Exemplar-Based Object Removal in video using GMM

Aijuan Xia, Yan Gui
Dept. of Computer Science & Engineering
Shanghai Jiaotong University
Shanghai, China
Angela87@sjtu.edu.cn

Li Yao
Dept. of Computer Science & Engineering
Southeast University
Nanjing, China
yao.li@seu.edu.cn

Lizhuang Ma
Dept. of Computer Science & Engineering
Shanghai Jiaotong University
Shanghai, China
ma-lz@cs.sjtu.edu.cn

Xiao Lin
Dept. of Computer Science & Engineering
Shanghai Jiaotong University
Shanghai, China
lin6008@126.com

***Abstract*—This paper presents an exemplar-based video inpainting mechanism that restores the area of the removal object, and this mechanism can be further employed to extract the background of videos. The region to be inpainted in video is still in background and moving in foreground. Our method consists of a simple preprocessing stage and video inpainting step. The preprocessing stage consists in constructing Gaussian Mixture Model (GMM) for both background and foreground separately, then make use of GMMs to distinguish background and foreground of the entire video. That saves the time for calculating the optical flow mosaics as many video inpainting algorithms do in the preprocessing step. As for video inpainting, we firstly fill the gap as much as possible by copying information from other frames pixel by pixel, and then inpaint the remaining holes in the background by extending the exemplar-based image inpainting algorithm. Experimental results demonstrate that our method for object removal in video is feasible and effective.**

Keywords- object removal; video inpainting; GMM; image inpainting

I. INTRODUCTION

Inpainting [1] is a technique which is to modify an image in an undetectable way. The original application of this technique is to restore precious art works. Recently, we extend its application to restore the area of a removed object in a digital image for extracting the background of a still image, and then incorporate other objects to form a new image. Considering that the scenes in a Children's TV program do not change frequently, this technique can be adopted to reuse these scenes and then improve the efficiency of creating these programs.

Several algorithms for image inpainting have been proposed in the past few years. These algorithms can be divided into two types: methods [2] [4] based on the Partial Differential Equation (PDE) and methods [3] inspired by texture synthesis. Compared to techniques based on PDE, [3] can produce fairly good results in still images within comparatively short time, especially when applied to large continuous area. But [3] adopted full search to find the best matching patch, which slowed the process of inpainting.

As for video inpainting, some researchers extended techniques [2] [3] [4] for image inpainting to restore the objects in video. The author in [5] proposed a video inpainting algorithm employing image inpainting approach in [3] with static camera assumption. In this algorithm, the holes in the background of a frame can be filled in with exemplar patches in the same frame. Considering that the moving object may be partially occluded by other objects, a search mechanism between frames is also needed to find the best candidate patch. The video inpainting algorithm discussed in [6] further extended the algorithm in [5] so that it can cope with non-stationary video under constricted camera motions. The time consuming optical-flow mosaics were created to separate background and foreground of video. According to different camera motions, they can firstly adopt a motion layer segmentation algorithm to separate a video sequences. The approach in [7] followed this idea and used motion compensation and image completion algorithms to restore each layer. Then, all layers except the layer containing removal object were combined to attain the final video. However, the algorithm in [7] failed to consider the temporal continuity of video. And the author in [8] proposed a video inpainting algorithm which can deal with several kinds of camera motion, including rolling and scaling, on the condition that these motions did not change fast and fiercely.

In this paper, we present a method for inpainting the area of a removed object in video sequences. We follow the spirits of [3], but make use of Gaussian Mixture Model (GMM) [9] to shorten the time for finding best candidate patches. Besides, considering that the former video inpainting algorithms employ complex and time consuming preprocessing methods like generating flow-flow mosaics to separate the foreground and background of video, we adopt GMM to accelerate the separation while maintaining a reasonable accuracy. Note that our method focuses on those videos which have a static background and a moving foreground.

Section 2 describes the details of our technique. Section 3 presents the results and related analysis. The conclusion of our work and future work are presented in section 4.

