Robust Bilayer Segmentation and Motion/Depth Estimation with a Handheld Camera

Guofeng Zhang, *Member*, *IEEE*, Jiaya Jia, *Senior Member*, *IEEE*, Wei Hua, and Hujun Bao

Abstract—Extracting high-quality dynamic foreground layers from a video sequence is a challenging problem due to the coupling of color, motion, and occlusion. Many approaches assume that the background scene is static or undergoes the planar perspective transformation. In this paper, we relax these restrictions and present a comprehensive system for accurately computing object motion, layer, and depth information. A novel algorithm that combines different clues to extract the foreground layer is proposed, where a voting-like scheme robust to outliers is employed in optimization. The system is capable of handling difficult examples in which the background is nonplanar and the camera freely moves during video capturing. Our work finds several applications, such as high-quality view interpolation and video editing.

Index Terms—Bilayer segmentation, depth recovery, motion estimation, video editing.

1 Introduction

The prevalence of Internet video facilitates the development of video editing techniques [4], [44]. Layer extraction is one of the vital tools that allow users to separate foreground and background images in a video. Its importance lies in the ability to produce layers whereby users can easily create special effects, such as inserting the foreground object into a virtual environment, and to accomplish necessary adjustment, including removing unwanted objects or deforming them.

High-quality layer separation from a video is a very challenging problem because tightly coupled color, depth, and motion give rise to a large number of variables and significant ambiguity in computation. Previous methods made various assumptions on the background scene or camera motion to simplify the problem. For example, bilayer segmentation methods [14], [20], [35], [43] assume that the camera is fixed and/or the background color distribution is not complex. It is, however, very common that a captured video does not meet these requirements. So, methods that can relax these conditions are in demand.

Background modeling is an important step for bilayer segmentation. If the background image is known,[1] the foreground estimate can be obtained with the color and contrast information [33], [35]. Otherwise, both layers have uncertain pixel assignments, making accurately identifying them challenging. The latter scenario commonly arises when using a handheld camera.

In this paper, we tackle the layer segmentation problem with the input of only a video sequence taken by a freely moving camera. Our objective is the high-quality dynamic foreground extraction, which requires that the computed layers have accurate and temporally consistent boundary in multiple frames. In addition, dense motion fields and depth maps need to be solved for. To accomplish these goals, our method uses several new measures and contributes an iterative optimization scheme to refine the depth and motion estimates.

In the bilayer segmentation step, depth and motion are used to handle layer occlusion and resolve color similarity. Unlike traditional solutions that weight different terms in an objective function, we employ a simple voting-like strategy to effectively balance the set of terms and automatically reject occasional outliers. The bilayer segmentation result is used to refine the optical flow field on the dynamic foreground, avoiding the errors caused by connecting a foreground pixel with a background one.

One example is shown in Fig. 1, which illustrates the input and output of our system. In this example, the background is nonplanar, the color distribution is complex, and the camera moves. All of them make bilayer segmentation challenging to solve. Our system can successfully accomplish foreground extraction, dense motion field construction, and background depth map estimation. Results are shown in Figs. 1d, 1e, and 1f.

A preliminary version of the work appeared in [46]. In this paper, we significantly enhance the system reliability. Major improvements also include 1) incorporating the shape matching cost and image segmentation into optical flow estimation, 2) using the multiview stereo method for more effective depth/motion estimation, and 3) a new method that combines depth, color, and motion to define the overall data cost. In addition, we apply our method to a group of applications, including video composition and

1. A static or rotating camera enables background construction by the clean plate or image mosaicing techniques [7], [36].

- *G. Zhang, W. Hua, and H. Bao are with the State Key Lab of CAD&CG, Zhejiang University, Zijingang Campus, Hangzhou 310058, P.R. China. E-mail: {zhangguofeng, huawei, bao}@cad.zju.edu.cn.*
- *J. Jia is with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong. E-mail: leojia@cse.cuhk.edu.hk.*

Manuscript received 8 Sept. 2009; revised 22 Feb. 2010; accepted 1 May 2010; published online 1 June 2010.
Recommended for acceptance by C. Stewart.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-2009-09-0599.
Digital Object Identifier no. 10.1109/TPAMI.2010.115.

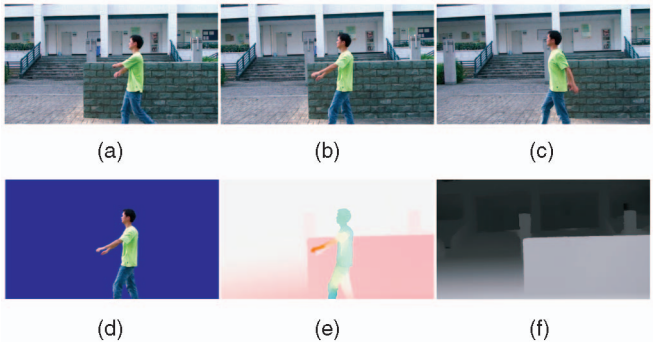


Fig. 1. Foreground extraction, motion estimation, and background reconstruction achieved in our system. (a)-(c) Three selected frames from a sequence. (d) Extracted foreground of (b). (e) Computed optical flow field. (f) Computed background depth map.

view interpolation, to demonstrate its effectiveness at solving different video editing problems.

