Zhang Y, Tang YL, Cheng KL. Efficient video cutout by paint selection. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 30(3): 467–477 May 2015. DOI 10.1007/s11390-015-1537-y

Efficient Video Cutout by Paint Selection

Yun Zhang [1] (张 赟), Yan-Long Tang [2] (唐延龙), and Ke-Li Cheng [2] (成可立)

[1] *Institute of Zhejiang Radio and TV Technology, Zhejiang University of Media and Communications Hangzhou 310018, China*

[2] *College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China*

E-mail: zhangyun_zju@zju.edu.cn; yanlongtang@gmail.com; chengkeli@zju.edu.cn

Received December 8, 2014; revised March 6, 2015.

Abstract Video cutout refers to extracting moving objects from videos, which is an important step in many video editing tasks. Recent algorithms have limitations in terms of efficiency, interaction style, and robustness. This paper presents a novel method for progressive video cutout with less user interaction and fast feedback. By exploring local and compact features, an optimization is constructed based on a graph model which establishes spatial and temporal relationship of neighboring patches in video frames. This optimization enables an efficient solution for progressive video cutout using graph cuts. Furthermore, a sampling-based method for temporally coherent matting is proposed to further refine video cutout results. Experiments demonstrate that our video cutout by paint selection is more intuitive and efficient for users than previous stroke-based methods, and thus could be put into practical use.

Keywords video cutout, progressive, graph cut, paint selection

1 Introduction

With rapid development of digital devices such as camera, smart phone and pad, videos can be easily obtained anytime and anywhere. Meantime, video sharing and social networks make the video data grow explosively. Given the ever-increasing availability of videos, it has become more and more important to manipulate them in an intuitive and efficient manner. Similar to image editing, video cutout is basic and important for many video editing operations[1-2]. For images, there are many commercial tools for object selection, like the quick selection and magic wand tool in Photoshop, which are very popular due to their simplicity and efficiency, and users can freely select region of interest. For videos, tools for object selection are neither robust nor efficient enough for users. Although Adobe company has provided Roto Brush in After Effects①, which can propagate foreground selection in keyframes to other frames, users always have to repair the propagation results frame by frame, which is labor-intensive.

Actually, video cutout is challenging due to the following reasons: 1) A large amount of data: even a 20 s short video contains at least 500 frames (size: 400 × 600), and thus algorithms for video cutout should be specially designed to handle a large amount of data. 2) Temporal coherence: directly processing video cutout frame by frame will generate discontinuous results, and people are usually very sensitive to this.

Now, the stroke-based image/video editing has become increasingly popular due to its simplicity and efficiency, and the essence is to propagate users' editing from strokes to other unmarked regions of an image/video, such as editing propagation[3-4], paint selection[5]. Inspired by [5], Tong *et al.* presented video brush[6], a new interface for progressive video object selection. Although being novel and creative, their

Regular Paper

Special Section on Computational Visual Media

This work was supported by the National High Technology Research and Development 863 Program of China under Grant No. 2013AA013903, the Zhejiang Provincial Natural Science Foundation of China under Grant No. LY14F020050, and the National Basic Research 973 Program of China under Grant No. 2011CB302205.

①Details of Roto Brush in Adobe After Effects. http://tv.adobe.com/watch/learn-after-effects-cs5/making-a-quick-matte-with-roto-brush/, Oct. 2014.

method is not efficient, and results are hard to control. In addition, the algorithm requires a large amount of memory and computation, and users cannot get fast feedback when dragging the brush, and thus it is far from practical use.

To address the problems in video brush[6], we propose *video paint selection*, an efficient solution for stroke-based progressive video cutout, which can largely reduce the memory and computation cost, and make the interactive video cutout more practical. In our approach, we first cluster pixels in each frame into several patches. Then, we construct a 3D graph to link spatial and temporal neighboring patches. Finally, the video cutout is configured to be a binary labeling problem, which could be solved by the graph cuts optimization. We further propose a sampling-based temporally coherent matting to refine the binary video cutout results. Experimental results show that our video paint selection can efficiently extract foreground regions with easy interactions and high accuracy.

Contributions of this paper include:

• An efficient solution is proposed for stroke-based progressive video cutout, which can provide users fast and accurate feedback with easy and intuitive interactions.

• A sampling-based temporally coherent matting is proposed to refine the binary video cutout results.

The remainder of this paper is organized as follows. Section 2 gives a brief survey of related work. In Section 3, we present the interaction tool and algorithm of video paint selection. Section 4 gives temporally coherent video matting which further refines the binary video cutout results. Experimental results and analysis are provided in Section 5. Finally, we conclude this paper in Section 6.

2 Related Work

Video cutout is a basic and important operation in many video editing tasks, and has always been a hot topic in computer vision and graphics. Hu *et al.*[7] presented a survey on Internet visual media processing, which provides recent progress of image/video analysis and editing. In video object selection, there exist two main categories: binary cutout and video matting[8]. Here, we briefly review recent work that is closely related to our paper.

Binary cutout refers to binary segmentation of foreground objects, which includes two main categories: keyframe propagation and graph cuts optimization. Agarwala *et al.*[9] applied an optimization scheme to implement a rotoscoping system based on the keyframe tracking, which can largely reduce the interactions in shape tracking, but cannot handle the topology changes, and thus can only extract objects with simple movement. In 2009, Bai *et al.* proposed video snapcut[10], which has already been applied in Adobe After Effects CS5. In video snapcut[10], users first accurately extract the foreground in keyframes, and then selections are propagated forward and backward according to a number of local features (color, texture, shape, movement, etc.) near the foreground boundary. By local classifiers, video snapcut can deal with foreground extraction from complex background. However, it may fail when the foreground moves fast or the topology changes, thus requiring a huge number of user interactions.

