Computational Visual Media
DOI 10.1007/s41095-xxx-xxxx-x Vol. x, No. x, month year, xx–xx

Article Type (Research/Review)

Practical automatic background substitution for live video

Haozhi Huang[1], **Xiaonan Fang**[1], **Yufei Ye**[1], **Songhai Zhang**[1] ✉, **Paul L. Rosin**[2]

Abstract In this paper we present a novel automatic background substitution approach for live video. The objective of background substitution is to extract the foreground from the input video and then combine it with a new background. In this paper, we use the color line model to improve the Gaussian mixture model in the Background Cut method to obtain a binary foreground segmentation result that is less sensitive to brightness difference. Based on the high quality binary segmentation results, we can automatically create a reliable trimap for alpha matting to refine the segmentation boundary. To make the composition result more realistic, an automatic foreground color adjustment step is added to make the foreground look consistent with the new background. Compared to previous approaches, our method can produce higher quality binary segmentation results, and to the best of our knowledge, this is the first time such an automatic and integrated background substitution system has been proposed to run in real time, which makes it practical for everyday applications.

Keywords Background substitution, Background replacement, Background subtraction, Alpha Matting.

1 Introduction

Background substitution is a fundamental post-processing technique for image and video editing. It has extensive applications in video composition [2, 8], video conferencing [25, 42] and augmented reality [37]. The process of background substitution can be basically separated into two steps. The first step is to extract the foreground from the input video, and the second step is to combine the original foreground with the new background. Given limited computational resources and time, it is even more challenging when we want to achieve satisfying background substitution results in real time for live video. In this paper, we focus on background substitution for live video and especially live chat video, in which the camera is monocular and static, the background is also basically static.

Foreground segmentation, also known as matting, is a popular fundamental problem in the literature. Formally, foreground segmentation takes as input an image I, which is assumed to be a composite of a foreground image F and a background image B. The color of the ith pixel can be represented as a linear combination of the foreground and background colors, where α represent the opacity value:

$$I_i = \alpha_i F_i + (1 - \alpha_i) B_i. \quad (1)$$

This is an ill-posed problem which needs assumptions or extra constraints to become solvable.

Generally, existing works on foreground segmentation can be categorized into automatic approaches or interactive approaches. Automatic approaches mostly assume that the camera and the background is static, and a pre-captured background image is available. They try to model the background using either generative methods [1, 4, 26, 36], or non-parametric methods [3, 19]. Those pixels which are consistent with the background model will be labeled as background, and the remainder will be labeled as foreground. Some recent works incorporate a conditional random field to include color, contrast and motion cues and use graph-cut to solve an optimization problem [14, 34, 39]. Most of the online automatic approaches only produce a binary

1 Department of Computer Science, Tsinghua University, Beijing, 100084, China. E-mail: huanghz08@gmail.com, wwjpromise@163.com, yeyf13.judy@gmail.com, shz@tsinghua.edu.cn, shimin@tsinghua.edu.cn.

2 School of Computer Science and Informatics, Cardiff University, Cardiff, CF24 3AA, UK. E-mail: rosinpl@cardiff.ac.uk.

Manuscript received: NaN; accepted: NaN.

foreground segmentation instead of fractional opacities for the sake of time, and then use feathering [34] or border matting [14] to compute rough fractional opacities along the boundary. Feathering is a relatively crude, but efficient, technique that fades out the foreground at a fixed rate. Border matting is an alpha matting method that is significantly simplified to only collect the nearby foreground/background samples for each unknown pixel for fitting a Gaussian distribution, which is later used to estimate the alpha value for that pixel. Although border matting also uses dynamic programming to minimize an energy function that encourages alpha values varying smoothly along the boundary, the result of border matting is far from the global optimal. On the other hand, interactive approaches are proposed to handle more complicated camera motion [2, 12, 18]. Since strictly real-time performance is unnecessary for these kinds of applications, they compute more precise fractional opacities along the segmentation boundary from the beginning. These kinds of methods require the user to draw some strokes or a trimap in a few frames to indicate if a pixel belongs to the foreground/background/unknown region. They then solve for the alpha values in the unknown region and propagate the alpha mask to other frames.

In contrast to the large amount of foreground segmentation publications, there are fewer studies on techniques for compositing the original foreground and a new background for background substitution. Since the light sources of the original video and the new background may be drastically different, directly copying the foreground to the new background will not achieve satisfying results. Some seamless image composition techniques [20, 29] may seem relevant at a first glance, but they require the original background and the new background to be similar. Other color correction techniques based on color constancy [6, 10, 11, 16] are more suitable in our context. Color constancy methods first estimate the light source color of the image, and then adjust pixel colors according to the specified hypothetical light source color.

In this paper we present a novel practical automatic background substitution system for live video, especially live chat video. Since real-time performance is necessary and interaction is inappropriate during live chat, our method is designed to be efficient and automatic. We first accomplish binary foreground segmentation by a novel method which is based on Background Cut [34]. To make the segmentation result less sensitive to brightness differences, we introduce a simplified version of the color line model [28] during the background modeling stage. Specifically, we build a color line for each background pixel and allow larger variance along the color line than in the perpendicular direction. We also include a more recent promising alpha matting method [24] to refine the segmentation boundary instead of feathering [34] or border matting [14]. To maintain real-time performance when including such complicated alpha matting process, we do foreground segmentation at a coarser level and then use simple but effective bilinear upsampling to generate a foreground mask for the finer level. After foreground segmentation, in order to compensate for any lighting difference between the input video and the new background, we estimate the color of the light sources in both the input video and new background, and then adjust the foreground color based on the color ratio of the light sources. This color compensation process follows the same idea as the white-patch algorithm [23], but to our knowledge this is the first time this kind of color compensation step has been applied to background substitution. Compared to previous approaches, thanks to its invariance to luminance changes, the binary segmentation result of our method is more accurate, and, thanks to the alpha matting border refinement and foreground color compensation, the appearance of the foreground in our result is more compatible to the new background.

In summary, the main contributions of our paper are:

- A novel practical automatic background substitution system for live video.
- Introduction of the color line model to the Gaussian mixture model at the background modeling stage, which makes the foreground segmentation result less sensitive to brightness differences.
- Application of the color compensation step to background substitution, which makes the inserted foreground look more natural in the new background.