Assumptions. Similarly to other bilayer segmentation methods, we assume that only the foreground object is dynamic or moving. Our method has a quick manual preprocessing step using GrabCut [32] to coarsely indicate the foreground layer from a single frame for color distribution estimation. Our method is capable of dealing with background with significant depth variation. In addition, our video can be taken by a freely moving camera with translation, rotation, or even forward/backward motion. Panning/static camera and occasionally stationary foreground can be handled thanks to the employment of a few effective measures.

Another assumption is that an abrupt and significant lighting variation does not frequently arise in successive frames. Otherwise, the basic photoconsistency constraint in multiview geometry would be constantly unusable.

2 Related Work

2.1 Bilayer Segmentation

Several bilayer segmentation methods [14], [16], [35] have been proposed assuming that the camera is mostly stationary and the background is known or can be easily modeled. Kolmogorov et al. [20] made use of stereo videos. Object color, gradient, and displacement are integrated to infer the foreground layer in real time. The approaches presented in [14], [35] estimate the foreground layer from a Web camera video using different spatial and temporal priors. In [43], Yin et al. computed two layers monocularly even in the presence of distracting background motion. All of these methods do not handle the scenarios in which the camera undergoes arbitrary motion and the background geometry is complex.

For high-quality foreground extraction, several interactive image/video segmentation/matting techniques [11], [12], [23], [24], [25], [32], [37], [38], [39] were developed. Most of them only use the color/contrast information or require special camera configuration. Frequent user interaction is generally needed for challenging video examples. Recently, Bai et al. [2] proposed a robust interactive video object cutout system. The motion estimate, instead of being incorporated into energy minimization to help reduce the segmentation ambiguity, is used to propagate a set of local classifiers across frames. Liu and Gleicher [27] proposed incorporating learned color and locality cues in an MRF framework. In this method, the motion information is estimated independently without considering layers. In [15], Dong et al. proposed a fast bilayer segmentation method that can effectively extract the dynamic foreground layer. However, it is limited to rotational camera motion.

TABLE 1
Overview of Our Method

1. **Structure from Motion**:
   1.1 Recover the camera parameter set $\mathbf{C}$.
2. **Dense Depth and Motion Estimation**:
   2.1 Use the multi-view stereo technique [45] to recover the depth map for each frame.
   2.2 Estimate dense motion field $\mathbf{d}$ and occlusion map o for each two consecutive frames.
   2.3 Generate dense motion tracks.
3. **Bilayer Segmentation**:
   3.1 Compute the data cost with a few measures.
   3.2 Compute the spatial and temporal smoothness terms.
   3.3 Solve for α by minimizing Eq. (21).
4. Repeat steps 2 and 3 for two iterations.
5. Complete the background layer, and further refine the depth and motion results.
6. Refine the binary segmentation result by matting.

2.2 Optical Flow and Motion Segmentation

As our method also estimates motion, we briefly review related work. The energy minimization framework was originally proposed by Horn and Schunck [18]. It was typically solved with a coarse-to-fine strategy [29]. There have been quite a number of methods to improve the robustness, accuracy, and efficiency [5], [9], [10], [22], [40], [42]. Nevertheless, obtaining high-quality optical flow fields in the presence of large displacement and occlusion is still difficult [8]. Boundary-accurate segmentation only based on the optical flow field is almost impossible.

Motion segmentation [1], [19], [21], [41] aims to coarsely group pixels that undergo similar motion and separate them into multiple layers. These methods cannot accomplish high-quality foreground extraction and usually yield imprecise object boundaries, especially when occlusion or disocclusion happens.

3 System Overview

Given a video sequence with n frames, our objective is to estimate the bilayer, motion, and depth information for each pixel. We denote by $I^t(\mathbf{x})$ the color of pixel $\mathbf{x}$ in frame t. $\alpha^t_{\mathbf{x}}$ (also denoted as $\alpha^t(\mathbf{x})$) has a binary value, representing the layer label for pixel $\mathbf{x}$ in frame t. The pixel belonging to the dynamic foreground makes $\alpha^t_{\mathbf{x}} = 1$; otherwise, $\alpha^t_{\mathbf{x}} = 0$. We set $\alpha = 0$ for all pixels initially. Denoting by $z^t_{\mathbf{x}}$ the depth value of pixel $\mathbf{x}$ in frame t, the disparity $D^t(\mathbf{x})$ is defined as $D^t(\mathbf{x}) = 1/z^t_{\mathbf{x}}$ by convention. $\mathbf{d}^{i,j}(\mathbf{x})$ denotes the motion vector of pixel $\mathbf{x}$ from frame i to j.

Our method iterates between two main phases, i.e., the dense motion and depth estimation and bilayer segmentation phases, for two passes. Table 1 gives an overview of our method.

Fig. 2. Depth recovery with dynamic objects. (a) Selected frames from an input video. (b) Recovered depth maps. Although the depth estimates for the dynamic foreground are problematic, they do not much affect the background depth estimation.

We start by using the structure from motion (SFM) method of Zhang et al. [47] to recover the camera motion parameters from the input video sequence. We apply the SIFT algorithm [28], which produces a set of feature tracks across frames, to estimate the camera poses. The feature tracks on the dynamic foreground can automatically be rejected according to the multiview geometry constraint [17], leaving only background tracks for the camera motion estimation. The output of SFM includes the recovered camera parameter set $\mathbf{C}$ for all frames, and the 3D positions of the background feature tracks. We denote by $\mathbf{C}^t = \{\mathbf{K}^t, \mathbf{R}^t, \mathbf{T}^t\}$ the camera parameters for frame t, where $\mathbf{K}^t$ is the intrinsic matrix, $\mathbf{R}^t$ is the rotation matrix, and $\mathbf{T}^t$ is the translation vector.