Recently, graph cuts optimization[5,11-12] has been widely applied to interactive image segmentation, and the quick selection tool in Photoshop is a successful application. Given the good performance in images, it is natural to extend it to videos. Li *et al.*[13] applied the graph cuts to video segmentation. They first accurately selected foreground in keyframes, and then constructed a 3D graph to express the spatial and temporal relations of video data. Finally the foreground was extracted by the graph cuts optimization. However, the global color model may result in incorrect segmentation in local regions, and thus they further proposed a local optimization based on tracking to improve the segmentation results. At the same time, Wang *et al.*[14] proposed a new interaction tool — video cube brush, and users can specify foreground and background regions by rotating, cutting and splitting the video cube. Furthermore, they proposed a hierarchical graph cuts method, which can efficiently extract foreground objects. However, this interaction tool is difficult for ordinary users, and the global optimization in the hierarchical graph cuts makes the process of foreground extraction hard to control. Different from Wang *et al.*[14], Tong *et al.*[6] allowed users to operate videos in a 2D manner. When users are specifying foreground regions by dragging a brush across frames, the video is playing forward and backward in a low speed, and thus video brush will leave a lot of foreground strokes in continuous frames. Finally, the cutout results are calculated progressively by the graph cuts optimization[11]. Although video brush is user-friendly, users cannot get fast feedback as they drag the brush across frames, and the local modifications may affect existing results. In addition,

the algorithm requires a large amount of memory and computation, and thus the method in [6] is still far from practical use. In this paper, we aim to improve the performance of video brush[6], and make this painting tool more efficient and practical.

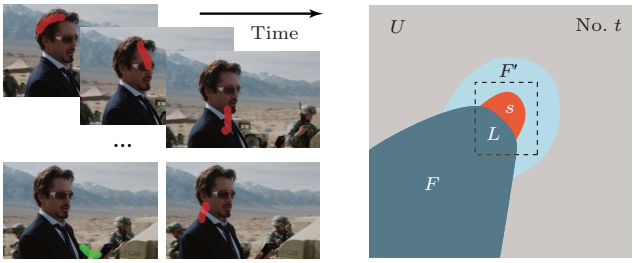
Video matting aims to refine the results of binary video cutout, which takes a source video and the corresponding trimap as input, and the output is the temporally coherent matte. Shahrian *et al.*[15] proposed a sampling-based method to improve the temporal coherence and spatial accuracy of video matting, which is achieved by sampling temporal and local samples that cover the foreground and background color distribution over all pervious frames. They also included a local texture descriptor to distinguish the foreground and background in low contrast videos. Ju *et al.*[16] modeled segmentation uncertainty in a novel tri-level segmentation procedure, which can deal with difficult cases like the topology changes. Zhong *et al.*[17] proposed an efficient video cutout system, which can deal with temporal discontinuities while remaining robust to inseparable foreground and background. In this paper, we propose a sampling-based video matting, and the temporal coherence is ensured by the temporally coherent filtering.

3 Video Paint Selection

In this section, we present the interaction tool and the algorithm of video paint selection.

3.1 Interaction Tool

Different from Wang *et al.*[14], who viewed a video as a 3D cube, we expand the video to be a number of continuous frames, and allow users to operate it in a 2D manner. Similar to paint selection[5], which enables progressive selection in images, our video paint selection can provide a tool to select video objects step by step as users paint across continuous frames. As shown in Fig.1, when users specify definite foreground/background regions across frames, the video is playing at a certain speed, and thus users' paintings will appear in continuous frames. The red and the green strokes refer to definite and unwanted foreground respectively, and the strokes are further used for the calculation of video cutout. In general, when the scene of a video is not too complicated, users only need to paint across a few frames, and the foreground regions could be easily extracted.



(a) (b)

Fig.1. Video painting tool and progressive selection. (a) Users' paintings across frames by progressive selection. (b) Progressive selection is triggered when the painting tool enters U, and new foreground $F^{'}$ is added to existing foreground F.

3.2 Algorithm

Given existing foreground F, our goal is to compute new foreground $F^{'}$ as users paint on continuous frames, and then the foreground region is updated as $F = F \cup F^{'}$. When users' painting P enters unknown region U, the calculation of $F^{'}$ is triggered. For robustness, when $F^{'}$ contains some disconnected regions, we only consider regions in each frame that connect with F as new foreground. Actually, the essence of the algorithm is to propagate the foreground selection from the seed region $S = P \cap U$ to spatially and temporally neighboring regions with similar appearance and locations.

To improve the efficiency and the robustness, we first cluster pixels of each video frame to be patches using mean-shift[18], which can cluster similar pixels while preserving the structure. The feature of each patch is simply represented by the average color in the patch.

Similar to previous studies in image segmentation[5,19-20], we configure the problem of video paint selection as binary labeling of the Markov random field. In this problem, the input video is represented by a 3D graph $\mathcal{G} = (\mathcal{P}, \mathcal{N})$, where $\mathcal{P}$ refers to all clustered patches in video frames, and $\mathcal{N}$ refers to all spatial and temporal links between neighboring patches. Figs.2(a) and 2(b) are source video frames and clustering results respectively, and Fig.2(c) shows the relationship of patches in and between frames. For neighboring frames, when two patches overlap and have similar features, there will be a link between them. The output of this problem is labels ($l_p \in \{0, 1\}$) for all patches in a video. When $l_p = 0$ or 1, p is labeled to be background or foreground respectively. The binary labels l_p can be efficiently calculated by the Markov random field[11], see (1).

$$E(l) = \sum_{p\in\mathcal{P}} D_p(l_p) + \lambda \times \Bigg(\sum_{(p,\,q)\in\mathcal{N}} V_{p,\,q}(l_p, l_q) + \sum_{(m,\,n)\in\mathcal{Q}} W_{m,\,n}(l_m, l_n) \Bigg). \quad (1)$$

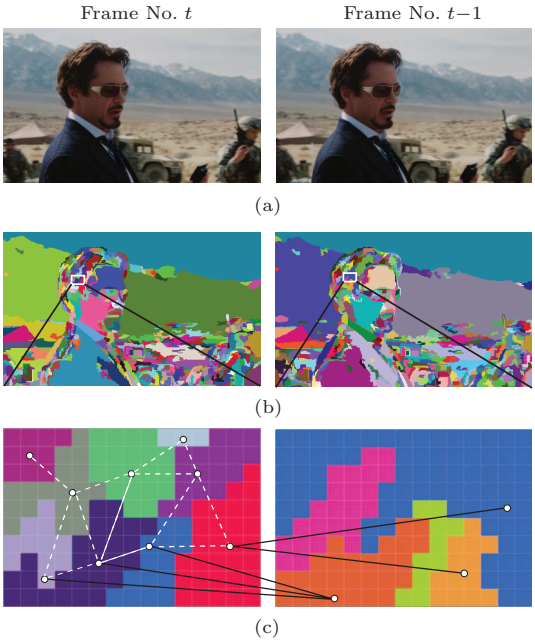


Fig.2. Video pre-segmentation and 3D graph construction. (a) Neighboring frames of a source video. (b) Clustering results of neighboring frames. (c) Links in and between neighboring frames based on the patches.