2 Related work

2.1 Automatic Video Matting

Different from interactive video matting methods [2, 12, 18, 41] which need user interaction during the playing of videos, automatic video matting is more appropriate for live video. The earliest kind of automatic video matting problem is constant color matting [32], which uses a constant backing color,

often blue, and its solution is often called blue screen matting. Although excellent segmentation results can be achieved by blue screen matting, it needs extra equipment such as a blue screen and careful setting of light sources. More recent video matting methods loosen the requirement that the background has constant color, and only assume that the background can be pre-captured and remains static or only contains slight movements. They model the background using either generative methods, such as a Bayesian model [1], self-organized map [26], Gaussian mixture model [4], independent component analysis [36], foreground-background mixture model [27] or non-parametric methods [3, 19]. Using these models, we can predict the probability of a pixel belonging to the background. These methods will create holes in the foreground and noise in the background if the colors of the foreground and background are similar, because they only make local decisions. Some recent techniques utilize the power of graph-cut to solve an optimization problem based on a conditional random field using color, contrast and motion cues [14, 34, 39], which are able to create more complete foreground masks since they constrain the alpha matte to follow the original image gradient. There are also some works [40] that focus on foreground segmentation for animation. In our case, in order to acquire real-time online matting for live video, it is inappropriate to include motion cues. Thus our model is only based on color and contrast like Sun et. al [34]. We also find that stronger shadow resistance can be achieved by employing the color line model [28]. Another drawback of existing online methods is that they only acquire a binary foreground segmentation and then use rough border refinement techniques such as feathering [34] or border matting [14] to compute fractional opacities along the boundary. In this paper, we will show that more precise alpha matting technique can be incorporated while real-time performance can still be achieved by doing foreground segmentation at a coarser level and then using simple bilinear upsampling to generate a finer level foreground mask.

2.2 Interactive Video Matting

Interactive video matting is an other popular kind of video matting method. It releases the requirement of known background and static camera, and takes a user drawn trimap or strokes to tell if a pixel belongs to the foreground/background/unknown region. For images, previous methods often use sampling-based methods [17], affinity-based methods [24], or the combination of both [9] to compute alpha values for the unknown region based on the known region information. For videos, Chuang et al [12] use optical flow to propagate the trimap from one frame to another. Video SnapCut [2] maintains a bunch of local classifiers around the object boundary, where each classifier subsequently solves a local binary segmentation problem, and classifiers of one frame will be propagated to next frames according to motion vectors estimated across frames. However, they need to take all frames all at once for computing reliable motion vectors and the time complexity is huge, which makes it unsuitable for online video matting. Gong et al [18] use two competing one-class support vector machines (SVM) to model the background and foreground separately for each frame at every pixel location, use the probability values predicted by the SVMs to solve estimate the alpha matte, and update the SVMs over time. Near real-time performance is available with the help of a GPU, but they still need user input trimap and an extra training stage, which makes it inconvenient for live video application.

There are three main categories of methods for color adjustment to improve the realism of image composites. The first category focuses on color consistency or color harmony. For example, Wong et al [38] adjust foreground colors to be consistent with nearby surrounding background pixels, but their method fails when the nearby background pixels do not correctly represent the overall lighting condition; Cohen-Or et. al [13] and Kuang et. al [22] consider overall color harmony based on either aesthetic rules or models learned from a dataset, but they tend to focus on creating aesthetic images rather than realistic images. The second category of methods focuses on seamless cloning based on solving a Poisson equation or coordinate interpolation [7, 8, 15, 20, 29]. There is a major assumption in these approaches that the original background needs to be similar to the new background, which we cannot guarantee in our application. The third category of methods is color constancy, which estimates the illuminant of the image first and then adjust colors accordingly [6, 10, 11, 16]. In this paper, we choose to utilize the most basic and popular color constancy method, the white-patch algorithm [23], to estimate the light source color, since we need its efficiency for real-time application.

3 Our approach

3.1 Overview

We now outline our method. Its pipeline can be separated into three steps: foreground segmentation, border refinement, and final composition. Firstly, for the foreground segmentation step, we suppose the background can be pre-captured and maintains static. Inspired by Background Cut [34], we build a global Gaussian mixture background model, local single Gaussian background models at all pixel locations, and a global Gaussian mixture foreground model. But unlike Background Cut, instead of using an isotropic variance for the local single Gaussian background models, we make the variance along the color line larger than that in the direction perpendicular to the color line. Here the concept of a color line is borrowed from [28]. The original color line model built multiple curves to represent all colors of the whole image, and assumed that colors from the same object lie on the same curve. To check which curve a pixel belongs to is a time consuming process. In order to achieve real-time performance, we adapt the color line model to a much simpler and more efficient version. In our basic version of the color line model, for each pixel we build a single curve color line model, which avoids the process of matching a pixel to one of the curves in the multiple curves model. Furthermore, instead of fitting a curve, we fit a straight line that intersects the origin in the RGB space, which means we ignore the non-linear transform of the camera sensor. In our experiments, we find this simplified model is sufficient and effective. By utilizing this color line model, we can avoid misclassifying background pixels which suffer color changes due to a shadow passing by, since the color changes caused by the shadow still remain along the color line. Using this Background Cut model, we can build an energy function that can be optimized by graph-cut and get a binary foreground segmentation matte. Secondly, we carry out border refinement for this binary foreground matte. Specifically, we use morphology operations to mark the border pixels between foreground and background. Considering these border pixels as the unknown region, we got a trimap and then carry out closed-form alpha matting [24], which computes fractional alpha values for these border pixels. It is important to emphasize that only when the binary foreground segmentation result is basically correct, is it reliable to automatically generate a trimap in this way. Lastly, for the final composition, we estimate the light source colors of the original input video and the new background separately, and adjust foreground colors accordingly to make the foreground look more consistent with the new background.

3.2 Foreground Segmentation

3.2.1 Basic Background Cut Model

In this section we briefly describe the Background Cut model proposed in [34]. The Background Cut algorithm takes a video and a pre-captured background as the input, and the output is a sequence of binary foreground masks, in which each pixel r is labelled 0 if it belongs to the background or 1 otherwise. Background Cut solves the foreground segmentation problem frame by frame. For each frame, the process of labelling can be transformed into solving a global optimization problem. The energy function to be minimized is in the form of a Conditional Random Field:

$$E(X) = \sum_r E_d(x_r) + \lambda_1 \sum_{r,s} E_c(x_r, x_s), \qquad (2)$$

where $X = \{x_r\}$, x_r denotes the label value, r, s are neighbouring pixels in one frame, E_d represents per-pixel energy (usually called the data term), E_c is a contrast term computed from the neighbouring pixels. Here λ_1 is a predefined constant balancing E_d and E_c, which is empirically set to 30 in our experiment. This is a classic energy function which can be minimized by graph-cut [5].