With the recovered camera poses, we employ the multiview stereo method of Zhang et al. [45] to infer the view-dependent dense depth maps for all frames. It is notable that the dynamic foreground object does not satisfy the multiview geometry constraint and thus possibly does not receive correct depth estimates, as demonstrated in Fig. 2. Contrarily, the background depth can be accurately computed even without masking the foreground object out. This is because the moving object is a type of "noise" in the depth estimation, and can be effectively handled together with image noise, occlusions, and estimation outliers. The computed dense depth maps will be used in the latter bilayer segmentation process.

4 Dense Motion Estimation

We estimate motion for all pixels (Step 2 in Table 1) to facilitate the following layer separation. The main difference between our method and traditional optical flow estimation is that our method uses the layer information to effectively handle occlusion and textureless regions. Note that in [26], the layers need to be manually labeled beforehand, while in our method the layer parameters can be automatically computed and updated. We use the in-plane displacement vector $\mathbf{d}^{t,t+1}(\mathbf{x})$ (or $\mathbf{d}^{t+1,t}(\mathbf{x})$) to denote the motion of pixel $\mathbf{x}$ from frame t to $t+1$ (or from $t+1$ to t). $o(\mathbf{x}) \in \{0,1\}$ labels occlusion. If the pixel $\mathbf{x}$ is occluded when mapping from frame t to $t+1$, $o^{t,t+1}(\mathbf{x}) = 1$; otherwise $o^{t,t+1}(\mathbf{x}) = 0$.

4.1 Initialization and Notations

To begin with, we partition each frame using the mean-shift color segmentation method [13] and then perform the following operations: 1) If a segment contains both foreground and background pixels (based on the current α estimate), it further splits. 2) A very small segment (with less than 10 pixels) will be merged to a neighboring segment with the most similar mean color. When these operations are done, we represent the motion of a segment between two images using a planar transformation. Let l_i^t denote the ith segment in frame t. For the pixel $\mathbf{x} \in l_i^t$, its displacement vector $\mathbf{d}^{t,t+1}(\mathbf{x})$ is written as

$$\mathbf{d}^{t,t+1}(\mathbf{x}) = A_{l_i^t}^{t,t+1} \lambda H^{t,t+1} \hat{\mathbf{x}} - \mathbf{x}, \tag{1}$$

where $\hat{\mathbf{x}}$ is the homogenous coordinate of $\mathbf{x}$ and λ is the scaling factor, i.e., the inverse of the third coordinate of $H^{t,t+1}\hat{\mathbf{x}}$. $H^{t,t+1}$ is a 3×3 matrix, which represents the global image transformation caused by camera rotation or varying focal length. With the estimated camera parameters $\mathbf{C}$, $H^{t,t+1}$ is written as

$$H^{t,t+1} = \mathbf{K}^{t+1}\mathbf{R}^{t+1}(\mathbf{R}^t)^\top(\mathbf{K}^t)^{-1}. \tag{2}$$

$\lambda H^{t,t+1}\hat{\mathbf{x}}$ is the rectified position of $\mathbf{x}$ after camera motion compensation. Further, to describe the remaining motion components, we use the general *affine* model $A_{l_i^t}^{t,t+1}$ to represent the transformation of a segment l_i^t from image t to $t+1$. It is a 2×3 matrix and is expressed as

$$A_{l_i^t}^{t,t+1} = \begin{bmatrix} a_1(l_i^t) & a_2(l_i^t) & b_1(l_i^t) \\ a_3(l_i^t) & a_4(l_i^t) & b_2(l_i^t) \end{bmatrix},$$

where $a_1(l_i^t)$, $a_2(l_i^t)$, $a_3(l_i^t)$, and $a_4(l_i^t)$ control rotation and scaling, and $b_1(l_i^t)$ and $b_2(l_i^t)$ are the translation components.

Equation (1) describes the parametrization of a displacement vector. In what follows, for simplicity's sake, we still denote by $\mathbf{d}$ the motion vector. In the optimization process, all $\mathbf{d}$s are substituted by the right-hand side expression of (1) and the variables to be optimized are actually the six elements in A.

4.2 Objective Function

We define the following objective function to compute the dense displacement maps:

$$\arg\min_{\mathbf{d},o} \sum_{t=1}^{n-1} (E^{t,t+1}(\mathbf{d},o) + E^{t+1,t}(\mathbf{d},o)), \tag{3}$$

where $E^{t,t+1}(\mathbf{d},o)$ and $E^{t+1,t}(\mathbf{d},o)$ are the bidirectional energy terms representing the mapping from frame t to $t+1$ and the other way around, respectively. Since they are similarly defined, we only describe the construction of $E^{t,t+1}(\mathbf{d},o)$.

$E^{t,t+1}(\mathbf{d},o)$ consists of the color constancy, motion/occlusion smoothness, and segmentation terms and is defined as

$$E^{t,t+1}(A^{t,t+1}, o^{t,t+1}) = \sum_{\mathbf{x}\in I^t} \left[m^{t,t+1}(\mathbf{x}) + \sum_{\mathbf{y}\in N(\mathbf{x})} s^{t,t+1}(\mathbf{x},\mathbf{y}) \right] + E_r^{t,t+1}(A^{t,t+1}), \tag{4}$$

where $N(\cdot)$ denotes the set of neighborhood. The three components are, respectively: 1) the data matching term $m^{t,t+1}(\mathbf{x})$, 2) the smoothness term $s^{t,t+1}(\mathbf{x},\mathbf{y})$ that is comprised of spatial motion smoothness and visibility consistency, and 3) a segmentation regularization term.