In (1), $E(l)$ is a linear combination of data term $D_p(l_p)$ and smooth term ($V_{p,\,q}$, $W_{m,\,n}$), and λ is used to balance the importance of the two terms. Finally, l_p can be obtained by minimizing $E(l)$. Details of the energy definition are given below.

$D_p(l_p)$ measures the distance between one patch and the foreground/background model, which is represented by the Gaussian Mixture Model (GMM), and the definition is similar to [6]. In previous studies[13-14], the GMM is constructed globally, which is not suitable for progressive selection, and current paintings may affect existing video cutout results. To make video cutout more robust, we construct the GMM locally and compactly. See Fig.1, $L \cup S$ is used to estimate the local foreground GMM (usually 2∼5 cores) G^f. Since the background is not specified directly, we first randomly sample a number of patches from U, and then replace patches which are labeled as foreground or similar to regions in $L \cup S$ by sampling the same number of patches from U. With suitable samples from U, the background GMM G^b is set up. Considering that different patches have different sizes, we give more weights to bigger patches in GMM construction, which makes the GMM more representative. In this paper, the weight of each patch is defined by the number of pixels in the patch. Similar to [5], the definition of $D_p(l_p)$ is given below.

$$D_d(l_p) = \begin{cases} (1-l_p)\times K, & \forall p \in S, \\ l_p \times K, & \forall p \in S^B, \\ l_p \times G_p^f + (1-l_p)\times G_p^b, & \forall p \in U \setminus (S \cup S^B), \end{cases} \quad (2)$$

where K is a large constant, $G_p^f = -\ln(g^f(p))$, $G_p^b = -\ln(g^b(p))$ refer to the distance between p and foreground/background GMM.

The smooth term consists of two components: $V_{p,\,q}(l_p, l_q)$, $W_{m,\,n}(l_m, l_n)$, and the definitions are as follows.

$$V_{p,\,q}(l_p, l_q) = |l_p - l_q| \times \exp(-\beta \times \|C(p) - C(q)\|^2), \quad (p, q) \in \mathcal{N}, \quad (3)$$

$$W_{m,\,n}(l_m, l_n) = \gamma \times |l_m - l_n| \times \exp(-\beta \times \sigma(m,n)^2), \quad (m,\ n) \in \mathcal{Q}, \quad (4)$$

where $V_{p,\,q}$ and $W_{m,\,n}$ refer to the costs of neighboring patches in and between frames respectively, which can ensure the spatial and temporal coherence of neighboring patches, and improve the performance of our video cutout. In (3), $C(\cdot)$ is the value of a patch, and $\mathcal{N}$ refers to all links in one frame. In (4), $\sigma(m,n)^2$ is the L_2 distance between neighboring patches, where $\sigma(m,n)^2 = \|C(m) - C(n)\|^2$. γ is used to balance the weights of smooth terms in one frame and between neighboring frames. To optimize the graph construction, not all neighboring patches have links between them, and only when they are similar enough ($\sigma(m,n)^2 < \varepsilon$), a link (m,n) is added to $\mathcal{G}$. $\mathcal{Q}$ refers to all links between neighboring frames.

3.3 Optimization

To provide users fast feedback in progressive selection, we propose the following optimization schemes.

1) For fast pre-segmentation and video cutout, we first downsample video frames to low-resolution ones, and then cluster them and calculate video cutout in low-resolution. Finally, we apply joint bilateral upsampling[21] to create an adaptive band in high-resolution level, and calculate the final video cutout by the banded graph cuts[22]. We can also employ the temporally coherent matting, which will be elaborated in Section 4, to refine the video cutout in the band region.

2) Similar to [6], we accelerate the 3D graph construction by pre-computing the smooth terms which are constants after the video pre-segmentation. Thus, during the process of video paint selection, only data terms are calculated, which largely reduces the computational cost.

3) We specify region of interest to reduce the data size in video cutout, and add hard constraints in definite foreground/background regions to make the video paint selection more effectively and efficiently. The implementation is as follows. $(D_d(l_p)) = (1 - l_p) \times K$ or $l_p \times K, \forall p \in S$ or S^B, see details in (2).

3.4 Implementation

We first add two terminal nodes s and t, which represent source (foreground) and sink (background) respectively, then add links from s and t to all nodes in $\mathcal{G}$, and finally obtain $\mathcal{G} = (\mathcal{P} \cup \{s, t\}, \mathcal{N})$. In $\mathcal{G}$, weights on the links from s, t to all nodes are defined by $D_p(l_p)$. $V_{p,\,q}(l_p, l_q)$ defines weights on links between neighboring nodes in one frame, and $W_{m,\,n}(l_m, l_n)$ defines weights on links between overlapped and similar nodes in neighboring frames. In general, it is hard to get the global optimization of (1), and thus we apply the graph cuts[11] to efficiently calculate the local optimization, and get the video cutout results according to the labels on patches.

There are four parameters to determine the performance of our video cutout: λ, β, γ, ε. We set $\lambda = 5$ and $\gamma = 10$ in our method, which places more importance on the smooth terms in one frame and between neighboring frames. β is used to determine the smoothness of results, and we set $\beta = 0.1$. ε is used to control the propagation of user specified strokes to neighboring frames, and we set $\varepsilon = 8$. For satisfying results, we have tested more than 40 videos to get the parameters above, which are always effective and robust for most videos. In experiments, we mainly adjust γ and ε for better results.

