DOI: 10.1111/j.1467-8659.2011.02038.x

Pacific Graphics 2011
Bing-Yu Chen, Jan Kautz, Tong-Yee Lee, and Ming C. Lin
(Guest Editors)

Volume 30 (2011), Number 7

Video Brush: A Novel Interface for Efficient Video Cutout

Ruo-Feng Tong[1] Yun Zhang[†1] Meng Ding[1]

[1]Institute of Artificial Intelligence, State Key Lab of CAD&CG, Zhejiang University, China

Abstract

We present Video Brush, a novel interface for interactive video cutout. Inspired by the progressive selection scheme in images, our interface is designed to select video objects by painting on successive frames as the video plays. The video objects are progressively selected by solving the graph-cut based local optimization according to the strokes drawn by the brush on each painted frame. In order to provide users interactive feedback, we accelerate 3D graph-cut by efficient graph building and multi-level banded graph-cut. Experimental results show that our novel interface is both intuitive and efficient for video cutout.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Video/Image processing—Video/Image Editing

1. Introduction

With the easy access to the ever increasing number of digital media such as images and videos, editing such media in an intuitive and efficient way has become extremely necessary. Recently, the high-level image and video editing has attracted much attention, and a lot of methods have been developed for edit transfer [AP08, XLJ*09], image/video cutout and composition [CCT*09, BWSS09], image analysis and manipulation [BSFG09, CZM*10]. These methods can produce visual-pleasing results using only a few strokes. In general, the key to image and video editing is to select objects of interest. For 2D images, significant progress has been made in recent years, especially the state-of-the-art progressive selection tools such as *Quick selection* provided in Adobe Photoshop CS4 [Ado10] and *Paint selection* [LSS09] proposed by Liu et al. which enable users to get instant feedback during selection and make this tedious job to be more interesting and efficient. However, there is still no intuitive and effective user interface(UI) to select video objects with ease.

So far, the only video cutout system to be used in commercial production is *Video SnapCut* [BWSS09](renamed *Roto Brush*) which has been transferred to Adobe After Effects CS5. This method adopts a keyframe-based forward-propagation workflow, which means that the accurate segmentations in keyframes provided by users are propagated to the following frames based on a set of local classifiers. Although this system has been put into practical use, it does not conform to users' habit in object selection. Besides, since the interactions only happen in keyframes, the final results depend on accurate selections in keyframes and stable propagations. Thus when the video scenes are complicated, a lot of tedious manual corrections are still needed to obtain good results. In addition to this propagation-based method, there are some other attempts on solving this problem with the global optimization in 3D volumes [LSS05, WBC*05, APB07]. However, the global 3D graph-cut can not always produce good results as the foreground models are complicated which makes the selection process hard to control, and the user interactions in these systems are not intuitive and always too complex for non-professional users. In addition, some of these systems involve the time-consuming pre-segmentation which limits their practicability.

Inspired by *Paint selection* [LSS09], we propose *Video Brush*, a novel UI which selects video objects progressively with quick feedback. *Video Brush* is used to select video objects step-by-step in the same way as they do in 2D images. As we drag the brush across continuous frames, the video is played at a certain speed, and then 3D graph-cut is performed according to the strokes drawn by the brush on each frame. After we have painted a few video frames, a rough selection of the foreground objects is obtained. As the brush paints across several frames, our method can cope with the illumination and viewpoint changes of the foreground ob-

† Corresponding author: zhangyun_zju@zju.edu.cn

jects. The novelty of our *Video Brush* lies in the progressive selection and quick feedback which are very important for interactive video cutout. The progressive selection is achieved by solving the graph-cut based local optimization, and this makes the selection results more stable. Selection with quick response is a big challenge. Even a small video(e.g. 600*400, 100 frames) contains millions of pixels, thus we propose efficient graph building and 3D banded graph-cut for acceleration. To the best of our knowledge, we are the first to propose progressive selection of video objects. Compared with the previous video cutout systems, our method is superior in the following aspects:

- **Friendly user interface:** Our novel interface(*Video Brush*) allows users to select video objects progressively as they do in 2D images. Using this interface, users can select objects from a video easily in an intuitive way.
- **Stable and efficient selection:** As we drag the brush across frames, the video objects are selected progressively according to our intention, and this is achieved by solving a series of local optimization problems. Although we do not consider other features such as shape, texture which are used in *Video SnapCut*, the progressive nature makes our method more stable and effective. To provide quick feedback, we accelerate the 3D graph-cut by efficient graph building and 3D banded graph-cut.
- **Easy to implement:** Our problems can be solved by Region Push-Relabel(RPR) and 3D banded graph-cut, which can be easily implemented by optimizing the binary labeling problem.

The rest of paper is organized as follows. After briefly summarizing the related work in Section 2, we discuss our novel interface for interactive video cutout, and the 3D graph-cut algorithm used in progressive video cutout in Section 3. Experimental results and conclusions are given in Section 4 and 5.

2. Related Work

Over the past two decades, a huge number of researches have been conducted on image and video segmentation, which is summarized in [WC07]. We focus on the work of recent years that is closely related to our interactive video cutout.