4.2.1 Data Term $m(\mathbf{x}) = m_I(\mathbf{x}) + m_S(\mathbf{x})$

It contains the color matching cost $m_I(\mathbf{x})$ and the shape matching cost $m_S(\mathbf{x})$. $m_I(\mathbf{x})$ models color constancy with regard to possible occlusion and is given by

$$m_I^{t,t+1}(\mathbf{x},\mathbf{d}^{t,t+1}(\mathbf{x})) = \begin{cases} \min\{\rho_d^{t,t+1}(\mathbf{x},\mathbf{d}^{t,t+1}(\mathbf{x})),\eta_d\}, & o^{t,t+1}(\mathbf{x})=0, \alpha_{\mathbf{x}}^t = \alpha_{\mathbf{x}'}^{t+1}, \\ \min\{\rho_d^{t,t+1}(\mathbf{x},\mathbf{d}^{t,t+1}(\mathbf{x})),\eta_o\}, & o^{t,t+1}(\mathbf{x})=0, \alpha_{\mathbf{x}}^t \neq \alpha_{\mathbf{x}'}^{t+1}, \\ \eta_o, & o^{t,t+1}(\mathbf{x})=1, \end{cases} \tag{5}$$

where $\mathbf{x}' = \mathbf{x} + \mathbf{d}^{t,t+1}(\mathbf{x})$, and $\rho_d^{t,t+1}(\mathbf{x},\mathbf{d}^{t,t+1}(\mathbf{x}))$ is a matching function:

$$\rho_d^{t,t+1}(\mathbf{x},\mathbf{d}^{t,t+1}(\mathbf{x})) = \|I^{t+1}(\mathbf{x}+\mathbf{d}^{t,t+1}(\mathbf{x})) - I^t(\mathbf{x})\|^2.$$

η_o in (5) is a penalty, preventing all pixels from being labeled as occlusion [34]. η_d is a truncated value larger than η_o to determine the upper limit of the cost. If $o^{t,t+1}(\mathbf{x}) = 0$ and $\alpha_{\mathbf{x}}^t \neq \alpha_{\mathbf{x}+\mathbf{d}^{t,t+1}(\mathbf{x})}^{t+1}$, there possibly exists occlusion.

On the other hand, we notice a foreground pixel $\mathbf{x}$ in frame t should have its corresponding pixel $\mathbf{x}'$ in frame $t+1$ also in the foreground. We therefore define the shape matching function as

$$m_S^{t,t+1}(\mathbf{x}) = \beta_\alpha \alpha_{\mathbf{x}}^t \left(\alpha_{\mathbf{x}}^t - \alpha_{\mathbf{x}+\mathbf{d}^{t,t+1}(\mathbf{x})}^{t+1}\right)^2, \tag{6}$$

where β_α is a weight. $\alpha_{\mathbf{x}}^t$ and $(\alpha_{\mathbf{x}}^t - \alpha_{\mathbf{x}+\mathbf{d}^{t,t+1}(\mathbf{x})}^{t+1})^2$ cause the correspondence to be enforced only in the foreground layer. This shape matching term significantly improves motion estimation, especially for pixels with large displacement in our experiments.

4.2.2 Smoothness Term $s(\mathbf{x},\mathbf{y})$

It encourages motion and occlusion smoothness and is defined as

$$\begin{aligned} s^{t,t+1}(\mathbf{x},\mathbf{y}) = {} & \beta_s \rho_s^{t,t+1}(\mathbf{x},\mathbf{y}) + \beta_o |o^{t,t+1}(\mathbf{x}) - o^{t,t+1}(\mathbf{y})| \\ & + \beta_w |o^{t,t+1}(\mathbf{x}) - W^{t,t+1}(\mathbf{x})|, \end{aligned} \tag{7}$$

where ρ_s and $|o^{t,t+1}(\mathbf{x}) - o^{t,t+1}(\mathbf{y})|$ are the spatial smoothness constraints for displacement and occlusion. $|o^{t,t+1}(\mathbf{x}) - W^{t,t+1}(\mathbf{x})|$ is the temporal constraint. All βs are weights. ρ_s is a robust function written as

$$\begin{aligned} \rho_s^{t,t+1}(\mathbf{x},\mathbf{y}) = {} & \left(1 - |\alpha_{\mathbf{x}}^t - \alpha_{\mathbf{y}}^t|\right) \\ & \cdot \min\{\|\mathbf{d}^{t,t+1}(\mathbf{x}) - \mathbf{d}^{t,t+1}(\mathbf{y})\|^2, \eta_s\}, \end{aligned}$$

which indicates that if two neighboring pixels belong to different layers after bilayer segmentation, the spatial smoothness does not need to be preserved. η_s controls the maximum cost.

In the last term $|o^{t,t+1}(\mathbf{x}) - W^{t,t+1}(\mathbf{x})|$, $W^{t,t+1}(\mathbf{x}) \in \{0,1\}$ is a precomputed binary value based on the displacement $\mathbf{d}^{t+1,t}$, indicating whether or not pixel $\mathbf{x}$ in I^t receives a projection from I^{t+1} based on the current $\mathbf{d}^{t+1,t}$ [34]. $W^{t,t+1}(\mathbf{x})$ has value 1 if there is no corresponding pixel in I^{t+1} for $I^t(\mathbf{x})$, implying $\mathbf{x}$ is likely to be occluded in I^{t+1}.