4 Temporally Coherent Video Matting

The results of video cutout are binary (0 or 1) and coarse near the boundary, and thus cannot be used to accurately extract transparent objects, such as hair, smoke and other foreground objects with soft boundaries. We further refine the binary cutout results to get high-quality video matte, which can be applied in a series of video editing operations, like video composition[1]. Inspired by [23], we combine sampling-based matting and temporal filtering to generate temporally coherent matte. Different from [10, 13], we ensure the temporal coherence by filtering, which avoids solving a large linear system, and thus can improve the efficiency. Fig.3 gives the algorithm flow chart, which is described as follows.

1) Generate temporally coherent trimap. Since the matting results depend on the quality of trimap, we first erode and dilate the binary video cutout results to generate initial trimap, and then fix the trimap of some frames by bandwidth adjusting and interactive editing.

2) Global sampling-based matting. We calculate the matte of each frame by the global sampling-based matting[24]. To avoid missing suitable foreground/background pairs, we expand the search from current frame to neighboring 2∼4 frames. See Fig.3(c), the red and blue points refer to the foreground and back-

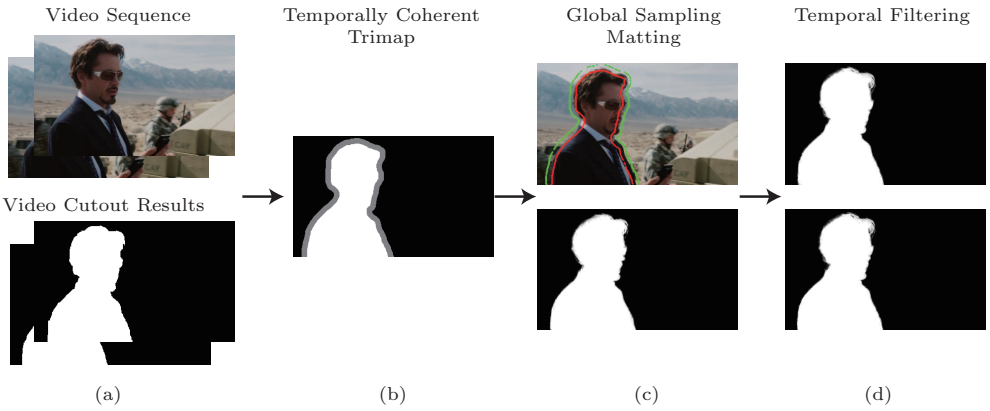


Fig. 3. Video matting flow chart. After obtaining (a) the binary cutout results, we first generate (b) temporally coherent trimap, and then compute the matte of each frame by (c) the global sampling; finally, the temporally coherent matte is calculated by (d) the temporal filtering.

ground samples in one frame respectively. To obtain satisfying foreground/background pairs efficiently, the random search[25] is applied.

3) Temporal matte filtering. Due to the variation of color, illumination, noises between neighboring frames, the frame-by-frame matting cannot ensure the coherence. We apply the temporal filtering to refine the frame-by-frame matte, see Fig.3(d). Given three continuous mattes α^{t-1}, α^t, α^{t+1}, the temporal matting is defined by (5), where $f_I(\cdot)$ is an interpolation function based on the level-set, see details in [23].

$$\alpha^t_{\text{new}} = f_I(\alpha^t, f_I(\alpha^{t-1}, \alpha^{t+1}, 0.5), 0.5). \tag{5}$$

Fig.4 gives video matting results. Fig.4(a) shows the binary cutout results of neighboring frames. Fig.4(b) shows the frame-by-frame matting results, which are not temporally coherent. Fig.4(c) shows the temporally coherent matting results, and the zoom-in view shows that our temporally coherent matting can generate high-quality matte with no flickering.

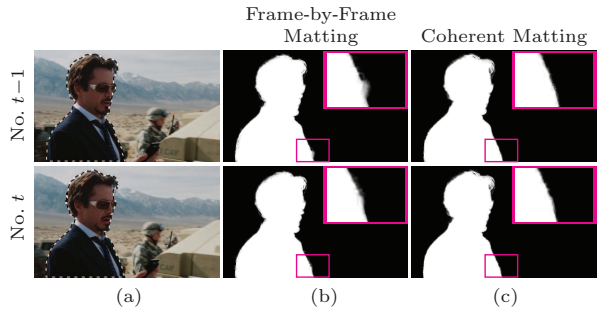


Fig.4. Video matting results. (a) Binary cutout results of continuous frames. (b) Frame-by-frame matting results. (c) Temporally coherent results by our method.

5 Results and Discussions

In this section, we show a number of examples and comparisons with state-of-the-art methods to demonstrate the advantages of our video paint selection. Furthermore, we discuss the effectiveness and limitations of our method. Our experiments are performed on a PC equipped with Intel Core 2 Duo E8400, 3 GHz CPU, 4 G Memory, and NVIDIA Geforce 8800. For better visual effects, boundaries of video objects are shown by black-and-white lines in all examples.

5.1 Results

Fig.5 describes the process of video paint selection in detail. The first row shows users' paintings in different frames. Below the first row, each row gives the cutout results in the same frame by different strokes, and each column shows the cutout results in different frames by the same stroke. Results in Fig.5 show that only a few strokes are enough to extract moving objects from a video. As shown in the first row, users can locally remove the unwanted foreground region by painting green strokes, and add missing foreground region by painting red strokes.

Fig.6 shows comparison of our method with video snapcut[10] and video brush[6]. Fig.6(a) shows accurate selection in a keyframe, and Fig.6(b) shows selection results by video snapcut with no user interactions. Fig.6(c) gives users' paintings in continuous frames. Figs.6(d) and 6(e) are results by video brush and our method respectively. It is obvious that our method is more effective and robust than state-of-the-art methods.

Fig.7 shows results of two challenging cases, which contain fast movement and low contrast in foreground and background. Results show that our method still performs well in some challenging cases, while other methods such as video snapcut[10] always require more user interactions, and their results are not satisfying.

Fig.8 gives more results of our video cutout. Figs.8(a) and 8(c) are the source videos. Figs.8(b) and 8(d) are the video cutout results, which are refined by temporally coherent matting. Observing from the results, we find that our method performs well in different cases, such as complex background, low contrast in foreground and background, and foreground with details like the hair.