2.1. Image selection

Selection is a basic but important step in image editing. Almost every commercial software for image processing provides various selection tools, and some of them are simple and naive, such as *Quick selection tool* and *Lasso tool* in Adobe Photoshop [Ado10]. The famous Normalized Cuts [SM00] treated the image segmentation as a graph partition problem and proposed a novel global criterion for segmentation. Recently, the seeded segmentation is very popular which includes graph-cut [BJ01, RKB04, BK04, SG07, DB08] and Random Walker algorithms [Gra06]. Their ability in image segmentation has been gradually improved. Generally speaking, these methods treat the image as a graph, and users need to roughly indicate some definite foreground and background pixels with a few strokes. Finally the segmentation results are obtained by minimizing certain energy functions. Although these approaches can generate high quality segmentations, they can not provide instant feedbacks and users have to work in an act-and-wait fashion. *Paint selection* [LSS09] and *Quick selection tool* in Adobe Photoshop CS4 [Ado10] are painting-based tools, in which users can make selections progressively by "*painting*" the objects of interest with instant feedback. This work fashion is more intuitive and suitable for such interactive selection tasks, thus can dramatically improve the user experience in image selection.

In order to get segmentations with high quality, matting technique is used to get the soft matte of complex boundary, such as hair and fur. Users have to provide *trimap* indicating foreground, background and unknown pixels near the boundary. Recently, [SJTS04] formulated the problem as solving a Poisson equation with the gradient of images by assuming that the colors in the foreground and background are smooth. Then Levin et al. [LLW08] proposed a closed-form solution to generate a high-quality matte by solving a sparse linear system. The state-of-the-art matting technique is shared sampling [GO10] which is based on the observation that pixels in a small neighborhoods tend to share similar attributes.

2.2. Video cutout

In general, existing methods for video cutout can be divided into two groups: 3D graph-cut and propagation based techniques. The former one treats the video as a 3D volume and uses the global optimization to select objects of interest. [LSS05] applied 3D graph-cut on the pre-segmented watershed regions from each frame for video cut-and-paste. [WBC*05] presented a 3D-based interactive system for efficient video cutout, and a novel volumetric painting interface was introduced in this work, which allows users to draw strokes and manipulate them directly on the spatio-temporal 3D video cube. However, the proposed UI is not intuitive and hard to use even for professional users, since it is difficult for users to access pixels by rotating, slicing and deforming the video cube. The global 3D graph-cut can not always produce good results as the foreground model is complex, and any local corrections users make in one frame may affect the existing good selections in the whole video. Besides, the pre-segmentation in these systems consumes too much time which affects their practicability.

In contrast to the global 3D graph-cut based methods, propagation-based methods take quite different strategies. They adopt a keyframe-based forward-propagation workflow, and users provide accurate segmentations in the keyframes which are then propagated to successive frames.

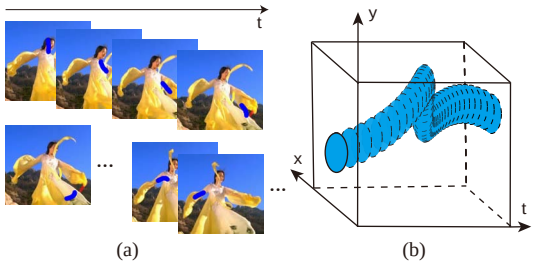

Figure 1: *User interface for progressive selection. As the brush moves across frames, strokes are left on the painted frames, and a tube is formed in the video volume.*

Many systems [CAC*02, LSS05, WBC*05, BS09] adopt optical flow to propagate the segmentations, using color information as the feature and constructing the classifiers with global color models. Relying on the global color statistics, they usually fail to distinguish foreground objects from the background in complex scenes. The state-of-the-art video cutout system *Video SnapCut* [BWSS09], which has been incorporated into Adobe After Effects CS5 [Ado10], collaborates a set of local classifiers with multiple features to make the propagation more reliable. However, according to our experiments, the propagation only works well in a relative short time period, and when colors of the scene are confusing or the foreground objects are experiencing some complicated issues, such as topology changes or occlusions, it turns to be unreliable and lots of local corrections are still needed. Another limitation of *Video SnapCut* is that those user refinements can only be propagated forward, which means that users must find the exact frame where the propagation begins to fail.

Our approach belongs to the first category, but it differs from the previous work in the following aspects. Firstly, we adopt a more intuitive and natural interaction manner to accomplish the equivalent spatio-temporal 3D interactions. Secondly, the proposed method works in a progressive way which is achieved by local optimization. Thirdly, instead of pre-segmenting the video frames, we use the Region Push-Relabel (RPR) and 3D banded graph-cut to provide quick feedbacks.

3. Video Brush

The main idea of *Video Brush* is a novel UI that assists users to select video objects in the same way as they do in 2D images. Different from previous video cutout systems [WBC*05, LSS05, BWS10], *Video Brush* allows users to select video objects progressively while providing quick feedback. In this section, details of our novel UI and segmentation algorithm are presented. We will introduce our UI in Section 3.1, and then give an overview of the algorithm in Section 3.2, finally the optimization details in Section 3.3.

3.1. User interface

According to most users' habit, object selection should be progressive and interactive. The reasons lie in the necessary user guidance in the procedure, as well as the evaluation of the segmentation results and whether further refinements are required. A good UI in this area should be kept with these characteristics, intuitive and easy-to-use. *Paint selection* [LSS09] is successful for its progressive nature which agrees with users' desire and ability to provide instant feedbacks. Unfortunately, existing video cutout systems can not provide such friendly UI. Hence, we present a novel UI – *Video Brush*, which allows users to select video objects step-by-step.