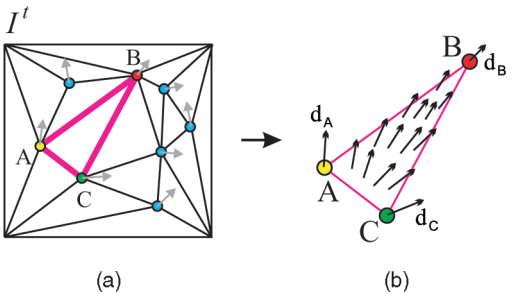


Fig. 3. Motion initialization and interpolation. (a) We triangulate the tracked features and interpolate the in-plane displacement vectors for all pixels using the triangular interpolation. (b) An amplified region.

4.2.3 Segmentation Regularization

We express the segment regularization term E_r as a function of the elements of $A_{l_i^t}^{t,t+1}$. It is written as

$$\begin{aligned} E_r^{t,t+1}(A^{t,t+1}) = \beta_A \sum_{i=1}^{K} |l_i^t| \big(& (a_1(l_i^t) - 1)^2 + a_2(l_i^t)^2 \\ & + a_3(l_i^t)^2 + (a_4(l_i^t) - 1)^2 \big), \end{aligned} \tag{8}$$

where $|l_i^t|$ denotes the number of pixels in l_i^t, K is the segment number in the frame t, and β_A is a weight. Equation (8) imposes a strong first-order intrasegment smoothness constraint, which regularizes the affine parameters and enforces translational motion. Because matching in ubiquitous textureless regions is usually ill-posed, incorporating this regularization term can avoid large affine distortion.

4.3 Solving the Energy Function

The energy defined in (4) is a complex one. We solve for a dense displacement map with the consideration of occlusion and segmentation. The occlusion variables in o are initially set to zeros. Note that in the aforementioned SFM step, the SIFT algorithm is used to obtain a set of sparse feature tracks linking corresponding pixels among frames. They define a set of features in each frame together with the displacement vectors. We use these points to triangulate the frames, as illustrated in Fig. 3a. Motion vectors of all pixels are initialized using triangular interpolation in each triangle.

The matrix $A_{l_i^t}^{t,t+1}$ for each segment l_i^t is initialized with $a_1(l_i^t) = 1$, $a_2(l_i^t) = 0$, $a_3(l_i^t) = 0$, and $a_4(l_i^t) = 1$. $b_1(l_i^t)$ and $b_2(l_i^t)$ are initialized as the respective element values in the mean motion vector for all pixels in segment l_i^t.

After initialization, the motion estimation method alternates between the following two steps in a maximum of three passes:

1. Fix o. Equation (7) is simplified to $\beta_s \rho_s^{t,t+1}(\mathbf{x},\mathbf{y})$. We solve for the six elements of $A_{l_i^t}^{t,t+1}$ using the Levenberg-Marquardt (LM) optimization. $\mathbf{d}$ is computed using (1).

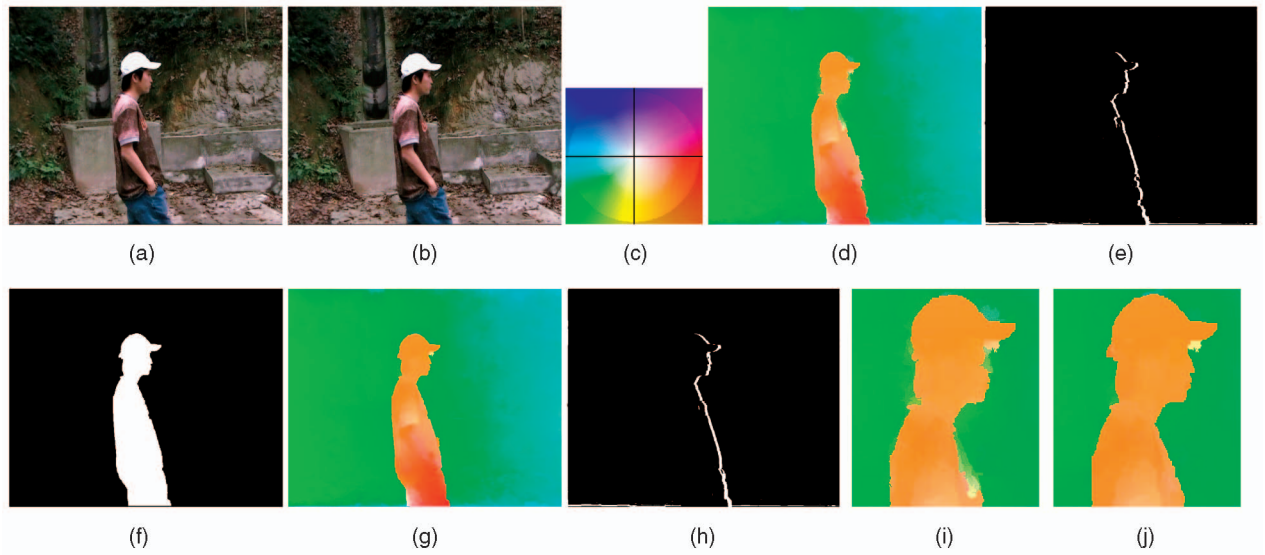


Fig. 4. Optical flow estimation. (a)-(b) Frames 16 and 17. (c) A reference color wheel for motion coding. (d) Initial optical flow field $\mathbf{d}^{16,17}$. (e) Initial occlusion map $o^{16,17}$. (f) Bilayer segmentation result α^{16} in the second iteration. (g) Updated optical flow field with the α map shown in (f). (h) Refined occlusion map. (i) Close-up of (d). (j) Close-up of (g).