Fig.9 gives applications of our video cutout. Figs.9(a) and 9(c) are the frames of source videos. Figs.9(b) and 9(d) show the composition results by alpha blending and gradient domain composition[1] respectively. The composition results are visual-pleasing, which demonstrates that our video paint selection can be used in video editing tasks.

We designed a user study to evaluate the effectiveness of our method. In the user study, we organized 25 undergraduate students (without experience in video editing) majored in computer science, liberal art, Radio & TV engineer, finance, and graphics design for user study, and asked them to extract foreground using our system, video snapcut[10] (implemented by Roto Brush) and video brush[6] respectively. See Fig.10, we developed a software system to implement the video paint selection. In this system, users can progressively extract foreground objects by the paint selection tool and observe the selected foreground in real time.

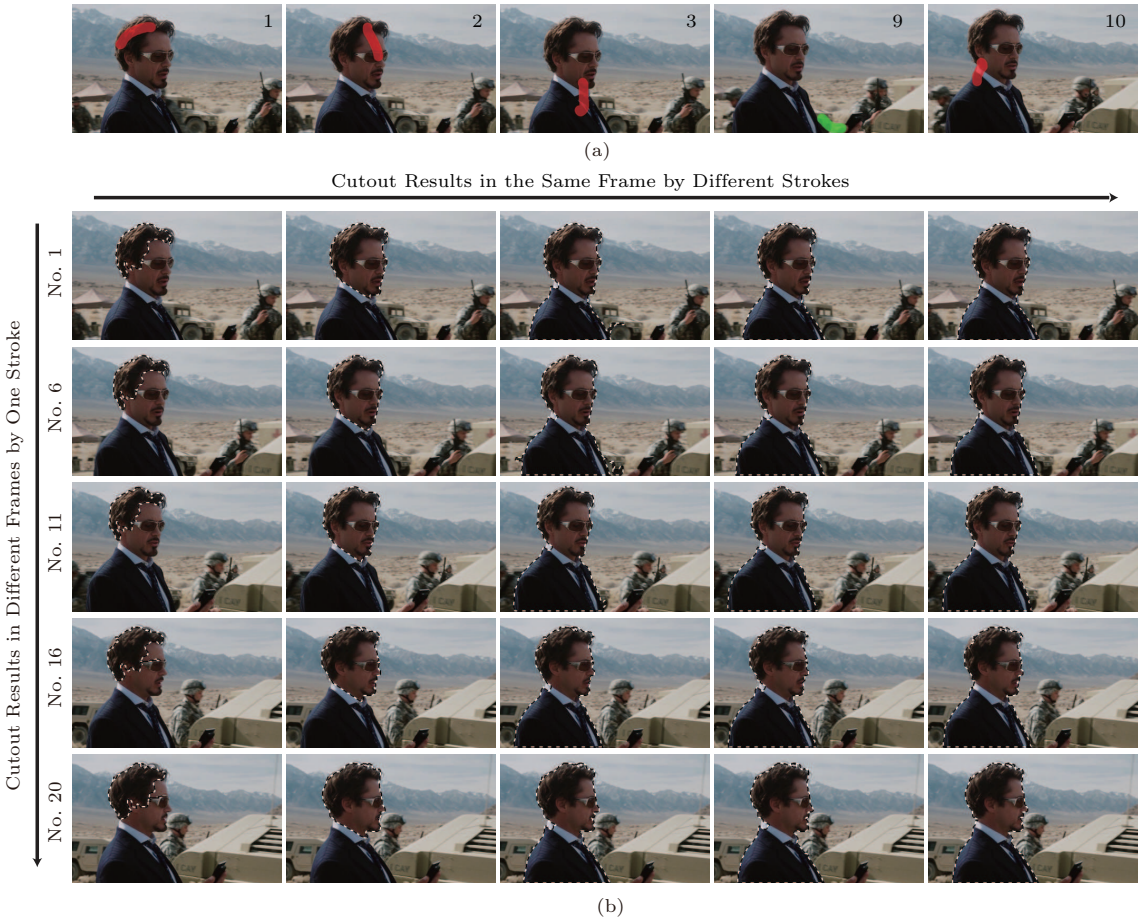


Fig.5. Video paint selection. (a) Strokes in different frames, and red/green strokes mark the foreground/background. (b) Each row shows the video cutout results in the same frame by different strokes, and each column shows video cutout results in different frames by the same stroke.

Table 1 gives comparisons of performance in video snapcut[10], video brush[6] and our method. The third column shows the average time of video foreground extraction by each painting, and the following four columns show the average number of strokes in each frame and the total time of foreground extraction by our paint selection, video snapcut[10], and video brush[6] respectively. In general, our method will be more efficient when the scene is simpler, as less patches are clustered. Observing from Table 1, we found that when scenes are relatively simple and contain no fast movement, the performance of the three methods above is better. However, when the background is complex, or the foreground moves fast, both video snapcut[10] and video brush[6] require more user interactions. Moreover, the local modifications in video brush may affect existing selections, and the calculation always costs much more time.

Furthermore, we conducted a survey to evaluate the diversity and consistency about users' feedback, and reported their average scores (0: definitely unsatisfied, 5: definitely satisfied) in answer to the questions as follows:

• Is the video paint selection tool intuitive and easy to use? (average score: 4.5)

• Does the video paint selection provide fast feedback as you paint across frames? (average score: 4.6)

• Are you satisfied with the robustness and accuracy of video paint selection results? (average score: 4.2)

• Do you think the video paint selection can be further put into practical use, and applied to video editing tasks, such as video composition, post production? (average score: 4.3)

5.2 Discussions

We give analysis on the effectiveness of our method. Firstly, we cluster video pixels into patches which can