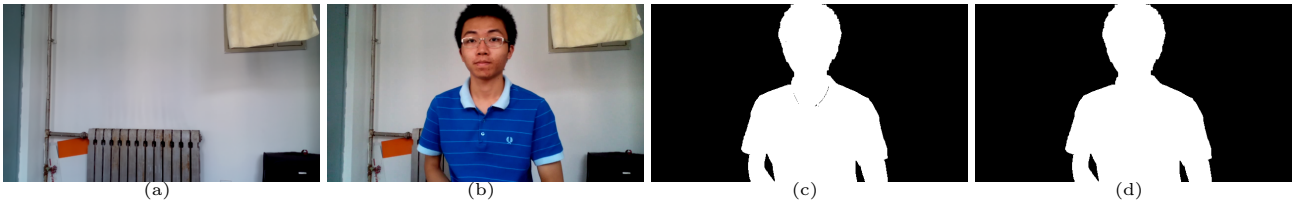
Now we explain how to construct E_d and E_c. First we model the foreground and the background using Gaussian models. For the foreground, we build a global Gaussian mixture model (GMM). For the background, we not only build a global GMM, but also a local single Gaussian distribution model at each pixel location (called a per-pixel model). The two global GMMs are defined as:

$$p(v_r|x_r = i) = \sum_{k=1}^{k_i} w_k^i N(v_r|\mu_k^i, \Sigma_k^i), i = 0, 1, \qquad (3)$$

where $i = 0, 1$ stands for background and foreground respectively, v_r denotes the color of pixel r, k_i denotes the number of mixture components, w_k^i denotes the weight of the kth component, N denotes the Gaussian distribution, μ_k^i denotes the mean, Σ_k^i denotes the covariance matrix. The single Gaussian distribution at every pixel location is defined as:

$$p_s(v_r) = N(v_r|\mu_r^s, \Sigma_r^s), \qquad (4)$$

where $\Sigma_r^s = \sigma_r^s I$, which means, following [34], that the variance of the per-pixel model is isotropic. The background global GMM and the background per-pixel model are initialized using pre-captured background data. The foreground global GMM is initialized using



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

Fig. 1 (a) pre-captured background. (b) one frame of the input video. (c) binary foreground matte after graph-cut. (d) foreground matte after filling holes.

pixels whose probabilities are lower than a threshold in the background model. After initialization, these Gaussian models will be updated frame by frame according to the segmentation result.

Based on the Gaussian models, the data term E_d is defined as:

$$E_d(x_r) = \begin{cases} -\log\left(\lambda_2 p(v_r|x_r) + (1-\lambda_2)p_s(v_r)\right), x_r = 0 \\ -\log p(v_r|x_r), x_r = 1. \end{cases} \quad (5)$$

Here λ_2 is a predefined constant balancing the global GMM and the local per-pixel model, which is empirically set to 0.1 in our experiments. The contrast term is

$$E_c(x_r, x_s) = |x_r - x_s| \exp(-\beta||v_r - v_s||^2 / d_B(r,s)), \quad (6)$$

$$d_B(r,s) = 1 + (||v_r^B - v_s^B||/K)^2 \exp(-z_{rs}^2/\sigma_z), \quad (7)$$

where $d_B(r,s)$ is a contrast attenuation term proportional to the contrast with respect to the background, $z_{rs} = max(||v_r - v_r^B||, ||v_s - v_s^B||)$ measures the dissimilarity between the pre-captured background and the current frame; β, K, σ_z are predefined constants. In our experiment, we set $\beta = 0.005$, $K = 1$, $\sigma_z = 10$. The introduction of the contrast attenuation term makes the calculation of E_c rely on the contrast from the foreground instead of the background.

The energy function Eq. (2) can be optimized using the graph-cut algorithm [5]. For more details of the model, please refer to [34]. One major drawback of this Background Cut model is that, when the color of a background pixel changes due to changes in illumination, it will have extremely low probability in the per-pixel model, which will cause the pixel to be misclassified from background to foreground.

3.2.2 Background Cut with Color Line Model

Now we will show how the color line model [28] can improve the effectiveness of the Background Cut model in the presence of shadows.

Based on the basic color line model, we make the assumption that colors of a certain material under different intensities of light form a linear color cluster that intersects the origin in the RGB space. Suppose the average color at a pixel location is $\mu_r^s = (r, g, b)$. When the illumination of the same pixel location changes, its color will also change from μ_r^s to v_r. According to the color line model, v_r will approximately lie on the line connecting the origin and μ_r^s in the RGB color space. With this insight, we can decompose v_r as

$$v_r = v_\perp + v_\parallel \quad (8)$$

such that $v_\perp \perp \mu_r^s$ and $v_\parallel \parallel \mu_r^s$. Define

$$f(v_r, \mu_r^s) = N\left(\|v_\perp\| \,\middle|\, 0, \sigma_{pe}\right) N\left(\|v_\parallel\| \,\middle|\, \|\mu_r^s\|, \sigma_{pa}\right), \quad (9)$$

where σ_{pe} and σ_{pa} are the respective variances of the Gaussian distributions for the perpendicular direction and parallel direction. Then the per-pixel single Gaussian distribution Eq. (4) is modified as

$$p_s(v_r) = f(v_r, \mu_r^s). \quad (10)$$

As discussed before, the color of an object is more likely to fluctuate in the parallel direction rather than in the perpendicular direction. Therefore, we set $\sigma_{pe} = \sigma_r^s$, $\sigma_{pa} = \lambda_3 \sigma_{pe}$, $\lambda_3 > 1$ to constrain variance in the perpendicular direction and tolerate variance in the parallel direction, which gives our model a strong resistance to shadow. Here we do not build a global color line model as in [28], which has multiple color lines for the whole image to replace the global GMM, because it takes a long time to determine which line each pixel belongs to when the number of lines is large (e.g. a model of 40 lines are used in [28]), and it will hinder the real-time performance.

3.3 Border Refinement

After graph-cut, we add an extra hole filling step by applying the morphology close operation to fill small holes in the foreground mask. See Fig. 1 for an example. However, what we currently have is still a binary foreground matte (Fig. 1d). In this subsection, we explain how to automatically compute fractional alpha values for the segmentation border.