2. Fix $\mathbf{d}$ and update o by minimizing (4). The segmentation regularization term (8) and the term ρ_s in (7) do not involve o and are thus omitted in this step. Each element in the occlusion map o has a binary label. We use graph cuts [6] to compute them.

Fig. 4 shows a motion estimation example. Figs. 4a and 4b show two consecutive frames. To visualize the dense optical flow, we adopt the color coding in [3], where the chromaticity is used to distinguish the motion direction and the intensity corresponds to the motion magnitude, as shown in Fig. 4c. The computed optical flow and occlusion maps using the above method are shown in Figs. 4d and 4e, respectively. Because, in the first place we have no layer information and set αs to all zeros, visual artifacts (due to some segments spanning over different objects) are caused around the object boundary.

The computed maps in Figs. 4d and 4e are used in the bilayer segmentation step thereupon (to be detailed in Section 5) to update the α map, where the result is shown in Fig. 4f. In the second iteration (Step 4 in Table 1), we perform optical flow estimation again to refine motion and occlusion. The results are shown in Figs. 4g and 4h.

4.4 Dense Track Generation

After solving for the dense motion vectors, we link each pixel forward and backward in the neighboring frames to eventually form dense motion *tracks*. They will facilitate the following bilayer segmentation. To reduce the accumulated error in this process, we limit the formed *tracks* (illustrated in Fig. 5) not longer than 20 frames. We also break a link in between pixels $I^t(\mathbf{x})$ and $I^{t+1}(\mathbf{x}')$ if any of the following criteria is triggered: 1) $o^{t,t+1}(\mathbf{x}) = 1$ or $o^{t+1,t}(\mathbf{x}') = 1$; 2) the optical flow consistency error

$$e_{flow}^{t,t+1}(\mathbf{x}) = \|\mathbf{d}^{t,t+1}(\mathbf{x}) + \mathbf{d}^{t+1,t}(\mathbf{x}')\| \tag{9}$$

is larger than 2 pixels. Here, $\mathbf{x}'$ is the corresponding pixel of $\mathbf{x}$ in frame $t+1$, i.e., $\mathbf{x}' = \mathbf{x} + \mathbf{d}^{t,t+1}(\mathbf{x})$. We denote the forward and backward half tracks as $\mathcal{X}^F = \{\mathbf{x}^t|_{t=i+1}^{r}\}$ and $\mathcal{X}^B = \{\mathbf{x}^t|_{t=l}^{i-1}\}$, respectively, for pixel $\mathbf{x}$ in frame i.

Further, with the recovered depth maps (detailed in Section 3), we project each pixel to other frames to obtain the corresponding $\mathcal{X}'^F = \{\mathbf{x}'^t|_{t=i+1}^{r}\}$ and $\mathcal{X}'^B = \{\mathbf{x}'^t|_{t=l}^{i-1}\}$. If pixel $\mathbf{x}$ refers to a static point, its depth estimate is usually very accurate, and consequently, $\mathcal{X}^F$ (or $\mathcal{X}^B$) should be near to $\mathcal{X}'^F$ (or $\mathcal{X}'^B$). Pixels on the dynamic object, on the contrary, generally receive mistaken depth estimate, as shown in Fig. 2. Also, to exclude the effect of occlusion, which goes either forward or back, we compare two half tracks, and select the pair with minimum difference to measure the foreground/background confidence:

$$\mathcal{M}^t(\mathbf{x}) = \min\{f(\mathcal{X}^F, \mathcal{X}'^F), f(\mathcal{X}^B, \mathcal{X}'^B)\}, \tag{10}$$

where $f(\mathcal{X}^F, \mathcal{X}'^F) = \max_{t=i+1,\ldots,r}\|\mathbf{x}^t - \mathbf{x}'^t\|$ and $f(\mathcal{X}^B, \mathcal{X}'^B) = \max_{t=l,\ldots,i-1}\|\mathbf{x}^t - \mathbf{x}'^t\|$. If $\mathcal{M}^t(\mathbf{x})$ is large, it is likely that the pixel $\mathbf{x}$ is in the foreground. $\mathcal{M}^t(\mathbf{x})$ will be used to define a confidence measure for bilayer segmentation.

5 Bilayer Segmentation

The computed optical flow can help identify layers. But its quality is not high enough to guarantee accurate foreground extraction. We integrate other cues, such as color, contrast, and depth, in our method to accomplish this goal.

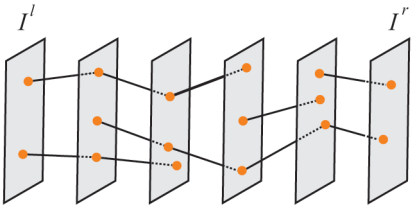


Fig. 5. Optical flow track illustration in multiple frames.