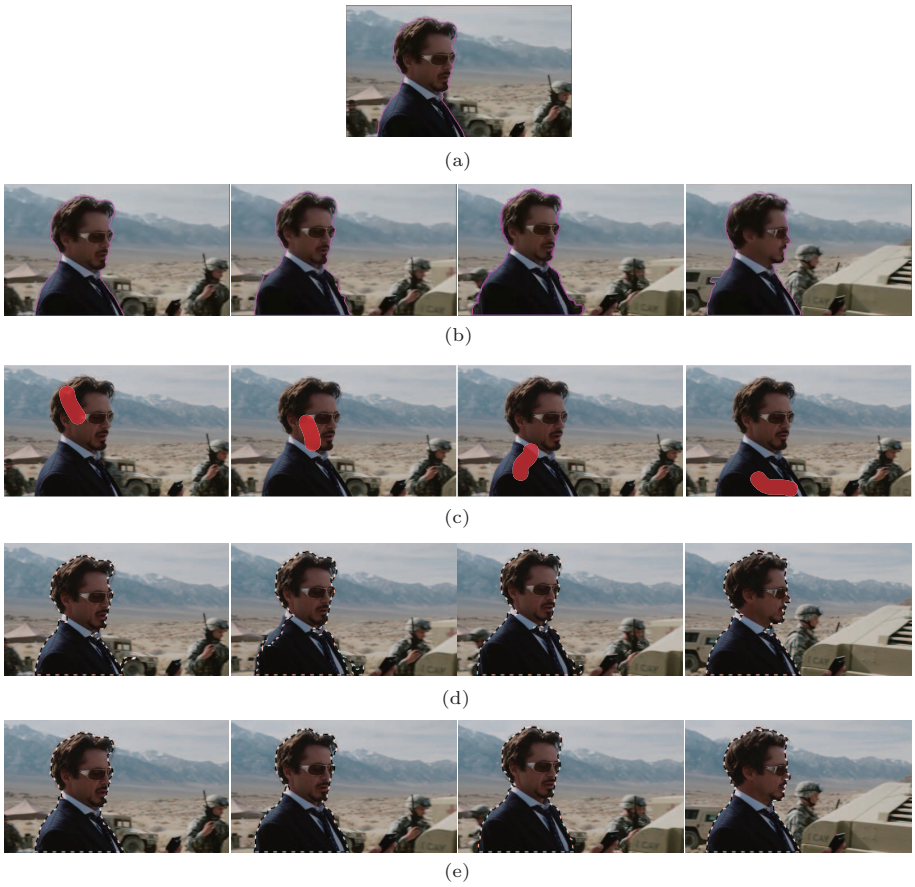


Fig.6. Comparison with video snapcut[10] and video brush[6]. (a) Accurate selections in a keyframe. (b) Results by video snapcut with no user interactions. (c) Users' paintings in continuous frames. (d) Results by video brush. (e) Results by our method.

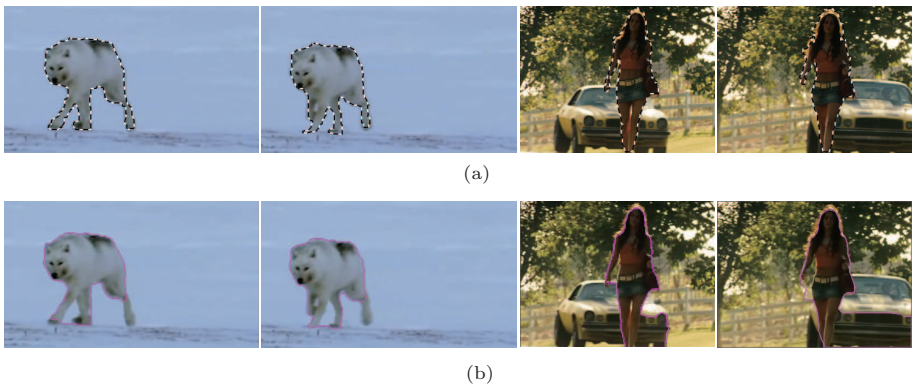


Fig.7. Challenging cases. Two examples of video cutout by (a) our method and (b) video snapcut[10]. The two examples are challenging, which contain fast movement and low contrast in foreground and background.

largely reduce the data size, and provide users fast feedback in video paint selection. In addition, our method propagates the foreground selection much faster in smooth regions. Secondly, we allow users to select objects as video plays, which can better express users' intention, and reduce the user interactions compared with the propagation-based method[10]. Thirdly, in the graph construction, we explore local and compact features to build spatial and temporal relations between patches, and consider similarities between neighboring patches, which can largely improve the robustness of video paint selection.