II. OBJECT REMOVAL IN VIDEO BASED ON GMM

A. Overview

In [8], they divided the problems of video inpainting into three categories: stationary with moving objects, non-stationary video with still objects and non-stationary video with moving objects. The videos we deal with are those that have a static background and a moving foreground. There

978-0-7695-4356-7/11 $26.00 © 2011 IEEE
DOI 10.1109/CMSP.2011.169

are two main parts in our method (based on a key frame selected from any frame in video sequences, Figure.1):

- In the preprocessing step, we construct GMM for background and foreground separately. Later, these two GMMs are used to distinguish background from foreground.
- In the step of video inpainting, we firstly fill the missing background as much as possible by copying information from other frames pixel by pixel, and then inpaint the remaining holes(if they exist) in the background by extending the exemplar-based image inpainting algorithm.

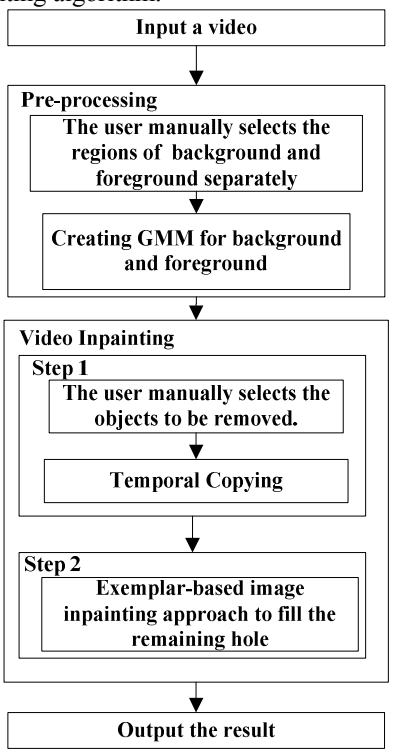


Figure 1. Overview of the proposed object removal in videos method

B. Preprocessing

We use the first frame (or any frame) as the base (called key frame). And the main task in this part is to create Gaussian Mixture Model for background and foreground of the key frame and then use these two GMMs to distinguish the background and foreground of the entire video.

The preprocessing step involves three procedures as following:

- Manually mark the regions of background and foreground separately as the input data to generate GMM;
- Adopt k-means algorithm to cluster the data roughly;
- Build up Gaussian Mixture Model via EM (expectation-maximization) algorithm, and GMM is defined as showing in the equation (1):

$$P(x_n)=\sum_k N(x_n|\mu_k,\Sigma_k)P(k) \quad (1)$$

where, the $N(x|\mu,\Sigma)$ is multivariate Gaussian density:

$$N(x \mid \mu,\Sigma)=\frac{1}{(2\pi)^{\frac{M}{2}}\det(\Sigma)^{\frac{1}{2}}}\exp[-\frac{1}{2}(x-\mu)\cdot\Sigma^{-1}\cdot(x-\mu)] \quad (2)$$

And EM algorithm is employed to estimate the means μ_k, covariance matrices Σ_k and mixture weights of each multivariate Gaussian density in the equation (1).

Note that the regions marked by the user should cover all kinds of information in the background or foreground. Otherwise the GMM we build cannot represent the feature of background or foreground. Considering that the EM algorithm is time and resource consuming, we first make use of k-means [10] for clustering data roughly so that the entire efficiency of building GMM can be improved.

C. Video inpainting

1) Image inpainting

Criminisi et al. [3] proposed exemplar-based image inpainting that can produce a fairly good result for large regions on digital images within comparatively short time. However, the algorithm in [3] adopted a time-consuming full search to find the best candidate patches and have some flaws in intensity continuity. However, our approach employs a new search mechanism to accelerate the process of finding the best matching patches and a gradient blending algorithm [11] to enhance intensity continuity of the image.

As described in the image inpainting approach [3], the user manually selected a target region (Ω) to be removed and filled and the size of template window Ψ (default size is 9×9 pixels) must also be specific. Then, the algorithm automatically computed the priorities of each patch on fill front (δΩ) to determine the filling order, and finally found the best candidate patches from the source region (Φ) to restore the target patch with the highest priority.