Similar to *Paint selection*, users can make a selection through painting the object with a brush. As shown in Figure 1(b), the video is treated as a 3D cube where X and Y axes are the spatial dimension of a single frame, and T axis represents the temporal dimension. As the *Video Brush* paints across several frames, the locus of the brush shown in Figure 1(a) forms a solid tube in this spatio-temporal space. To avoid complicated interactions when manipulating the 3D video cube, we adopt a manner used in video playing: when painting object of interest, video is played forward or backward at a certain speed. As shown in Figure 2(a), U is the background region, and S refers to the seed pixels which is the intersection of the brush and U. when the brush touches the background region U of a frame and begins to leave the frame, the progressive selection is triggered, and a new selection F' is computed which is then added to the existing selection F. Figure 2(b) shows that the new selection is expanded from S in one frame to the whole video when the progressive selection is triggered.

Actually, according to the local continuity nature of the video, it is unnecessary for users to paint over every frame, as the selection will be properly expanded from the current frame to the nearby frames. Thus, we allow users to stop the procedure at some frames and navigate the video to examine the obtained results, and then the progressive selection can continue from any position and any frame.

3.2. Algorithm Overview

We formulate the progressive selection as a binary labeling problem. The input is a video or image sequence, and a mask M in a certain frame indicates a new definite foreground region. For each pixel p, there is a binary label $x_p \in \{0,1\}$ (0 indicating background, and 1 indicating foreground). After the labeling optimization, the foreground area is expanded from M in the current frame to the whole video.

In general, the binary labels $X = \{x_p\}$ are obtained by minimizing the following energy function [BK04]:

$$E(X) = \sum_p E_d(x_p) + \lambda \sum_{p,q} E_c(x_p, x_q) \qquad (1)$$

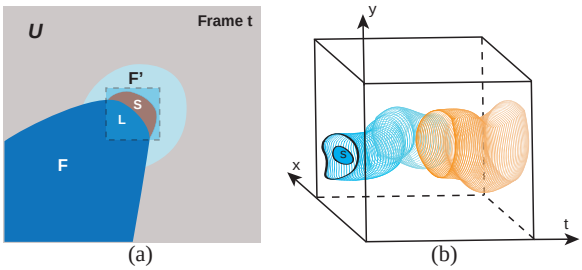

Figure 2: *Progressive video selection. (a) the selection is triggered when the brush touches the background region U of a frame and begins to leave the frame, and S (seed region) refers to the intersection of the brush and U. (b) when triggered, the selection is expanded from S to the whole video.*

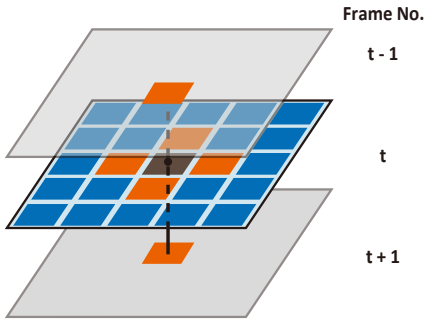

Figure 3: *Structure of the 3D grid-like graph. Each node has four neighbors in the same frame and two neighbors in the adjacent frames.*

where E_d is the data term which measures the conformity of a pixel in one frame to the foreground and background color models, and the models are constructed by fitting the Gaussian Mixture Model(GMM) [RKB04] with several components. E_c is the contrast or smooth term which measures the color differences between two neighboring pixels in the same frame and two adjacent frames. λ is a weight that balances the importance of data and smooth terms in the energy function. Since we select the video objects progressively, the data and smooth terms can be defined similar to that in *Paint selection* [LSS09].

$$E_d(x_p) = \begin{cases} (1-x_p)\cdot K & \forall p \in S \\ x_p \cdot K & \forall p \in S^B \\ x_p \cdot L_p^f + (1-x_p)\cdot L_p^b & \forall p \in U \setminus (S \cup S^B) \end{cases} \quad (2)$$

$$E_c(x_p, x_q) = |x_p - x_q| \cdot (\beta \cdot \|I_p - I_q\| + \varepsilon)^{-1} \quad (3)$$

where K is a large constant(set to 100000 in all experiments), $L_p^f = -lnp^f(I_p)$ and $L_p^b = -lnp^b(I_p)$ are the distances of a pixel to the foreground and background GMM respectively. $\beta = (< \|I_p - I_q\|^2 >)^{-1}$, and $< \cdot >$ is the expectation operator over all video frames.

The foreground and background GMM are constructed once the progressive selection is triggered. As shown in Figure 2(a), L is intersection of the existing selection F and the dilated S. $L \cup S$ is used to build the foreground GMM, which makes the foreground estimation more compact and stable. As we do not indicate the background region directly, the background GMM is initiated by randomly sampling a number of pixels from the background of all frames. After each user interaction, the samples which are labeled as foreground are replaced by new samples from the background.

Generally speaking, minimizing the energy function (1) falls into the categories of either *augmenting path* [BK04, EK72] or *push-relabel* [GT88]. For a 2D image which can be regarded as a regular 2D grid graph, BK(Boykov-Kolmogorov) algorithm [BK04] which is used to solve the max-flow/min-cut problem offers the best performance. However, even a short video with 100 frames(640*480) contains more than 30 million pixels which makes the BK algorithm impractical in terms of memory and time. Thus, we introduce the region push relabel(RPR) [DB08] which has good performance on immense graphs. We construct a regular 3D grid-like graph $\mathcal{G} =< \mathcal{V}, \mathcal{A} >$ for the input video. The node set $\mathcal{V}$ consists of the pixels in all frames, and the arc set $\mathcal{A}$ contains arcs that connect the adjacent 4 nodes in the same frame and arcs that connect the corresponding nodes in the adjacent frames. See the structure of 3D grid-like graph in Figure 3.