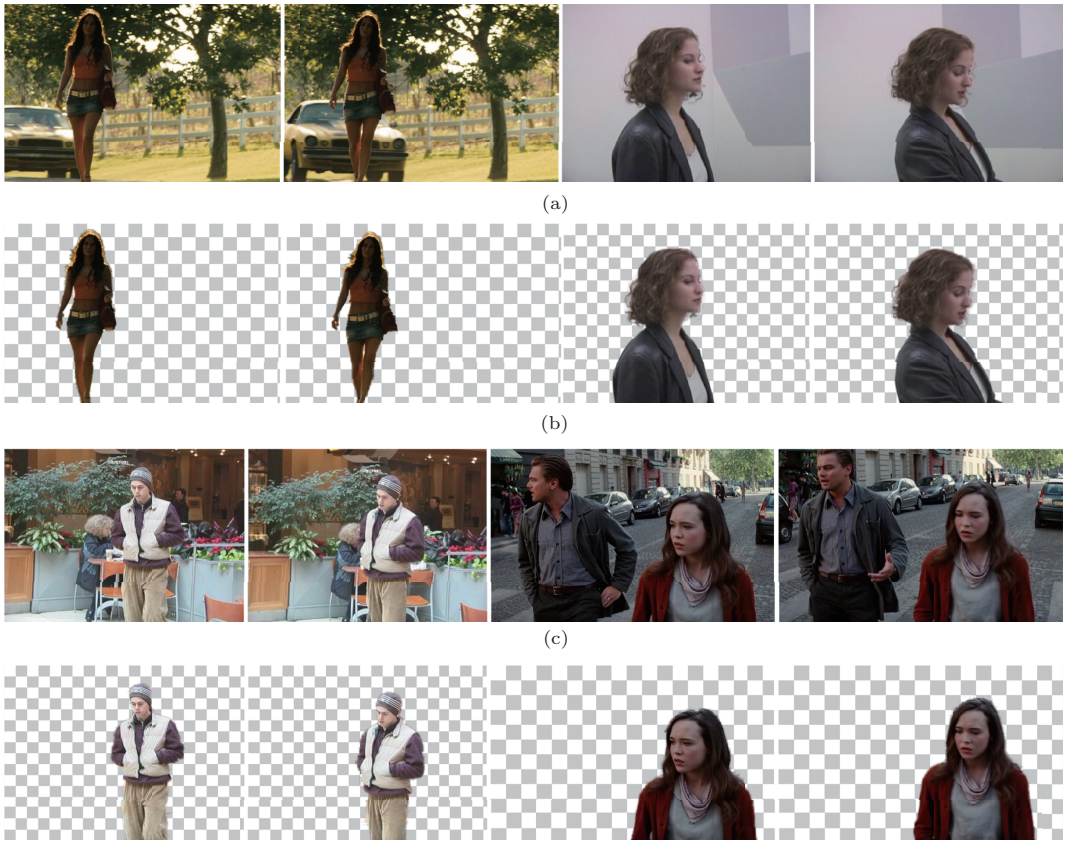


Fig.8. More results of our video painting. (a)(c) Source videos. (b)(d) Video cutout results, which are refined by temporally coherent matting.

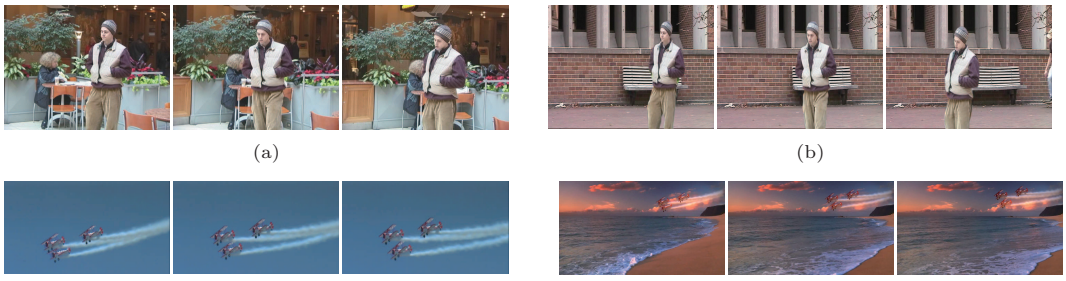


Fig.9. Applications of our method. (a)(c) Frames of two source videos. Composition results by (b) alpha blending and (d) gradient domain blending.

Table 1. Performance of the Proposed Algorithm

Video	Size	Avg. Time of Video Foreground Extraction by Each Paint (s)	Avg. Number of Strokes	Time of Foreground Extraction (s)		
				Our Method	Snapcut	Video Brush
Fig.3	$600 \times 240 \times 80$	0.516	1	13	17	32
Fig.7(a)-wolf	$1\,024 \times 276 \times 80$	0.477	2	25	34	50
Fig.8(a)-girl	$605 \times 406 \times 80$	0.610	1	16	23	35
Fig.8(a)-lady	$640 \times 480 \times 80$	0.540	1	14	20	33
Fig.8(c)-man	$640 \times 480 \times 80$	0.508	2	23	30	48
Fig.8(c)-girl	$562 \times 356 \times 100$	0.428	2	21	34	47

Similar to other methods[6,10], our method may fail when there exists fast movement, motion blur, or incorrect video clustering. Fig.11 gives failure cases, and Figs.11(a) and 11(b) are the foreground selection results by our method and video snapcut[10] respectively. Due to the incorrect clustering (left: feet and the ground are clustered together) and fast movement (right), both our method and video snapcut fail to produce satisfying results.

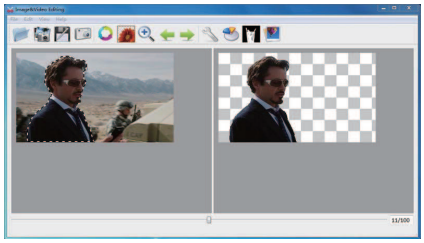
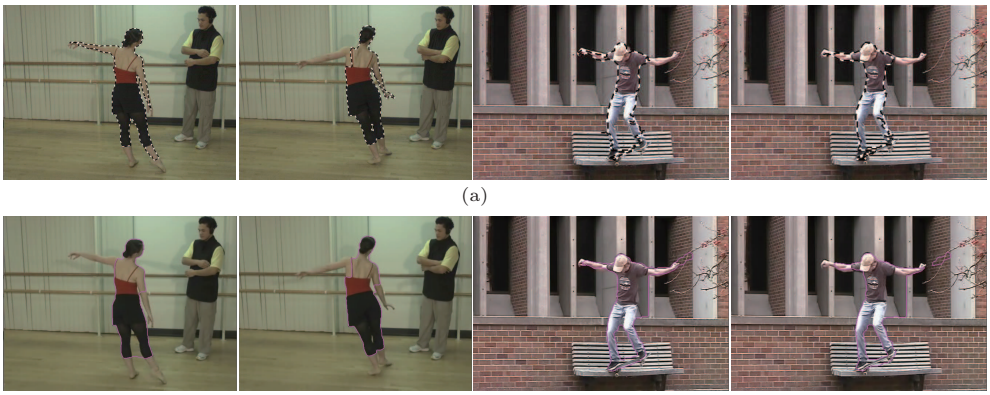


Fig.10. Software interface of our system. The left part is the source video, and the right part shows the selected foreground by progressive selection.



(a)

(b)

Fig.11. Failure cases. (a) and (b) are the foreground selection results by our method and video snapcut[10] respectively. Due to the incorrect clustering and fast movement, both our method and video snapcut fail to get satisfying results.

6 Conclusions

In this paper, we proposed video paint selection whose users can extract foreground regions progressively by painting across frames. We formulated the video cutout as a binary labeling problem. Firstly, we clustered pixels of each frame into patches according to the color and spatial similarities, which greatly reduces the size of video data. Secondly, we formulated an optimization based on a 3D graph, which constructs the spatial and temporal relationship among patches according to the local and compact features in frames. Finally, this optimization can be efficiently solved by graph cuts[11], and the binary cutout results are obtained according to the labels on patches. We further proposed a sampling-based method for temporally coherent video matting, which refines the binary cutout results. Finally, we gave results and applications of video paint selection to show the advantages of our method.