First, we automatically generate a mask covering the segmentation border as the unknown region:

$$U_i = 1 - (\text{erode}(F)_i \; or \; \text{erode}(B)_i). \quad (11)$$

Here U_i denotes the value of the ith pixel of the unknown mask, erode() denotes the morphology erode

Fig. 2 (a) automatically generated trimap. (b) alpha matting result.

operation, F is the binary foreground matte, B is the binary background matte where $B_i = 1 - F_i$. The morphology operation radius is set to 2 for 640 × 480 input. The eroded foreground mask, eroded background mask, unknown region mask are separately painted in white, black and gray in the final trimap. Using this trimap with one of the most popular alpha matting methods [24], we calculate the fractional alpha values for the unknown region. See Fig. 2 for an example of the generated trimap and alpha matting result.

3.4 Final Composition

For an ideal final composition, the new composite image should be:

$$I_{new} = \alpha F_{old} + (1 - \alpha) B_{new}. \quad (12)$$

Here I_{new} denotes the new composite image, F_{old} denotes the original foreground, B_{new} denotes the new background (Fig. 3(c)). For previous methods whose pre-captured background (Fig. 3(a)) is unavailable, F_{old} is approximated by I_{old}:

$$I_{new} = \alpha I_{old} + (1 - \alpha) B_{new}. \quad (13)$$

However, in our case, since the pre-captured background B_{old} is available, we can calculate the original foreground more accurately:

$$F_{old} = (I_{old} - (1 - \alpha) B_{old})/\alpha. \quad (14)$$

So the final composition formula should be:

$$I_{new} = I_{old} + (1 - \alpha)(B_{new} - B_{old}). \quad (15)$$

Directly applying the above composition will create unrealistic results due to the difference of the light source colors between the original input and the new background. Thus, we propose a color compensation process to deal with this problem.

First, we need to estimate the light source colors of the original input video and the new background image. The white-patch method [23], which is a popular color constancy method, assumes that the highest values in each color channel represent the presence of white in the image. In this paper, we use the variant of white-patch method which is designed for CIE-Lab space, a color space that is naturally designed to separate lightness and chroma. We first calculate the accumulated histogram in the lightness channel L of an image in CIE-Lab space, and consider the 10% pixels with the largest lightness values as the white pixels. Fig. 3 shows an example of the light source masks. The estimated light source color is then computed as the mean color value of all light source pixels. Denote the estimated light source color of the input video as c_{old}, that of the new background image as c_{new}, the new composite image after color compensation is:

$$I_{new} = r I_{old} + (1 - \alpha)(B_{new} - r B_{old}), \quad (16)$$

$$r = c_{new}/c_{old}. \quad (17)$$

Fig. 4 shows a comparison between results with and without light source color compensation. We can clearly see that the result with color compensation is more realistic.

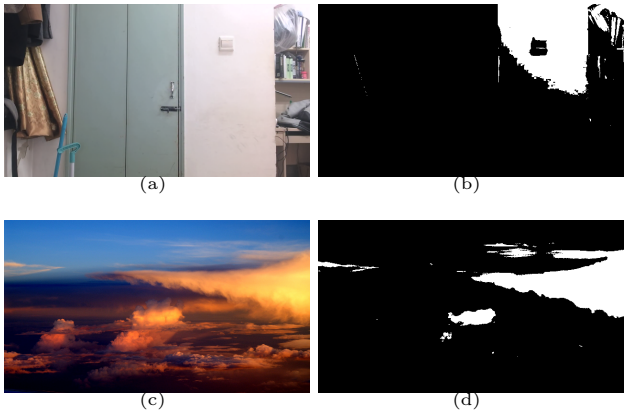


Fig. 3 (a) pre-captured background. (b) estimated light source mask of the pre-captured background(a). (c) new background. (d) estimated light source mask of the new background(c)

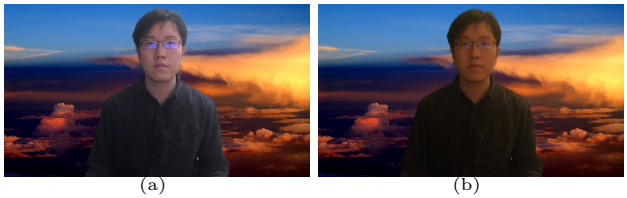


Fig. 4 (a) composite result without color compensation. (b) composite result with color compensation.

4 Results and discussion

In this section, we report results generated under different conditions. All results of our method are generated using fixed parameters.

Results for different frames of the same input video. Fig. 5 shows that our method can create generally good background substitution results for different frames, no matter what the gesture is. Sometimes there may be residual background between the fingers (e.g. Fig. 5c) due to the holes filling post-processing, but it does not do much harm to the overall

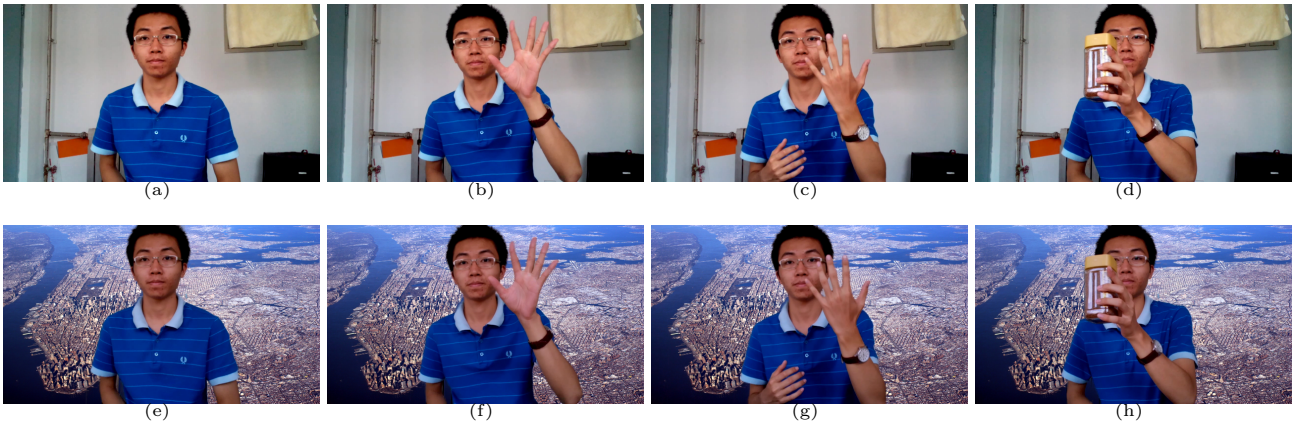


Fig. 5 (a-d) input video frames. (e-h) background substitution results.

effect.

