefore, the logo place in all the frames of a shot will be filled.

The idea presented in this paper, focus on the point that the information of the best match for a logo in the second frame can have some useful information for the first frame and vice versa. Thus, this information should not be discarded and should be used.

Consider Fig. 3, blue pixels show the place where the second match has a value, while the first match does not have a value and vice versa. If we copy the information of this place from the best match in frame no.2 to frame no.1, some pixel values of the best match of frame no.1 will be changed and updated. This information will be used later for logo filling of frame no.3.

It is obvious that all invalid pixels of first frames of a shot, cannot be filled in one best match determination. So this method should be repeated until all pixels be valid. But it is expected that most of pixels in last frames are valid in one best match finding. According to our algorithm, the more shot frames be the better results will be.

The algorithm of our method is shown in Table 1.

## I. EXPERIMENTAL RESULTS

For experimental evaluation, our algorithm was applied to a video of a real soccer game. Fig. 4(a) is the frame no.1 of the second shot which the logo is shrunk after a few iterations (Fig. 4(b)) until it was omitted completely (Fig. 4(c)). In fact, as time goes on, more information is copied from the next frames to the previous frames and from the previous frames to the next frames and the logo is omitted smoothly.


Figure 2. Black regions are logo places. (a) Frame no.1 of first shot, (b) frame no.2 of first shot, (c) frame no.1 with a red rectangle around logo place, (d) frame no.2 with a red rectangle around logo place. (e) The frame no 15 (f) The best match for logo place in frame no.1 with respect to valid pixels.

.

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

(a)

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

(b)

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 | * | * | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 | * | * | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 | * | * | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | * | * | * | * | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

(c)

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | * | * | * | * | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | * | * | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | * | * | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | * | * | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

(d)

Figure 3. Number 1 shows a valid pixel and number 0 shows an invalid pixel. (a) Frame no.1 of first shot, (b) frame no.2 of first shot, (c) Frame no.1 that some of its invalid pixels have been filled(with *), (d) frame no.2 1 that some of its invalid pixels have been filled(with *).

TABLE1. The algorithm of iterative subsequent matching

**Iterative subsequent matching**

***Initialization:***

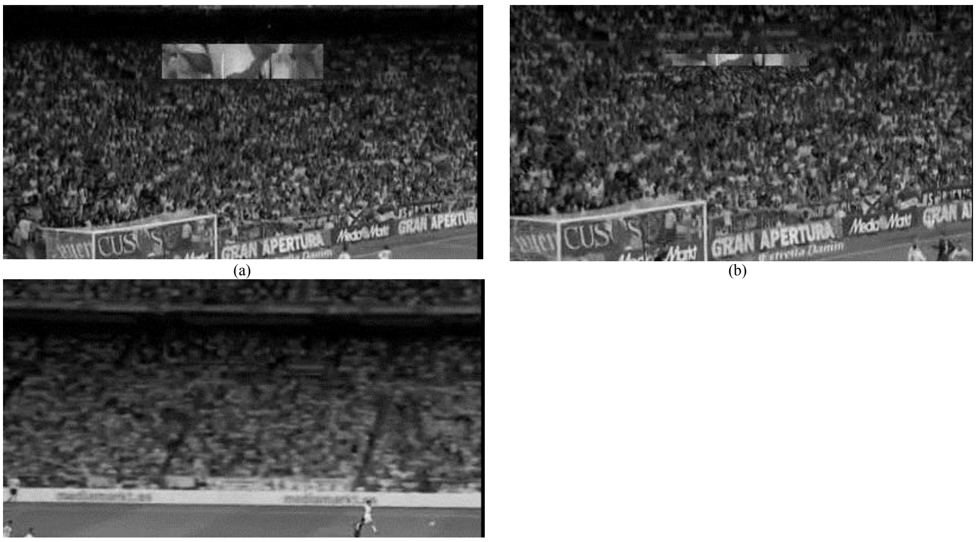
1. Define the video shots based on total average of difference of each two subsequent frames.
2. Set the pixels intensity of frames to 1 except for logo place which is set to zero value.

***Subsequent Matching:***

For every video shot do
  For every frame of a video shot do
    1. Define a window around the logo in current frame and one around the logo in next frame (temp0 and temp1).
    2. Move the window in next frame and Find the best match for current frame in the next frame (temp1).
    3. Copy the information which is only in the current frame (not in the next frame) to the next frame.
       temp1 ((temp1==0) & (temp0~=0)) =temp0 ((temp1==0) & (temp0~=0));

    4. Copy the information which is only in the next frame (not in the current frame) to the current frame.
       temp0 ((temp1~=0) & (temp0==0)) = temp1 ((temp1~=0) & (temp0==0));

  If all invalid pixels in all frames of shot have been filled end
  Else reverse index number of frames and do this loop again until all pixels be valid.
End



(a) (b)

(c)

Figure 4. (a) Frame 1, Shows the logo place. (b) Frame 102, the frame which the logo has been omitted. (c) Frame 143, Shows the logo place. (d) Frame 216, Shows the logo which is being omitted. (e) Frame 300, the frame in which the logo has been omitted.

When there is no blocking pixel for the logo, the logo will be preserved until a new frame from another video shot comes. Then the algorithm restarts and tries to remove logo from the new shot.