3.3. Optimization

Although the performance has already been improved by applying RPR [DB08], it still can not support the interactive-level operations. In order to select video objects with quick feedback, we further accelerate our progressive selection from the following two aspects: (1) accelerate the graph building in 3D graph-cut; (2) adopt the multilevel banded optimization, which performs 3D graph-cut in a down-sampled video space, and then use 3D banded graph-cut to obtain the full resolution results in the original video space.

3.3.1. Efficient graph building

In our experiments, it is time-consuming to build a grid-like graph for a video, e.g. a video with 100 frames(640*480) contains more than 30 million nodes, and most of the nodes may contain 6 neighbors. Generally, it requires more than 7 seconds to build a 3D grid-like graph even with the multi-core acceleration. Furthermore, in our video cutout system, a new 3D graph needs to be built once the progressive selection is triggered. Thus, we need to accelerate the graph building so as to provide users quick response. From Equation

(3), we find that the smooth terms are fixed once the video is loaded. When the progressive selection is triggered, only the data terms have to be changed according to the interactions of current frame. In fact, the most time-consuming part in graph building is adding the smooth terms to the graph.

Thus, we figure out a solution to construct the immense graph without adding the smooth terms explicitly, and the idea is simple but effective. When we set the smooth terms for the first time, for every node, its residuals with its connecting nodes are not only added to the graph, but mapped to a dynamic memory M_1, and then we make a copy of M_1 (refer to it as M_2) to store the original residuals of every node. After calculating the max-flow, the residuals of related nodes are changed. Thus, when the progressive selection is triggered again, M_1 can be quickly updated by copying M_2 directly, and the smooth items are reset automatically.

3.3.2. 3D Banded Graph-Cut

For efficient graph building, we have to allocate a huge number of memory (usually more than 500MB) to restore the smooth terms of the 3D graph, which always exceeds the limit of our system. Instead of pre-segmenting video frames using the traditional Watershed [LSS05] or Meanshift [WBC*05] to reduce the scale of data, which is time-consuming. We adopt the 3D banded graph-cut [LSGX05] to solve this problem. Firstly, we down-sample the video to a coarse level, and then 3D graph-cut solved by RPR are performed on the low resolution video. The structure of the graph is a regular grid-like graph which is shown in Figure 3. After obtaining the selection results on the coarse level, the narrow bands are generated in each frame which are further up-sampled to the original resolution, and then graph-cut is performed on the 3D band to get the final result. In general, the 3D banded graph-cut is very efficient, as the number of pixels on the band is largely reduced compared with the whole video. However, since the graph is arbitrary and small, we prefer the BK algorithm [BK04] which has superior performance for the small scale graph.

4. Results and Comparisons

4.1. Results

We first demonstrate the selection results using our *Video Brush*. As shown in Figure 4, the video about two men talking contains 60 frames(size: 640*272), and the top row shows the user's paintings across continuous frames using *Video Brush*. After painting, a lot of strokes are left on several frames which are used for the progressive selection. In the following part, each row shows the selection results on the same frame after different user interactions, and each column is the selection results on different frames after the same user interactions. The results show that user interactions in one frame can be propagated forward and backward to other frames effectively. As the video is not very complicated, the man is well selected after the brush paints only a few frames.

data	size of video	progressive selection	*Video Brush*	*Video SnapCut*
talking	640*272*60	1.115s	24s	15s
dancing	640*272*50	1.12s	18s	40s
Tom	480*320*80	1.01s	14s	25s
skiing	640*272*50	2.1s	30s	20s
Jerry	640*480*60	1.11s	16s	26s
fish	480*320*50	1.2s	20s	50s

Table 1: *Performance of examples used in this paper. Each row shows the running time of video selection in three aspects: progressive selection refers to the average time of each progressive selection; Video Brush and Video SnapCut refer to their total time used for video selection.*

Figure 5 compares the results of our method with global 3D graph-cut and the state-of-the-art video cutout system *Video SnapCut* [BWSS09]. The top two rows show the user interactions on successive 11 frames as well as the keyframe, and the following 3 rows show the initial selection results on several frames using *Video Brush*, global 3D graph-cut and *Video SnapCut* respectively. For *Video Brush*, the selection is performed progressively as the brush paints across continuous frames. While for the global 3D graph-cut, the selection is performed globally on the video volume after the brush paints several frames. The last row shows the selection results using the *Video SnapCut*, and the keyframe object shown on the second row is selected by users. After that, the selection result is propagated to the following frames. As shown from the zoom-in views, our *Video Brush* performs better than the global 3D graph-cut and *Video SnapCut*. As we select the video objects progressively, which is guided by users' intension according to the feedback, the selection may be more stable and better adaptive to the video scenes. The global 3D graph-cut is not better than our method, because the foreground model is constructed globally which makes the selection unstable and uncontrollable. The *Video SnapCut* may fail when the selections on keyframes are not precise, or there is fast movement and motion blur. As shown in the results, after the keyframe propagation, a lot of local corrections are needed. In addition, our method can deal with the viewpoint changes in video selection. In this experiment, although the women suddenly appears to be much nearer since frame 55, she is correctly selected using our method.

Figure 6 is another comparison of our method with *Video SnapCut*. Although the video is about a simple cartoon character *Tom*, *Video SnapCut* fails to cope with the fast movement of *Tom*, while our progressive selection performs well after a few user interactions on the first row.