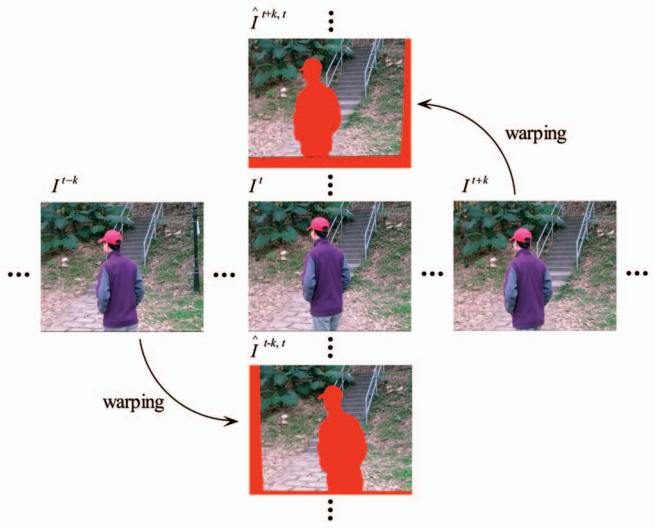


Fig. 6. 3D warping. The neighboring frames are warped to I^t. The red pixels are those receiving no projection during the warping process.

5.1 Data Term

With the computed depth maps (described in Section 3), the 3D warping technique [30] can be used to render new views by projecting pixels from one frame to other. In our method, we warp the neighboring $2l$ frames, i.e., $\{I^{t-l}, \ldots, I^{t+l}\}$ to frame t using the depth information. We reiterate that we do not assume all pixels have correct depths. Instead, our method collects depth statistics, using multiple frames to mitigate the influence of mistakes in the layer computation.

Warping neighboring frames to the present one allows using multiple cues. We employ the background subtraction, local color statistics, and depth/motion consistency measures (denoted by L_c, L_g, and L_m, respectively), whose construction will be detailed later in this section. It is notable that these measures evaluate layer separation from different angles and thus are similarly important for producing the final result. It is also unknown in advance which one is more reliable for a specific example. In our method, we adopt a simple and yet very effective voting-like scheme to combine these measures and express the data cost as

$$E_d(\alpha_{\mathbf{x}}^t) = \text{median}\{L_c(\alpha_{\mathbf{x}}^t), L_g(\alpha_{\mathbf{x}}^t), L_m(\alpha_{\mathbf{x}}^t)\}, \tag{11}$$

where $\alpha_{\mathbf{x}}^t$ is the foreground label for pixel $\mathbf{x}$ in frame t. The data term favors the majority of the measures. It is robust to outliers because each measure does not directly affect the final result. An occasional degradation of one term does not matter as much as using a weighted sum scheme.

During warping, we exclude foreground-layer pixels, where $\alpha_{\mathbf{x}}^t$ is labeled 1 in the previous iteration. Initially, since all αs are set to zeros, we alternatively exclude pixels for which $\mathcal{M}^t(\mathbf{x})$ is larger than a threshold. The image and disparity map warped from $I^{t'}$ to I^t are denoted as $\hat{I}^{t',t}$ and $\hat{D}^{t',t}$ respectively. Fig. 6 shows an example. The red pixels are those receiving no projection during the warping. We now describe the three measures.

5.1.1 Background Subtraction Measure

With the inevitable estimation error, the warped points may deviate from their correct positions. We thus apply the following method to locally search the best match. The appearance consistency error for $\mathbf{x}$ with respect to I^t and $\hat{I}^{t',t}$ is given by

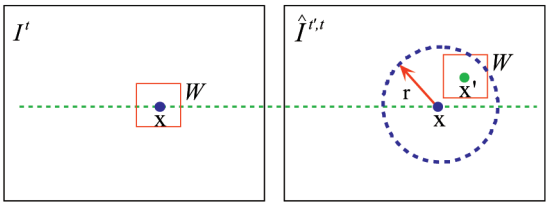


Fig. 7. Local block matching. The warped pixel $\mathbf{x}$ in $\hat{I}^{t',t}$ may slightly deviate from its correct position. We perform local search to reduce error. The red solid rectangle is the matching windows W, and the blue dashed circle indicates the search region.

$$\mathcal{A}^{t',t}(\mathbf{x}) = \frac{1}{|W|}\min_{\mathbf{x}'}\sum_{\mathbf{y}\in W}\|I^t(\mathbf{x}+\mathbf{y}) - \hat{I}^{t',t}(\mathbf{x}'+\mathbf{y})\|, \tag{12}$$

where W is a 3×3 window for block matching and $\mathbf{x}'$ is in the neighborhood region of $\mathbf{x}$, where $\|\mathbf{x}'-\mathbf{x}\| \le r$, as illustrated in Fig. 7. In our experiments, $r = 2$ (pixels).

In addition to warping color images, we also construct the warped disparity map $\hat{D}^{t',t}$ and define the disparity consistency error as the blockwise minimum difference between pixels. It is given by

$$\mathcal{D}^{t',t}(\mathbf{x}) = \frac{1}{|W|}\min_{\mathbf{x}'}\sum_{\mathbf{y}\in W}|D^t(\mathbf{x}+\mathbf{y}) - \hat{D}^{t',t}(\mathbf{x}'+\mathbf{y})|. \tag{13}$$

This disparity consistency error will be used in defining the depth/motion consistency measure.