We have also compared our algorithm with the state-of-the-art general-purposed video inpainting algorithms on a same video (Table 2). Second column is the computational cost of these algorithms for processing a video. Third column is the Normalized mean Square Error (NMSE). It is the average difference between the result of our algorithm and the ground truth video in logo place.

It is observed that while our algorithm has a quality comparable with the well-known video inpainting algorithms, it is much faster than these algorithms so it can be used online. This was the main goal of this paper which wants to remove the logo as fast as possible. It is also worthy to express that some algorithm like [12] has scarified the speed for the sake of precision so they are useful for offline applications and they cannot be used for live TV application.

TABLE 2. Computational cost of video inpainting algorithm in a soccer game video of 1000 frames with 30 frames per second. Every frame has a 1024*680 pixels. Logo has a 10*30 pixels' area.

| Methods | Computational cost per frame | NMSE |
|---|---|---|
| Space-Time Completion Of Video [12] | 200.214 Second | 0.011 |
| Video Inpainting Under Constrained Camera Motion [13] | 28.654 Second | 0.094 |
| Video Completion Using Bandlet Transform [17] | 20.171 Second | 0.078 |
| Region Filling And Object Removal By Exemplar-Based Image Inpainting [10] | 5.322 Second | 0.080 |
| Broadcast Video Logo Detection and Removing [7] | 0.031 Second | 0.082 |
| Automatic TV Logo Detection, Tracking and Removal in Broadcast Video [6] | 0.032 Second | 0.083 |
| Online version of our Algorithm | 0.015 Second | 0.083 |
| Non-casual version of our Algorithm | 0.030 Second | 0.081 |

The main factor which causes NMSE of our algorithm increases, is the time between video shots which our algorithm has a new startup and it needs more frames to omit the logo. This error will be decreased in the subsequent frames considerably. So if we do not consider this state, the result of our algorithm will be much better.

## REFERENCES


[1] Schodl, R. Szeliski, D. H. Salesin et al., "Video textures," in Proceedings of SIGGRAPH, New Orleans, LA, 2000, pp. 489-498.

[2] M. Bertalmio, A. L. Bertozzi, and G. Sapiro, "Navier-Stokes, Fluid Dynamics, and Image and Video Inpainting," in IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2001 pp. 355-362.

[3] A. Kokaram, "Practical, Unified, Motion and Missing Data Treatment in Degraded Video," Journal of Mathematical Imaging and Vision, vol. 20, no. 1-2, pp. 163-177, 2004.

[4] V. Cheung, B. J. Frey, and N. Jojic, "Video epitomes," in CVPR, 2005.

[5] J. Jia, Y.-W. Tai, T.-P. Wu et al., "Video Repairing under Variable Illumination Using Cyclic Motions," IEEE Transactions on Pattern Analysis And Machine Intelligence, vol. 28, no. 5, pp. 832-839, 2006.

[6] J. Wang, Q. Liu, L. Duan et al., "Automatic TV Logo Detection, Tracking and Removal in Broadcast Video," in 13th International Multimedia Modeling Conference, 2006, pp. 63-72.

[7] C.-M. Kuo, C.-P. Chao, W.-H. Chang et al., "Broadcast Video Logo Detection and Removing," in Eighth International Conference on Intelligent Information Hiding and Multimedia Signal Processing, 2008, pp. 837 - 840.

[8] A. Efros, and T. Leung, "Texture synthesis by non-parametric sampling," in International Conference on Computer Vision, 1999, pp. 1033-1038.

[9] G. Doretto, A. Chiuso, Y. N. Wu et al., "Dynamic Textures," Int'l J. Computer Vision, vol. 51, no. 2, pp. 91-109, 2003.

[10] A. Criminisi, P. P´erez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," IEEE Transactions on Image Processing, vol. 13, no. 9, pp. 1200–1212, 2004.

[11] P. Sand, and S. Teller, "Video Matching," ACM Transactions on Graphics (Proc. SIGGRAPH), vol. 22, no. 3, pp. 592-599, 2004.

[12] Y. Wexler, E. Shechtman, and M. Irani, "Space-Time Completion of Video," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, no. 3, pp. 463 - 476, 2007.

[13] K. A. Patwardhan, G. Sapiro, and M. Bertalmío, "Video Inpainting Under Constrained Camera Motion," IEEE Transactions on Image Processing, vol. 16, no. 2, pp. 545-553, 2007.

[14] C.-H. Ling, C.-W. Lin, C.-W. Su et al., "Virtual Contour Guided Video Object Inpainting Using Posture Mapping and Retrieval," IEEE Transactions on Multimedia, vol. 13, no. 2, pp. 292-302, 2011.

[15] Hatheele and M. A. Zaveri, "Tracking based depth-guided video inpainting," in Computer Vision, Pattern Recognition, Image Processing and Graphics (NCVPRIPG), 2013 Fourth National Conference on, 2013, pp. 1-4.

[16] I. Daribo, and H. Saito, "A Novel Inpainting-Based Layered Depth Video for 3DTV," IEEE Transactions on Broadcasting, vol. 57, no. 2, pp. 533-541, 2011.

[17] A. Mosleh, N. Bouguila, and A. B. Hamza, "Video Completion Using Bandlet Transform," IEEE Transactions on Multimedia, vol. 14, no. 6, 20.