4.2. Performance

We report the performance on a PC equipped with a 2.27GHz of Intel(R) Xeon(R) CPU E5520, 16GB memory and NVDIA GeForce GTS 250 in Table 1 for the examples

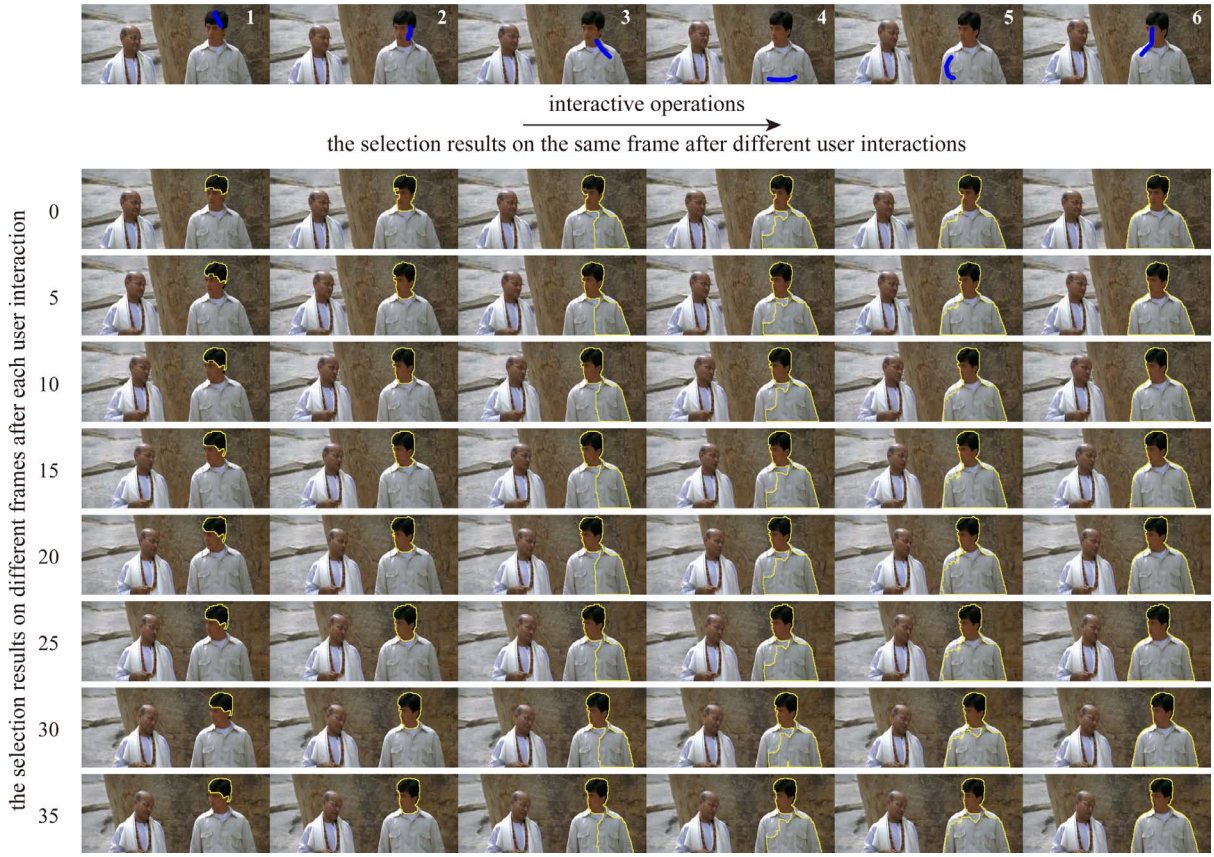

Figure 4: *Progressive selection using Video Brush to select the talking man. The top row shows user interactions on different frames, and the following part shows the progressive selection results in several frames. Each row represents the selection results on the same frame after different user interactions, and each column shows the selection results on different frames after each user interaction.*

in this paper. We report the total time used for video selection by *Video Brush* and *Video SnapCut*. The users are college students who are not familiar with video editing. Before the experiments, they had been told how to use this system and content of the videos. They segmented every video sequence twice, and the *Video Brush* was used first. The progressive selection refers to the average time of each progressive selection by our approach. Observing from the table, we find that our method is very efficient in video selection. When the foreground objects are very simple with big smooth area, *Video SnapCut* is a bit more efficient, e.g. talking and skiing. However, when a lot of user interactions are needed to make good segmentations in keyframes or there is obvious fast movement, *Video SnapCut* usually consumes much more time than our method, e.g. *Tom*, *Jerry*, fish and dancing. As we select the video objects by users' intention based on the feedback, the selection results are more precise and less user interactions are needed to make local corrections, thus the time cost for good segmentations can be largely reduced. In this paper, the criterion to decide a good segmentation can be shown in the following applications.

4.3. Applications

Video composition In video composition, the extraction of objects is an important step. In general, the composition consists of simple cut-and-paste and gradient-based methods. Figure 7 is an example of gradient-based composition, and we apply the MVC-based [FHL*09] method to pasting the video objects into the target scene seamlessly and naturally. As shown in Figure 7, the skiing man is first selected by our method, and then be cloned into the desert. Although the object does not have a clear boundary, our method still works well, and the final results seem natural and vivid.