Results for different input video. Fig. 6 shows that our method can deal with different kinds of foreground and background. Color compensation works fine for various lighting condition. Although the matting border is not 100% perfect for Fig 6b due to mixing up of hair and the background, the composition result is generally good.

Comparison with previous methods. We compare Fuzzy Gaussian [31], Adaptive-SOM [26], Background Cut [34] using RGB color space, CIE-Lab color space and our color line model. For the implementation of Fuzzy Gaussian and Adaptive-SOM, we use the code in BGSLibrary [33]. There are also other background subtraction methods shown in BGSLibrary, we choose these two methods because they show the most promising results under real time conditions. Fig. 7 shows foreground masks created by different methods. After the person walks into the picture, some shadow will be cast onto the wall. Fuzzy Gaussian and Adaptive-SOM creates lots of noise and holes since they have not utilized the gradient information between neighbouring pixels. Background Cut in RGB color space does a better job by using the graph-cut model to introduce gradient information. However, it is sensitive to brightness difference, which causes shadow to be misclassified as foreground. If we set the variance of the Gaussian to be larger to tolerate some shadow, part of the true foreground will be misclassified as background. Background Cut in CIE-Lab space also suffers from the same issue. Although allowing a larger variance in the L channel can also give greater tolerance to brightness changes, in actual test cases, even when we only increase the variance in the L channel by a small amount, part of the collar will disappear. In contrast, using our color line model with the Background Cut can constantly create a better foreground segmentation result.

To further quantitatively evaluate the comparison, we create a large number of 'ground truth' foreground masks following a similar idea to [30]. The key idea is to use some balls as the moving foreground objects, and use a circle detection technique to detect the balls, which will automatically create 'ground truth' masks for evaluation of our foreground segmentation methods. Specifically, we first calculate the difference image between the pre-captured background and the current frame (where one or more balls appear). Then we perform circle detection using the Hough Transform [21] on the difference image, which generally produces reliable and accurate detection results. Finally, we manually eliminate the small number of outliers that occur when the circle detection fails. In total, 4105 frames and their circle detection results are collected as the 'ground truth'. Fig. 9 shows a few examples. We did not use the 'ground truth' from the VideoMatting benchmark [35], because their synthesized test images do not have shadows cast on the background, which is one of the fundamental aspects we wish to test. Using the ground truth we generated, we test different methods including Fuzzy Gaussian, Adaptive-SOM, Background Cut using RGB color space, CIE-Lab color space and our color line model. For Fuzzy Gaussian and Adaptive-SOM, we use the default parameters provided by BGSLibrary. For the Background Cut method using different color spaces, we test several parameters and show those with the highest F1 score. From Table 1, we can see that Background Cut with our color line model acquires the highest F1 score, CIE-Lab space follows closely, others are substantially worse. However, as we have already shown in Fig. 7, CIE-Lab space shows an obvious drawback in actual application scenarios. We also test an outdoor scene with different methods to show the

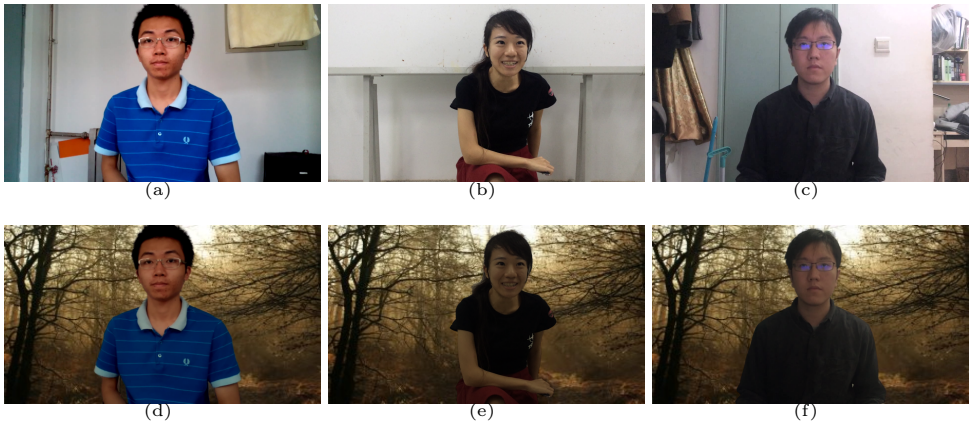


Fig. 6 (a-c) input frames from different videos. (d-f) background substitution results.

Tab. 1 Methods Comparison on Ground Truth Dataset

Method	Precision	Recall	F1
Fuzzy Gaussian	0.252	0.993	0.402
Adaptive-SOM	0.510	0.963	0.667
BC-RGB	0.839	0.962	0.896
BC-Lab	0.900	0.968	0.933
BC-Colorline	0.907	0.964	0.935

effectiveness of our model in Fig. 8. In conclusion, our color line model generally creates a better foreground segmentation boundary, and is effective at coping with differences in brightness.

Results with different new background. We also test our color compensation method under new backgrounds with different light sources. In Fig. 10 the first row shows the new input backgrounds, and the second row shows the light source pixel masks. The third row contains the composition results; we can see that the color of the foreground varies correctly according to different backgrounds.

Acceleration. Although we restrict the alpha matting computation to a very small unknown region, it is still computationally expensive. In order to make our algorithm run at real time, we first downsample the input frame by a scale of two, carry out foreground cut and alpha matting on the downsampled images, and then upsample the matting result to the original scale. We finish the final composition step at the original scale. Let us call this process 'sampling acceleration'. As we can see in Fig. 11, the matting result with sampling acceleration is very similar to the original one. If we do not used alpha matting to refine the border, apparent jags will appear along the border (Fig. 11c).

Performance. We have implemented our method in C++ on a PC with an Intel 3.4GHz Core i7-3770 CPU. For a 640×480 input video, our background substitution program can run at 10 frames per second using just the CPU, and it can run at a real-time frame rate with GPU parallelization.