In the future, we will explore more effective algorithms to deal with complex texture, motion blur, fast movement, low contrast foreground and background, and design more user-friendly interface for better performance in video cutout. For better video matting, we will introduce optical flow[26] to ensure the temporal coherence.

Acknowledgement The authors would like to thank anonymous reviewers and editors for their valuable comments.

References

[1] Chen T, Zhu J Y, Shamir A, Hu S M. Motion-aware gradient domain video composition. *IEEE Transactions on Image Processing*, 2013, 22(7): 2532-2544.

[2] Lu S P, Zhang S H, Wei J, Hu S M, Martin R R. Timeline editing of objects in video. *IEEE Trans. Vis. Comput. Graph.*, 2013, 19(7): 1218-1227.

[3] Xu K, Li Y, Ju T, Hu S M, Liu T Q. Efficient affinity-based edit propagation using K-D tree. *ACM Trans. Graph.*, 2009, 28(5): 118:1-118:6.

[4] Ma L Q, Xu K. Efficient antialiased edit propagation for images and videos. *Computers & Graphics*, 2012, 36(8): 1005-1012.

[5] Liu J Y, Sun J, Shum H Y. Paint selection. *ACM Trans. Graph.*, 2009, 28(3): 69:1-69:7.

[6] Tong R F, Zhang Y, Ding M. Video brush: A novel interface for efficient video cutout. *Comput. Graph. Forum*, 2011, 30(7): 2049-2057.

[7] Hu S M, Chen T, Xu K, Cheng M M, Martin R R. Internet visual media processing: A survey with graphics and vision applications. *The Visual Computer*, 2013, 29(5): 393-405.

[8] Wang J, Cohen M F. Image and video matting: A survey. *Foundations and Trends® in Computer Graphics and Vision*, 2007, 3(2): 97-175.

[9] Agarwala A, Hertzmann A, Salesin D, Seitz S M. Keyframe-based tracking for rotoscoping and animation. *ACM Trans. Graph.*, 2004, 23(3): 584-591.

[10] Bai X, Wang J, Simons D, Sapiro G. Video SnapCut: Robust video object cutout using localized classifiers. *ACM Trans. Graph.*, 2009, 28(3): 70:1-70:11.

[11] Kolmogorov V, Zabih R. What energy functions can be minimized via graph cuts? *IEEE Trans. Pattern Anal. Mach. Intell.*, 2004, 26(2): 147-159.

[12] Rother C, Kolmogorov V, Blake A. "Grabcut": Interactive foreground extraction using iterated graph cuts. *ACM Trans. Graph.*, 2004, 23(3): 309-314.

[13] Li Y, Sun J, Shum H Y. Video object cut and paste. *ACM Trans. Graph.*, 2005, 24(3): 595-600.

[14] Wang J, Bhat P, Colburn A, Agrawala M, Cohen M F. Interactive video cutout. *ACM Trans. Graph.*, 2005, 24(3): 585-594.

[15] Shahrian E, Price B, Cohen S, Rajan D. Temporally coherent and spatially accurate video matting. *Comput. Graph. Forum*, 2014, 33(2): 381-390.

[16] Ju J L, Wang J, Liu Y B, Wang H Q, Dai Q H. A progressive tri-level segmentation approach for topology-change-aware video matting. *Comput. Graph. Forum*, 2013, 32(7): 245-253.

[17] Zhong F, Qin X Y, Peng Q S, Meng X X. Discontinuity-aware video object cutout. *ACM Trans. Graph.*, 2012, 31(6): 175:1-175:10.

[18] Comaniciu D, Meer P. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2002, 24(5): 603-619.

[19] Li Y, Sun J, Tang C K, Shum H Y. Lazy snapping. *ACM Trans. Graph.*, 2004, 23(3): 303-308.

[20] Huang H, Zhang L, Zhang H C. RepSnapping: Efficient image cutout for repeated scene elements. *Comput. Graph. Forum*, 2011, 30(7): 2059-2066.

[21] Kopf J, Cohen M F, Lischinski D, Uyttendaele M. Joint bilateral upsampling. *ACM Trans. Graph.*, 2007, 26(3): 96:1-96:5.

[22] Lombaert H, Sun Y Y, Grady L, Xu C Y. A multilevel banded graph cuts method for fast image segmentation. In *Proc. the 10th IEEE International Conference on Computer Vision*, Oct. 2005, pp.259-265.

[23] Bai X, Wang J, Simons D. Towards temporally-coherent video matting. In *Proc. the 5th MIRAGE*, Oct. 2011, pp.63-74.

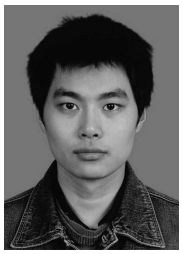
[24] He K M, Rhemann C, Rother C, Tang X O, Sun J. A global sampling method for alpha matting. In *Proc. the 24th IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2011, pp.2049-2056.

[25] Barnes C, Shechtman E, Finkelstein A, Goldman D B. PatchMatch: A randomized correspondence algorithm for structural image editing. *ACM Trans. Graph.*, 2009, 28(3): 24:1-24:11.

[26] Zhang S H, Li X Y, Hu S M, Martin R R. Online video stream abstraction and stylization. *IEEE Transactions on Multimedia*, 2011, 13(6): 1286-1294.

Yun Zhang is an assistant professor in Zhejiang University of Media and Communications, Hangzhou. He received his B.S. and M.S. degrees in computer science from Hangzhou Dianzi University, Hangzhou, in 2006 and 2009 respectively, and Ph.D. degree in computer science from Zhejiang University, Hangzhou, in 2013. His research interests include image/video editing and computer vision.

Yan-Long Tang is a Ph.D. candidate in College of Computer Science and Technology, Zhejiang University, Hangzhou. He received his B.S. degree in applied mathematics from Shandong University, Jinan in 2013. His research interests include image/video editing.



Ke-Li Cheng is a Ph.D. candidate in College of Computer Science and Technology, Zhejiang University, Hangzhou. He received his B.S. and M.S. degrees in communication engineering from Chongqing University, Chongqing, in 2007 and 2010. His research interests include image/video processing.