Let Ψ_p to be a patch centered at the point p (p∈δΩ)

$$P(p)=C(p)D(p) \quad (3)$$

where, C(p) is confident term which can be considered as the measure of reliable information surrounding the pixel p and D(p) is data term which includes the structure information of the fill front, and they are defined as follows:

$$C(p)=\frac{\sum_{q\in\Psi_p\cap(I-\Omega)}C(q)}{|\Psi_p|} \quad (4)$$

$$D(p)=\frac{\left|\nabla I_p^{\perp}\cdot n_p\right|}{\alpha} \quad (5)$$

where, $|\Psi_p|$is the area of Ψ_p ,α is a normalization factor (e.g. α=255 for a grayscale image), n_p is a unit vector which is normal to the boundary of target region δΩ and $\nabla I_p^{\perp}$ denotes the unit vector which is orthogonal to the image gradient. At the initialization step, the function C(p) is set to C(p)=1 ∀p∈Φ, C(p) =0 ∀p∈Ω. To calculate D(p), we follow the method in [12].Specifically, they calculated the unit vectors of unknown pixels in the patch Ψ_p to the center p and uniting the average of these vectors to get n_p. And they employed difference method to compute the image gradient and then determined $\nabla I_p^{\perp}$.

The approach in [3] employed a time-consuming full search mechanism to find the best candidate patches. However, our method adopts a new searching method to accelerate this process. Specifically, we firstly build the GMM to cluster the patches (the complete template, i.e. 9×9 pixels) of source region, according to the average RGB value of each patch. That is, we divide the source region into several different components and each component is a collection C_{Ψ_q} of patches that belong to the same GMM. As for a target patch, the average RGB value of known pixels represents the average value of entire patch. Then, we determine which components of the source region the target patch belongs via GMM. In this way, we can narrow down the search space to find the best matching patch. And the best candidate patch is found using the following formula:

$$\Psi_{\hat{q}} = \arg \min_{\Psi_q \in C_{\Psi_q}} d(\Psi_{\hat{p}}, \Psi_q) \quad (6)$$

where, the distance $d(\Psi_{\hat{p}}, \Psi_q)$ is defined as the sum of squared difference(SSD) of already filled pixels in the two patches.

Once the best candidate patch is found, in order to better keep the intensity continuity, we make use of gradient blending algorithm in [11] to blend the known pixels of the patch $\Psi_{\hat{p}}$ with the corresponding pixels in $\Psi_{\hat{q}}$ and then copy the information from the best matching patch $\Psi_{\hat{q}}$ directly to fill the unknown pixels in $\Psi_{\hat{p}}$. If we only consider the duplication of unknown pixels, there may be the color discontinuity in the target region. The gray area represents the composite of known pixels from both source patch $\Psi_{\hat{q}}$ and target patch $\Psi_{\hat{p}}$ (Figure.2). And the image blending algorithm will calculate the weights (α) of pixels in the overlapping area and it is calculated as a distance from the boundary of unknown pixels (white area in Figure.2):

$$N(x,y) = \alpha I(x,y)+(1-\alpha)C(x,y) \quad (7)$$

where, C(x, y) is the known pixel in the target patch $\Psi_{\hat{p}}$ and I(x, y) is the corresponding pixel in the source patch $\Psi_{\hat{q}}$.

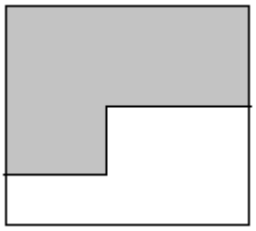


Figure 2. An example of blending pixels (Gray for the composite of known pixels and white for unknown pixels in target patch)

In general, our method for image inpainting for one frame of the video sequences can be describes as the following (the superscript t indicates the current iteration):

- Input an image and manually select the target region Ω;
- Calculate the average RGB values of source region Φ and build GMMs;
- Repeat until the region Ω is empty;

a) Compute boundary $\delta\Omega^t$ of target region Ω^t;

b) Calculate priorities P(p), $\forall p \in \delta\Omega^t$;

c) Find the patch $\Psi_{\hat{p}}$ with the maximum priority;

d) Estimate the average RGB values of target region Ω^t;

e) Determine the candidate collection C_{Ψ_q};

f) Find the best matching patch $\Psi_{\hat{q}} \in C_{\Psi_q}$ that minimizes $d(\Psi_{\hat{p}}, \Psi_{\hat{q}})$;

g) Copy image data from $\Psi_{\hat{q}}$ to $\Psi_{\hat{p}}$ $\forall p \in \Psi_{\hat{p}} \cap \Omega$, and blend the image data of $\Psi_{\hat{q}}$ and $\Psi_{\hat{p}}$ $\forall p \in \Psi_{\hat{p}} \cap \Phi$;