5 Conclusions

In this paper, we have presented a novel background substitution method for live video. It optimizes a cost function based on Gaussian mixture models and Conditional Random Field by graph-cut. The color line model is introduced when computing the Gaussian mixture model to make the model less sensitive to brightness differences. Before final composition, we use alpha matting to refine the segmentation border. Light source colors of the input video and new background are estimated by a proposed simple method, and we adjust the foreground colors accordingly to give more realistic composition results. Compared to previous methods, our approach can automatically produce more accurate foreground segmentation masks and more realistic composition results, while still maintaining real-time performance.

Acknowledgements

We thank the reviewers for their valuable comments. This work was supported by the National High Technology Research and Development Program of China (Project Number 2012AA011903), the National Natural Science Foundation of China (Project Number 61373069), and Research Grant of Beijing Higher Institution Engineering Research Center,and Tsinghua-Tencent Joint Laboratory for Internet Innovation Technology.

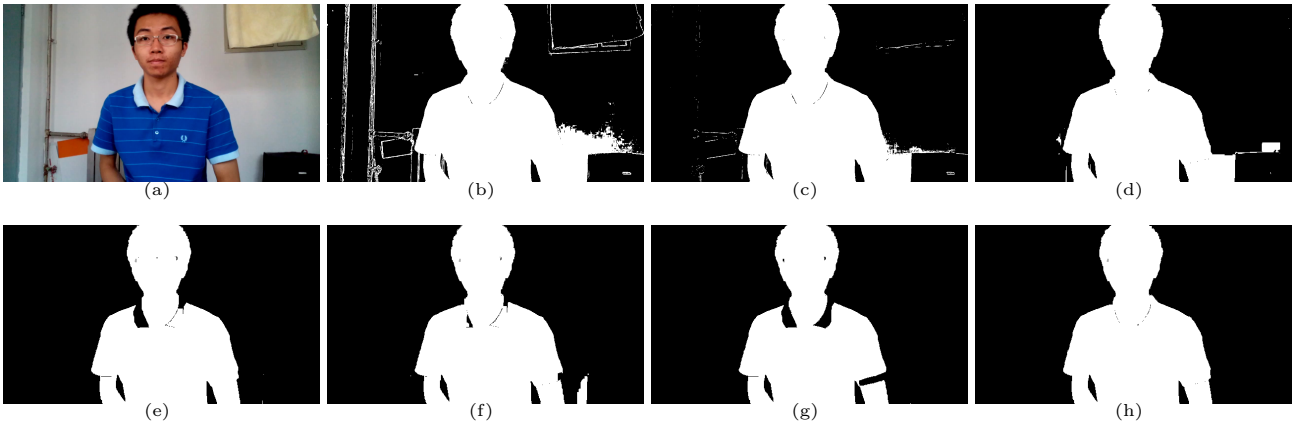


Fig. 7 (a) input frame. (b) foreground mask created by Fuzzy Gaussian. (c) result created by Adaptive SOM. (d) result created by Background Cut in RGB space with a smaller variance ($\sigma_r^s = (5/255)^2$) of the Gaussian model. (e) result created by Background Cut in RGB space with a larger variance ($\sigma_r^s = (20/255)^2$). (f) result created by Background Cut in CIE-Lab space with $\sigma_L = \sigma_a = \sigma_b = (5/255)^2$. (g) result created by Background Cut in CIE-Lab space with a larger variance in L channel ($\sigma_L = 5 * (5/255)^2$). (h) result of our method ($\sigma_{pe} = (10/255)^2, \sigma_{pa} = 10 * (10/255)^2$).

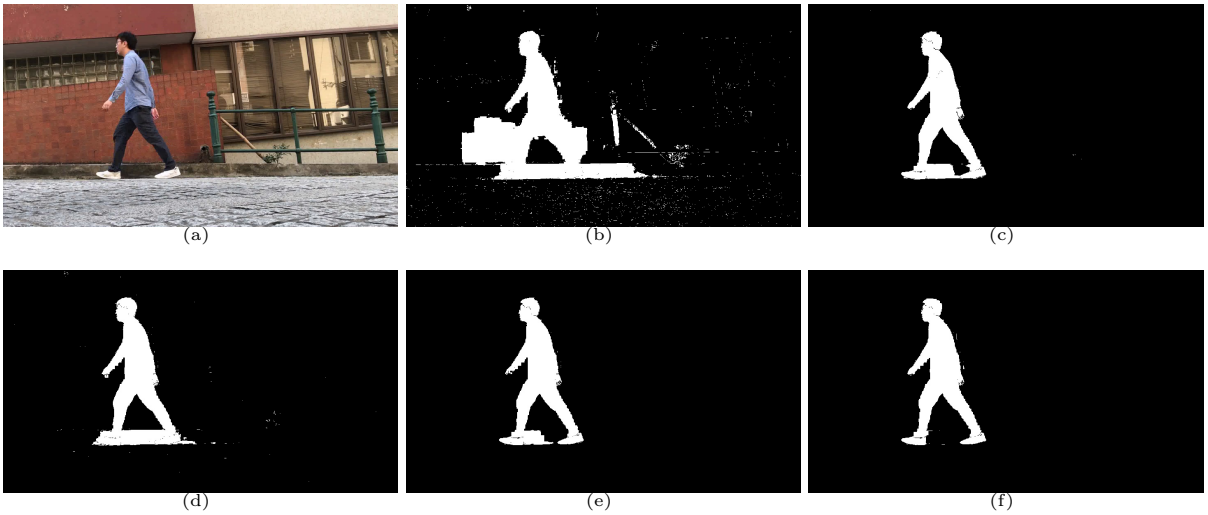


Fig. 8 (a) input frame. (b) result created by Background Cut in RGB space with a smaller variance ($\sigma_r^s = (5/255)^2$) of the Gaussian model. (c) result created by Background Cut in RGB space with a larger variance ($\sigma_r^s = (20/255)^2$). (d) result created by Background Cut in CIE-Lab space with $\sigma_L = \sigma_a = \sigma_b = (5/255)^2$. (e) result created by Background Cut in CIE-Lab space with a larger variance in L channel ($\sigma_L = 5 * (5/255)^2$). (f) result of our method ($\sigma_{pe} = (10/255)^2, \sigma_{pa} = 10 * (10/255)^2$).

Fig. 9 Example frames for creating ground truth.

References

[1] N. Apostoloff and A. Fitzgibbon. Bayesian video matting using learnt image priors. In *Conference on Computer Vision and Pattern Recognition*, volume 1, pages I–407. IEEE, 2004.