Figure 8 shows an example of video composition by direct cut-and-paste. The first row shows the user interactions on continuous frames. After progressive selection, the foreground object *Jerry* with a brush is extracted from the origi-

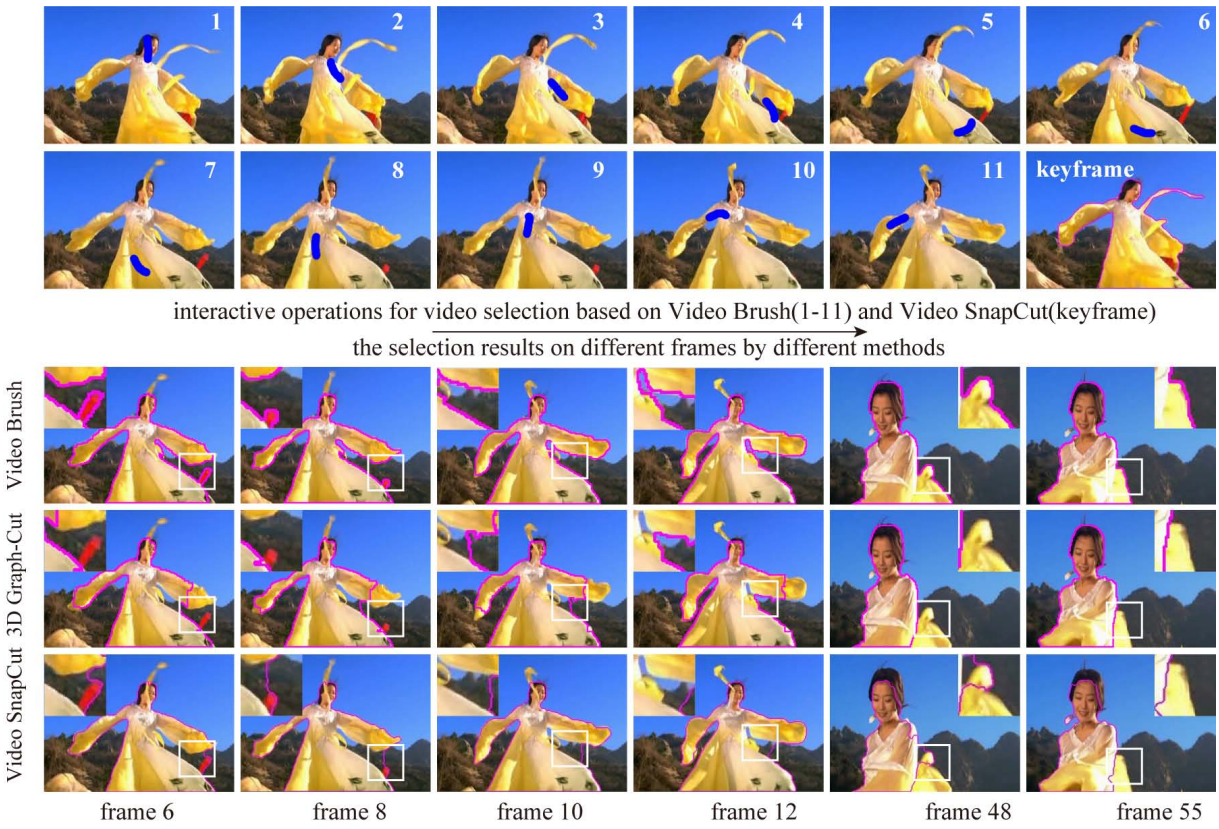

Figure 5: *Comparisons with global 3D graph-cut and Video SnapCut [BWSS09]. The first two rows are the user interactions by Video Brush(1-11) and Video SnapCut(keyframe). The following three rows show the initial selection results using Video Brush, global 3D graph-cut and Video SnapCut respectively, and the zoom-in views show the comparisons of the three methods.*

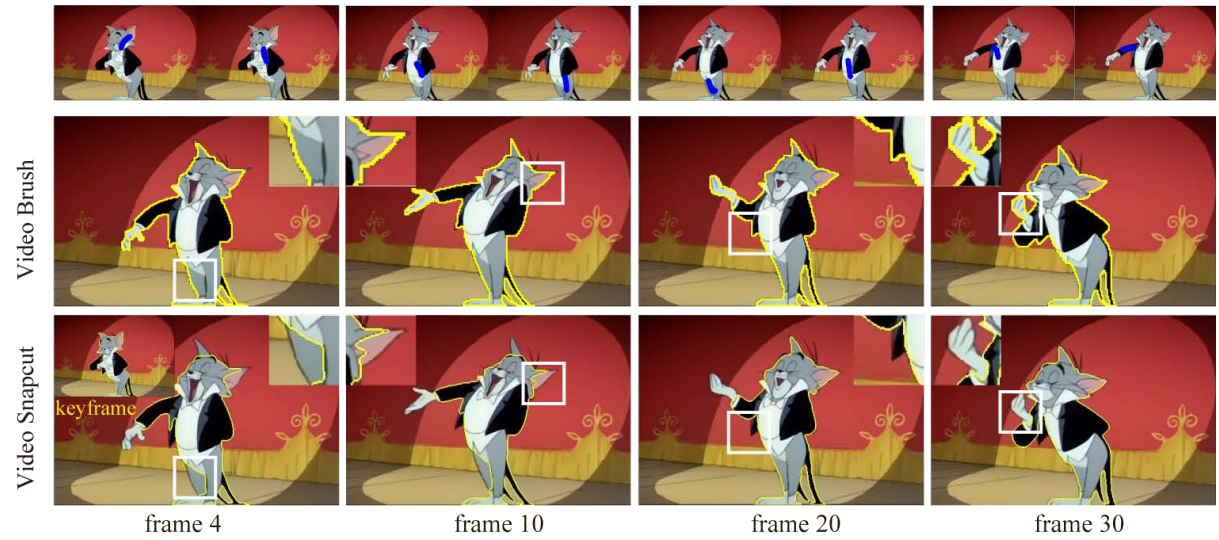

Figure 6: *Another comparison with Video SnapCut [BWSS09]. First row is the user interactions by Video Brush. The following two rows show the initial selection results using our method and Video SnapCut(the keyframe segmentation is shown on the up-left corner), and the zoom-in views show the comparisons of the two methods.*