h) Update C(p), $\forall p \in \Psi_{\hat{p}} \cap \Omega$

2) Background extraction

In the preprocessing step, we build GMM for background and foreground based on the key frame. At this part, we use these two GMMs to separate the background and foreground of the entire video. This work plays a key role in the entire process. If the partition of background and foreground is not precise, we cannot get a sound result or the process of inpainting is time consuming. For more complex video sequences, optical flow mosaics [6] may be adopted to distinguish background and foreground. Considering that we are working on the videos with a static background and a moving foreground, GMM is enough in precision for our method.

Since that the values of the Gaussian density function for a given point will often be as small as to under flow to zero, we often work with logarithms of these densities. In our method, the logarithmic probability of a given point in a specific GMM is the maximization of logarithmic probability in its k components. Note that when employing GMMs to distinguish background and foreground, theoretically, we only compare the logarithmic probability in each GMM. In fact, when these two values are approximate to each other, we should set a threshold to distinguish them,

$$\ln(pb)-\ln(pf)<=threshold \quad (8)$$

where, pb is the probability of a point in background GMM and pf is the probability of the same point in the foreground GMM, and the threshold can be set to 2 or 3 (not more than 3). For example, if a given pixel satisfies equation (8), it belongs to the background.

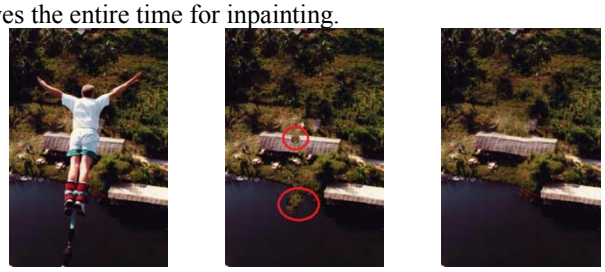
For any unknown pixel in the target region of background, we first choose the corresponding pixels in other frames as the source pixels, which constitute the candidates. Then we copy the image data from the best source pixel (the one with the greatest logarithmic probability in background GMM) into the target pixel. We

execute the aforementioned procedure to complete remaining pixels of the target region. After this step, if there are holes in the target region, we apply the method in section C-1 for obtaining the desirable background.

III. EXPERIMENTAL RESULTS

We use several images and video examples for experiments. All the experiments were on a 3.06GHZ Pentium IV with 1GB RAM.

Figure.3 shows the comparison of the result via our image inpainting methods and the result by the algorithm in [3] .Original image from [1] has the resolution of 205×307 pixels. From Figure.3 (b), we find that the flaws of the method [3] are apparent (marked with red circle in (b)). Comparatively, we maintain good structure and color continuity in these two circle-marked areas. The time for algorithm in [3] is 18'', while our method is 10.8''. That indicates that the new search mechanism we adopt actually saves the entire time for inpainting.



(a) (b) (c)

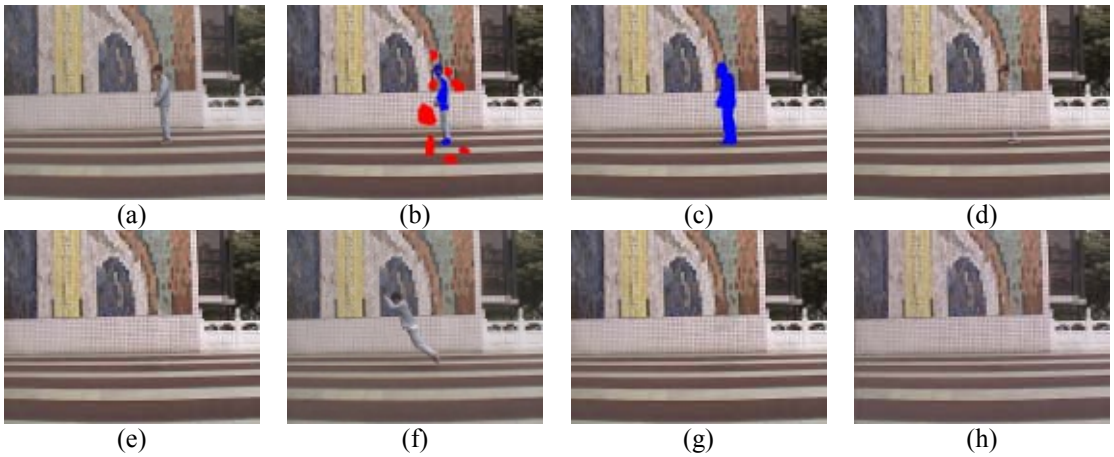
Figure 3. Comparing the results by our image inpainting methods with algorithm in [3]. (a) Original image from [1], 205×307 pixels. (b) The result (18'') by the approach in [3]. (c) The result (10.8'') via our methods.