[2] X. Bai, J. Wang, D. Simons, and G. Sapiro. Video snapcut: robust video object cutout using localized classifiers. *ACM Transactions on Graphics (TOG)*, 28(3):70, 2009.

[3] O. Barnich and M. Van Droogenbroeck. ViBe: A universal background subtraction algorithm for video sequences. *IEEE Transactions on Image Processing*,

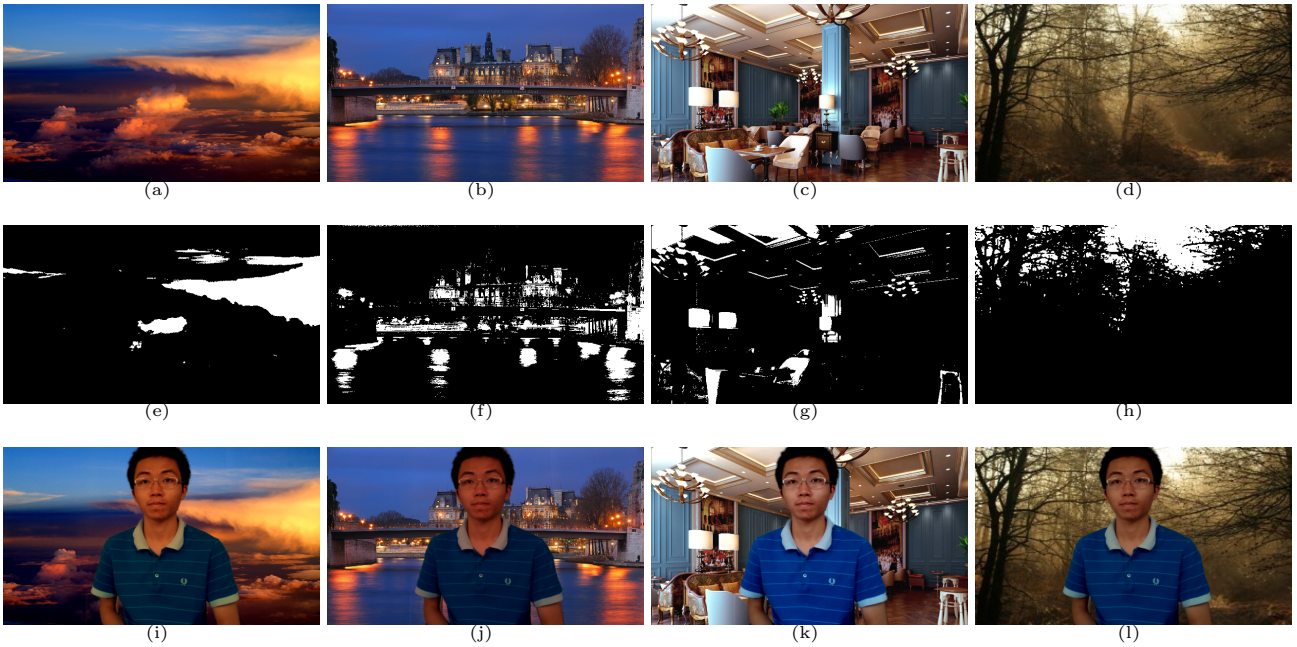


Fig. 10 (a-d) input new backgrounds. (e-h) estimated light source pixel mask. (i-l) background substitution results.

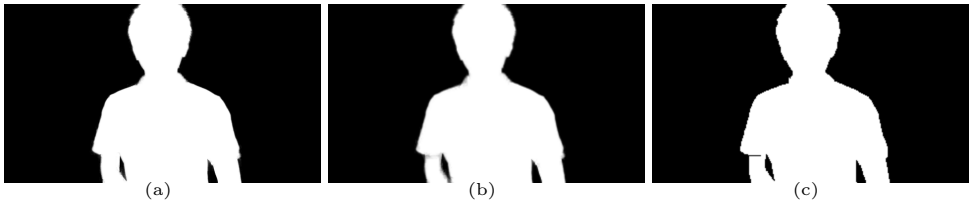


Fig. 11 (a) matting result without sampling acceleration. (b) matting result with sampling acceleration. (c) foreground segmentation result with sampling acceleration but without alpha matting border refinement.

20(6):1709–1724, 2011.

[4] T. Bouwmans, F. El Baf, and B. Vachon. Background modeling using mixture of gaussians for foreground detection-a survey. *Recent Patents on Computer Science*, 1(3):219–237, 2008.

[5] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. In *Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 359–374. Springer, 2001.

[6] G. Buchsbaum. A spatial processor model for object colour perception. *Journal of the Franklin Institute*, 310(1):1–26, 1980.

[7] T. Chen, M.-M. Cheng, P. Tan, A. Shamir, and S.-M. Hu. Sketch2photo: internet image montage. *ACM Transactions on Graphics (TOG)*, 28(5):124, 2009.

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

[9] X. Chen, D. Zou, S. Zhou, Q. Zhao, and P. Tan. Image matting with local and nonlocal smooth priors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1902–1907, 2013.

[10] D. Cheng, D. K. Prasad, and M. S. Brown. Illuminant estimation for color constancy: why spatial-domain methods work and the role of the color distribution. *JOSA A*, 31(5):1049–1058, 2014.

[11] D. Cheng, B. Price, S. Cohen, and M. S. Brown. Beyond white: Ground truth colors for color constancy correction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 298–306, 2015.

[12] Y.-Y. Chuang, A. Agarwala, B. Curless, D. H. Salesin, and R. Szeliski. Video matting of complex scenes. In *ACM Transactions on Graphics (TOG)*, volume 21, pages 243–248. ACM, 2002.

[13] D. Cohen-Or, O. Sorkine, R. Gal, T. Leyvand, and Y.-Q. Xu. Color harmonization. In *ACM Transactions on Graphics (TOG)*, volume 25, pages 624–630. ACM, 2006.

[14] A. Criminisi, G. Cross, A. Blake, and V. Kolmogorov. Bilayer segmentation of live video. In *Conference on Computer Vision and Pattern Recognition*, volume 1, pages 53–60. IEEE, 2006.

[15] Z. Farbman, G. Hoffer, Y. Lipman, D. Cohen-Or, and D. Lischinski. Coordinates for instant image cloning. In *ACM Transactions on Graphics (TOG)*, volume 28, page 67. ACM, 2009.