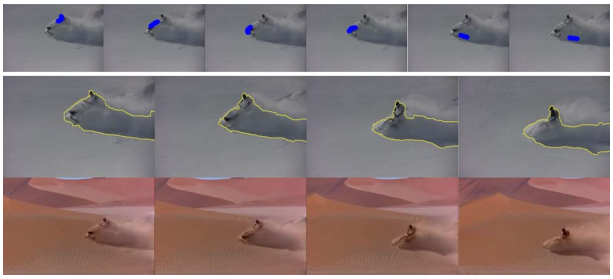

Figure 7: *MVC-based [FHL*09] blending of selected video objects. The first row is the user interactions by Video Brush, and the following two rows correspond to four original frames with the skiing man selected and MVC-based blending results.*

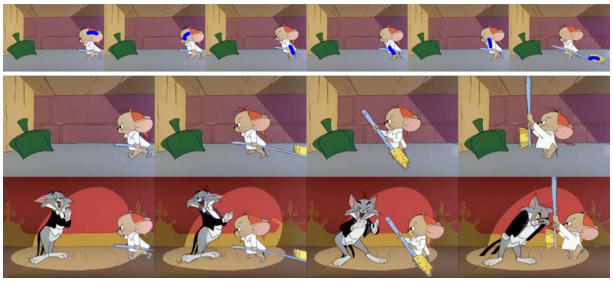

Figure 8: *Direct cut-and-paste of selected video objects. The first row shows the user interactions by Video Brush, and the following two rows correspond to four original frames about Jerry walking and the funny composition results with Tom.*

nal four frames on the second row, and then we simply feature their boundaries by two pixels to composite them with new backgrounds. The final composition results shown in the last row are natural and funny. Figure 9 is a difficult example with complex background. The first row shows three frames of fish in the beautiful seabed, and the second row gives the extracted foreground objects and a composition with a new background. This result shows that our method can deal with some challenging examples.

Selective video stylization Video stylization is an important branch of NPR(Nonphotorealistic rendering) which enables a wide variety of expressive artistic styles such as painting, drawing and abstraction. Selective video stylization refers to stylizing a video while selectively emphasizing information which can attract our attention, and this is an interesting artistic form. We apply the video selection results to making a video which contains a mixture of realism and non-realism. Figure 10 shows two kinds of stylization: abstraction and stroke-based painting. The left two columns are the abstraction results with the gymnast unchanged, and the right two columns show the stroke-based painting while preserving the content of the walking man.

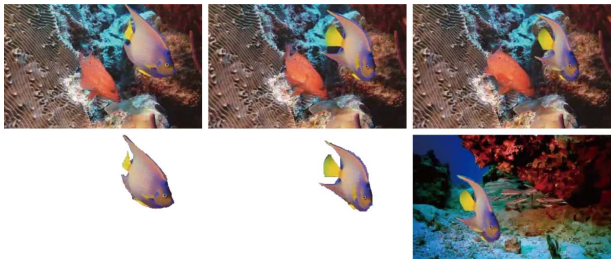

Figure 9: *A difficult example with complex background. The first row is the three original frames with beautiful fish, and the second row shows the extracted foreground objects and composition with a new background.*

Figure 10: *Application of selective video stylization. The top row corresponds to the four original frames to be stylized, and the following row shows two kinds of stylization with the foreground unchanged. The left two images are results obtained by selective abstraction, and the right two images by selective stroke-based painting.*

5. Conclusions

We have presented *Video Brush*, a novel interface to select video objects progressively with quick feedback. With *Video Brush*, users can select video objects progressively in the same way as they do in 2D images. The progressive selection is achieved by solving the graph-cut based local optimization. Given the large number of data in videos, we introduce RPR [DB08] to solve the 3D graph-cut with immense graph. To provide users quick feedback, the 3D graph-cut is accelerated by efficient graph building and 3D banded graph-cut. Experimental results show that the *Video Brush* is helpful for users to extract video objects more efficiently. We believe that the *Video Brush* has a bright future for interactive video cutout.

Our method works well in relative simple videos, however, it fails in some complicated cases. In particular, background and foreground with similar colors, foreground with complex color patterns and fast motion could make our method fail to select objects correctly. Actually, these complex cases are also challenges for previous methods. In addition, our method is still not fast enough to provide instant feedback for users, and the memory cost is also a problem when selecting objects from big videos.

In the future, we would like to further accelerate the algorithm for progressive selection, so as to provide instant feedback with less memory consumption. To make our method robust to the complicated environment, other features such as shape, texture and motion will be considered. Coherent matting techniques are needed to deal with the fuzzy boundaries such as fur and hair, thus the binary segmentations can be further refined. In addition, We believe that the recent progress in image and video saliency detection [RKSH10, CZM*11] could be used to further improve the user interaction and performance of video cutout.

Acknowledgements

We thank all anonymous reviewers for their valuable comments. This work was supported by the National Basic Research Program (No. 2011CB302205) of China and the National Natural Science Foundation (No. 60873126) of China.

References

[Ado10] ADOBE CORPORATION:. http://www.adobe.com, 2010. 1, 2, 3

[AP08] AN X., PELLACINI F.: Appprop: all-pairs appearance-space edit propagation. *ACM Trans. Graph. 27*, 3 (2008). 1

[APB07] ARMSTRONG C. J., PRICE B. L., BARRETT W. A.: Interactive segmentation of image volumes with live surface. *Computers & Graphics 31*, 2 (2007), 212–229. 1

[BFL06] BOYKOV Y., FUNKA-LEA G.: Graph cuts and efficient n-d image segmentation. *International Journal of Computer Vision 70*, 2 (2006), 109–131.