Another application of image inpainting is demonstrated in Figure.4. When we take landscape photos, we often include the people in the landscape photos. In order to remove the figures, we can apply our proposed method to remove them and attain a complete landscape photo. Meanwhile, (c) presents that structure continuity and texture continuity can be well kept by our method even after two figures are removed in the same image.

(a) (b) (c)

Figure 4. Another example of our proposed method. (a) Original image. (b) Remove one figure. (C)Final result.

The resolution of video from [8] in Figure.5 is 320×240 pixels per frame. In Figure.5, (g) indicates that applying image inpainting method without considering the correlation among frames in video is weak in the continuity of structure. Compared to the result by approach in [8], our result is not as good as theirs. However, our method is much faster. Time for [8] is 0.332 seconds/frame, while our time is 0.0361 seconds/frame, almost one-tenth of [8]. The reason for our high efficiency is that we adopt time-saving GMM to separate the background and foreground of video while [8] employs time consuming mean-shift algorithm. As a result, for the video has a still background and moving foreground, GMM can be a sound tool to distinguish them.



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 5. An example of video inpainting. (a) The key frame. Based on this frame, we build the GMM for background and foreground. And the user manually selects the object to be removed. (b) The regions (red for foreground and blue for background) selected to build GMM. (c) The target region. (d) The middle result. (e) Final result of video inpainting. (f) A frame of the video. (g) The result of applying image inpainting method directly. (h)The result by the method in [8].

Another example from [8] is shown in Figure.6. The video in this example is also 320×240 pixels per frame and the background and foreground is very close to each other. In fact, GMM cannot distinguish very close background and foreground well. Thus, in order to improve the precision, when we select the regions on the background, we avoid those regions that are quite similar with the regions on the foreground .As presented in (e), our method can produce a fairly good result. And the total time used is 0.030 seconds/frame.

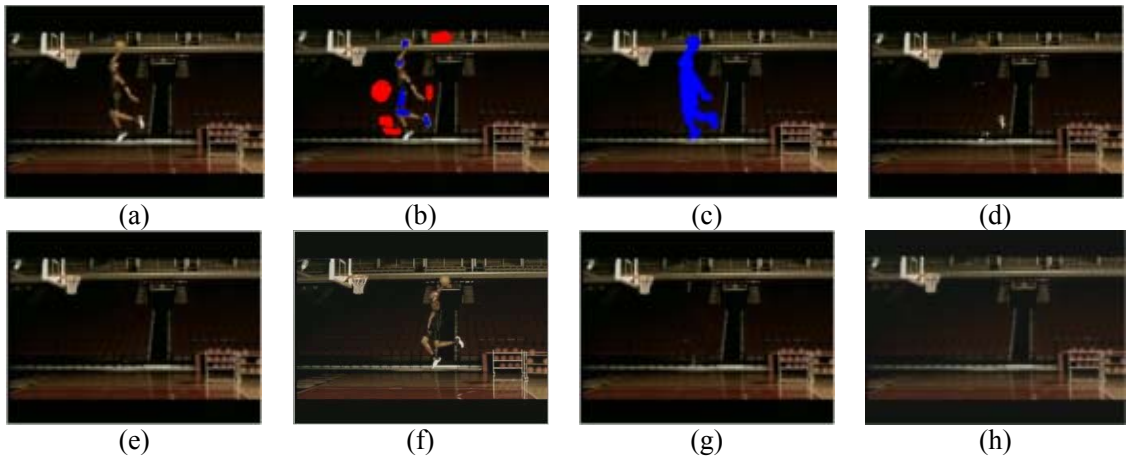


Figure 6. Another example by our method. (a) The key frame. (b) The regions (red for foreground and blue for background) selected to build GMM. (c) The target region. (d) The middle result. (e) Final result via our method. (f) A frame of the video. (g) The result of applying image inpainting method directly. (h)The result by the method in [8].