[16] G. D. Finlayson, S. D. Hordley, and P. M. Hubel. Color

by correlation: A simple, unifying framework for color constancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1209–1221, 2001.

[17] E. S. Gastal and M. M. Oliveira. Shared sampling for real-time alpha matting. In *Computer Graphics Forum*, volume 29, pages 575–584. Wiley Online Library, 2010.

[18] M. Gong, Y. Qian, and L. Cheng. Integrated foreground segmentation and boundary matting for live videos. *IEEE Transactions on Image Processing*, 24(4):1356–1370, 2015.

[19] M. Hofmann, P. Tiefenbacher, and G. Rigoll. Background segmentation with feedback: The pixel-based adaptive segmenter. In *Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 38–43. IEEE, 2012.

[20] J. Jia, J. Sun, C.-K. Tang, and H.-Y. Shum. Drag-and-drop pasting. *ACM Transactions on Graphics (TOG)*, 25(3):631–637, 2006.

[21] D. Kerbyson and T. Atherton. Circle detection using hough transform filters. In *Proceeding of Image Processing and its Applications*, pages 370–374. IET, 1995.

[22] Z. Kuang, P. Lu, X. Wang, and X. Lu. Learning self-adaptive color harmony model for aesthetic quality classification. In *Sixth International Conference on Graphic and Image Processing (ICGIP)*, pages 94431O–94431O. International Society for Optics and Photonics, 2015.

[23] E. H. Land and J. J. McCann. Lightness and retinex theory. *JOSA*, 61(1):1–11, 1971.

[24] A. Levin, D. Lischinski, and Y. Weiss. A closed-form solution to natural image matting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):228–242, 2008.

[25] Z. Liu and M. Cohen. Head-size equalization for better visual perception of video conferencing. In *IEEE International Conference on Multimedia and Expo*, pages 4–pp. IEEE, 2005.

[26] L. Maddalena and A. Petrosino. A self-organizing approach to background subtraction for visual surveillance applications. *IEEE Transactions on Image Processing*, 17(7):1168–1177, 2008.

[27] A. Mumtaz, W. Zhang, and A. B. Chan. Joint motion segmentation and background estimation in dynamic scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 368–375, 2014.

[28] I. Omer and M. Werman. Color lines: Image specific color representation. In *Proceedings of the Computer Vision and Pattern Recognition*, volume 2, pages II–946. IEEE, 2004.

[29] P. Pérez, M. Gangnet, and A. Blake. Poisson image editing. In *ACM Transactions on Graphics (TOG)*, volume 22, pages 313–318. ACM, 2003.

[30] P. L. Rosin and E. Ioannidis. Evaluation of global image thresholding for change detection. *Pattern Recognition Letters*, 24(14):2345–2356, 2003.

[31] M. H. Sigari, N. Mozayani, and H. Pourreza. Fuzzy running average and fuzzy background subtraction: concepts and application. *International Journal of Computer Science and Network Security*, 8(2):138–143, 2008.

[32] A. R. Smith and J. F. Blinn. Blue screen matting. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, pages 259–268. ACM, 1996.

[33] A. Sobral. Bgslibrary, 2016.

[34] J. Sun, W. Zhang, X. Tang, and H.-Y. Shum. Background cut. In *European Conference on Computer Vision*, pages 628–641. Springer, 2006.

[35] Graphics and Media Lab. Videomatting benchmark, 2016.

[36] D.-M. Tsai and S.-C. Lai. Independent component analysis-based background subtraction for indoor surveillance. *IEEE Transactions on Image Processing*, 18(1):158–167, 2009.

[37] D. Van Krevelen and R. Poelman. A survey of augmented reality technologies, applications and limitations. *International Journal of Virtual Reality*, 9(2):1, 2010.

[38] B.-Y. Wong, K.-T. Shih, C.-K. Liang, and H. H. Chen. Single image realism assessment and recoloring by color compatibility. *IEEE Transactions on Multimedia*, 14(3):760–769, 2012.

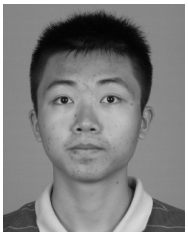
[39] P. Yin, A. Criminisi, J. Winn, and I. Essa. Bilayer segmentation of webcam videos using tree-based classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):30–42, 2011.

[40] L. Zhang, H. Huang, and H. Fu. Excol: an extract-and-complete layering approach to cartoon animation reusing. *IEEE Transactions on Visualization and Computer Graphics*, 18(7):1156–1169, 2012.

[41] Y. Zhang, Y.-L. Tang, and K.-L. Cheng. Efficient video cutout by paint selection. *Journal of Computer Science and Technology*, 30(3):467–477, 2015.

[42] Z. Zhu, R. R. Martin, R. Pepperell, and A. Burleigh. 3d modeling and motion parallax for improved videoconferencing. *Computational Visual Media*, 2(2):131–142, 2016.

Haozhi Huang Hao-Zhi Huang is currently a Ph. D. student in the Department of Computer Science and Technology, Tsinghua University. He received a Bachelor's degree from Tsinghua University, Beijing, China, in 2012. His research interests include image and video editing, and computer graphics.



Xiaonan Fang Xiaonan Fang is currently an undergraduate student in Department of Computer Science and Technology, Tsinghua University, China. His research interests include computational geometry, image processing and video processing.

Yufei Ye Yufei Ye is currently an undergraduate student in the Department of Computer Science and Technology, Tsinghua University, China. Her research interests lie in video processing, object detection, generative model, unsupervised learning and representation learning.

Songhai Zhang Song-Hai Zhang received his Ph.D. degree in 2007 from Tsinghua University. He is currently an associate professor in the Department of Computer Science and Technology of Tsinghua University, Beijing, China. His research interests include image and video processing, geometric computing.

Paul Rosin Paul Rosin is Professor at the School of Computer Science & Informatics, Cardiff University. Previous posts include lecturer at the Department of Information Systems and Computing, Brunel University London, UK, research scientist at the Institute for Remote Sensing Applications, Joint Research Centre, Ispra, Italy, and lecturer at Curtin University of Technology, Perth, Australia.

His research interests include the representation, segmentation, and grouping of curves, knowledge-based vision systems, early image representations, low level image processing, machine vision approaches to remote sensing, methods for evaluation of approximation algorithms, etc., medical and biological image analysis, mesh processing, non-photorealistic rendering and the analysis of shape in art and architecture.