[BJ01] BOYKOV Y., JOLLY M.-P.: Interactive graph cuts for optimal boundary amp; region segmentation of objects in n-d images. In *ICCV* (2001), vol. 1, pp. 105 –112 vol.1. 2

[BK04] BOYKOV Y., KOLMOGOROV V.: An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans. Pattern Anal. Mach. Intell. 26*, 9 (2004), 1124–1137. 2, 3, 4, 5

[BS09] BAI X., SAPIRO G.: Geodesic matting: A framework for fast interactive image and video segmentation and matting. *International Journal of Computer Vision 82*, 2 (2009), 113–132. 3

[BSFG09] BARNES C., SHECHTMAN E., FINKELSTEIN A., GOLDMAN D. B.: Patchmatch: a randomized correspondence algorithm for structural image editing. *ACM Trans. Graph. 28*, 3 (2009). 1

[BWS10] BAI X., WANG J., SAPIRO G.: Dynamic color flow: A motion-adaptive color model for object segmentation in video. In *ECCV* (2010), pp. 617–630. 3

[BWSS09] BAI X., WANG J., SIMONS D., SAPIRO G.: Video snapcut: robust video object cutout using localized classifiers. *ACM Trans. Graph. 28*, 3 (2009), 70:1–11. 1, 3, 5, 7

[CAC*02] CHUANG Y.-Y., AGARWALA A., CURLESS B., SALESIN D., SZELISKI R.: Video matting of complex scenes. *ACM Trans. Graph. 21*, 3 (2002), 243–248. 3

[CCT*09] CHEN T., CHENG M.-M., TAN P., SHAMIR A., HU S.-M.: Sketch2photo: Internet image montage. *ACM Transactions on Graphics 28*, 5 (2009), 124:1–10. 1

[CZM*10] CHENG M.-M., ZHANG F.-L., MITRA N. J., HUANG X., HU S.-M.: Repfinder: finding approximately repeated scene elements for image editing. *ACM Trans. Graph. 29*, 4 (2010). 1

[CZM*11] CHENG M.-M., ZHANG G.-X., MITRA N. J., HUANG X., HU S.-M.: Global contrast based salient region detection. In *IEEE CVPR* (2011), pp. 409–416. 9

[DB08] DELONG A., BOYKOV Y.: A scalable graph-cut algorithm for n-d grids. In *IEEE CVPR* (2008). 2, 4, 8

[EK72] EDMONDS J., KARP R. M.: Theoretical improvements in algorithmic efficiency for network flow problems. *J. ACM 19* (April 1972), 248–264. 4

[FHL*09] FARBMAN Z., HOFFER G., LIPMAN Y., COHEN-OR D., LISCHINSKI D.: Coordinates for instant image cloning. *ACM Trans. Graph. 28*, 3 (2009). 6, 8

[GO10] GASTAL E. S. L., OLIVEIRA M. M.: Shared sampling for real-time alpha matting. *Comput. Graph. Forum 29*, 2 (2010), 575–584. 2

[Gra06] GRADY L.: Random walks for image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell. 28*, 11 (2006), 1768–1783. 2

[GT88] GOLDBERG A. V., TARJAN R. E.: A new approach to the maximum-flow problem. *J. ACM 35* (October 1988), 921–940. 4

[LLW08] LEVIN A., LISCHINSKI D., WEISS Y.: A closed-form solution to natural image matting. *IEEE Trans. Pattern Anal. Mach. Intell. 30*, 2 (2008), 228–242. 2

[LSGX05] LOMBAERT H., SUN Y., GRADY L., XU C.: A multilevel banded graph cuts method for fast image segmentation. In *ICCV* (2005), pp. 259–265. 5

[LSS05] LI Y., SUN J., SHUM H.-Y.: Video object cut and paste. *ACM Trans. Graph. 24*, 3 (2005), 595–600. 1, 2, 3, 5

[LSS09] LIU J., SUN J., SHUM H.-Y.: Paint selection. *ACM Trans. Graph. 28*, 3 (2009), 69:1–7. 1, 2, 3, 4

[RKB04] ROTHER C., KOLMOGOROV V., BLAKE A.: "grabcut": interactive foreground extraction using iterated graph cuts. *ACM Trans. Graph. 23*, 3 (2004), 309–314. 2, 4

[RKSH10] RAHTU E., KANNALA J., SALO M., HEIKKIL "A J.: Segmenting salient objects from images and videos. *ECCV* (2010), 366–379. 9

[SG07] SINOP A. K., GRADY L.: A seeded image segmentation framework unifying graph cuts and random walker which yields a new algorithm. In *ICCV* (2007), pp. 1–8. 2

[SJTS04] SUN J., JIA J., TANG C.-K., SHUM H.-Y.: Poisson matting. *ACM Trans. Graph. 23*, 3 (2004), 315–321. 2

[SM00] SHI J., MALIK J.: Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell. 22*, 8 (2000), 888–905. 2

[WBC*05] WANG J., BHAT P., COLBURN A., AGRAWALA M., COHEN M. F.: Interactive video cutout. *ACM Trans. Graph. 24*, 3 (2005), 585–594. 1, 2, 3, 5

[WC07] WANG J., COHEN M. F.: Image and video matting: A survey. *Foundations and Trends in Computer Graphics and Vision 3*, 2 (2007), 97–175. 2

[XLJ*09] XU K., LI Y., JU T., HU S.-M., LIU T.-Q.: Efficient affinity-based edit propagation using k-d tree. *ACM Trans. Graph. 28*, 5 (2009). 1