IV. CONCLUSION AND FUTURE WORK

In this paper, there are three major contributions: improvements of exemplar-based image inpainting, the separation of background and foreground using GMM, and how to improve the precision of GMM. We adopt the algorithm in [3] and combine GMM to accelerate image inpainting process and use a gradient blending algorithm [11] to improve intensity continuity. And we get reasonably good results. As for separation of background and foreground in the video that is still in background and moving in foreground, GMM is not only enough but also contributes to a high-efficiency of preprocessing stage. The methods we adopt to improve the precision of GMM also work well.

However, the limitation of GMM indicates that it does not do well in those videos that have quite close background and foreground, thus sometimes we cannot get ideal results via our method. Therefore, in our future work, we will incorporate optical flow to improve the precision of GMM. And since that our method aims only at videos with a static background and a moving foreground, we will extend our mechanism into other types of videos. Although the pure automatic video inpainting method does not exist, we can find some approaches to reduce the amount of human interference in our method.

ACKNOWLEDGMENT

This work is supported by 863 Program of China (No. 2009AA01Z334), 973 Program of China (No. 2006CB303105) and the National Natural Science Foundation of China under Grant No. 60803057.

REFERENCES

[1] M. Bertalmio, G. Sapiro, C. Ballester and V. Caselles, “Image inpainting,” *Computer Graphics, SIGGRAPH 2000*, pp. 417-424, July 2000.

[2] M. Bertalmio, A. L. Bertozzi, and G. Sapiro, Navier-stokes, fluid dynamics, and image and video inpainting. Proc. IEEE Computer Vision and Pattern Recognition (CVPR), 2001, Vol. 1, pp. 355–362.

[3] A. Criminisi, P. Perez, and K. Toyama, Region filling and object removal by exemplar-based inpainting, IEEE Transactions on Image Processing, 2004, Vol. 9, pp.1200–1212.

[4] I. Drori, D. Cohen-Or, and H. Yeshurun, Fragment-based image completion, ACM Trans. Graphics (SIGGRAPH), vol. 22, pp. 303−312,2003, San Diego, CA.

[5] K. A. Patwardhan, G. Sapiro, and M. Bertalmio, “Video inpainting of occluding and occluded objects,” in Proc. IEEE Int. Conf. ImageProcess., Sep. 2005, vol. 2, pp. 69–72.

[6] K. A. Patwardhan, G. Sapiro, and M. Bertalmío, “Video inpainting under constrained camera motion,” IEEE Trans. Image Process., vol.16, no. 2, pp. 545–553, Feb. 2007.

[7] Y. Zhang, J. Xiao, and M. Shah, Motion layer based object removal in videos, in Proc. 7th IEEE Workshop Appl. Comput. Vision, 2005,pp. 516–521.

[8] Timothy K. Shih, T.K.Tang, N.C.,and Hwang,J.N., Exemplar-based video inpainting without ghost shadow artifact by maintaining temporal continuity, IEEE Transactions on Circuits and Systems for Video Technology,2009,vol.19, no.3,347-360.

[9] William H. Press, Saul A. Teukolsky, William T. Vetterling and Brian P. Flannery, "NUMERICAL RECIPES: The Art of Scientific Computing Third Edition”, Cambridge University Press,2007,842-848

[10] William H. Press, Saul A. Teukolsky, William T. Vetterling and Brian P. Flannery, "NUMERICAL RECIPES: The Art of Scientific Computing Third Edition”, Cambridge University Press,2007,848-850

[11] Rankov V, Locke R, Edens R, Barber P, Vojnovic B. An algorithm for image stitching and blending. In: Proceedings of SPIE, three-dimensional and multidimensional microscopy: image acquisition and processing XII, vol. 5701; 2005. p. 190–9.

[12] Dongdong Nie, Research on the theory and algorithms of digital inpainting(数字图像和视频修复理论及其算法研究),[PhD thesis],Shanghai Jiaotong University, 2007