After block matching, we gather a set of $\mathcal{A}^{t',t}(\mathbf{x})$ for each pixel $\mathbf{x}$ in frame t with respect to different ts as we have warped multiple frames to frame t. Their statistics reflect the chance that one pixel receives the correct depth estimate. For instance, if the residual error $\mathcal{A}^{t',t}(\mathbf{x})$ is consistently large for multiple ts, it is quite possible that pixel $\mathbf{x}$ will be in the dynamic foreground layer or be occluded in most frames. To abstract this type of information, for each $\mathbf{x}$, we apply the median filter to all $\mathcal{A}^{t',t}(\mathbf{x})$s, where $t' = t-l, \ldots, t+l$, which yields

$$\overline{\mathcal{A}}^t(\mathbf{x}) = \text{median}\{\mathcal{A}^{t-l,t}(\mathbf{x}), \ldots, \mathcal{A}^{t+l,t}(\mathbf{x})\}. \tag{14}$$

A large median value $\overline{\mathcal{A}}^t(\mathbf{x})$ implies that the color of pixel $\mathbf{x}$ in the present frame is quite different from the majority of the warped background color. So, pixel $\mathbf{x}$ is very likely to be in foreground.

Similarly, we compute the median value of the disparity consistency errors

$$\overline{\mathcal{D}}^t(\mathbf{x}) = \text{median}\{\mathcal{D}^{t-l,t}(\mathbf{x}), \ldots, \mathcal{D}^{t+l,t}(\mathbf{x})\}. \tag{15}$$

$\overline{\mathcal{D}}^t(\mathbf{x})$ will be used in defining the depth/motion consistency measure L_m.

Finally, we express the layer likelihood based on the background subtraction measure as

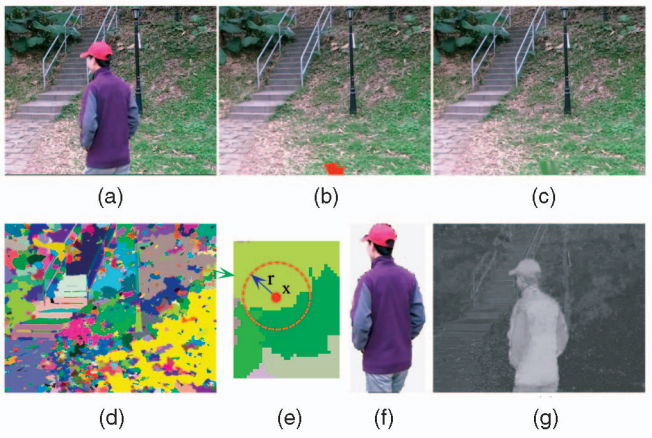


Fig. 8. Warped color statistics. (a) One frame from a sequence. (b) Estimated background image by warping 30 neighboring frames to (a). The red pixels do not receive projection. (c) Completed background image by inpainting. (d) Segmented background image by Mean Shift. (e) The color in each segment is modeled by a Gaussian distribution. Three segments are included in the local region centered at $\mathbf{x}$. (f) Initial foreground layer for GMM training. (g) The confidence map of the local color statistics.

$$L_c(\alpha_{\mathbf{x}}^t = 0) = \frac{\overline{\mathcal{A}}^t(\mathbf{x})}{\overline{\mathcal{A}}^t(\mathbf{x}) + \delta_{\mathbf{c}}}, \qquad L_c(\alpha_{\mathbf{x}}^t = 1) = \frac{\delta_c}{\overline{\mathcal{A}}^t(\mathbf{x}) + \delta_c}, \tag{16}$$

where $\delta_c = 5 \sim 13$ in our experiments, controlling the labeling threshold. $\overline{\mathcal{A}}^t(\mathbf{x}) > \delta_c$ yields $L_c(\alpha_{\mathbf{x}}^t = 0) > 0.5 > L_c(\alpha_{\mathbf{x}}^t = 1)$, indicating that $\mathbf{x}$ is likely a dynamic foreground pixel.

This measure is sometimes error-prone if the foreground and background pixels have similar colors. To overcome it, in this system, other statistics models are also employed.

5.1.2 Local Color Statistics Measure

Previous bilayer segmentation methods [25], [32], [37] employ the Gaussian mixture model (GMM) to describe color. The typical scheme is to sample color and build two global GMMs for background and foreground, respectively. In our method, as we process general videos in which the background image varies over time, building a global background GMM is not appropriate. Here, we construct them separately for each frame.

With the set of warped images for each frame t, as shown in Fig. 6, we stack them and apply median filtering to each pixel to approximate the background image $\hat{B}^t$, as shown in Fig. 8b. Background regions that are consistently occluded in the whole sequence find no color information (shown in red in Fig. 8b). We apply a simple completion method [31] to infer them, as shown in Fig. 8c. Then, the Mean Shift algorithm [13] is employed to segment $\hat{B}^t$ (Fig. 8d) followed by using a Gaussian distribution $N(\mu_k^b, \Sigma_k^b)$ to model the color in each segment S_k. Directly propagating the estimated background color models into the missing areas for completion is also a choice. Note that background completion may produce error. It is, however, not influential because we only use the statistical color model but not the respective pixel values. Moreover, as we remove the foreground estimate in 3D warping, occasional error is quickly eliminated in iterations.

With the background color models, for each pixel, we search the distributions that it possibly belongs to in a local area, as shown in Figs. 8d and 8e. The background color probability is written as

$$p(\alpha_{\mathbf{x}}^t = 0) = \max_{j=1}^{l} N\big(I^t(\mathbf{x}) \big| \mu_{m_j}^b, \Sigma_{m_j}^b\big), \tag{17}$$

where l is the number of the background samples in the local window (Fig. 8e) and m_j indexes the corresponding Gaussian cluster for each sample.

The foreground color model construction is easier. We compute from a single frame the foreground region, as shown in Fig. 8f, using GrabCut [32]. Based on it, we construct the foreground color GMM with Gaussian distributions $\{N(\mu_1^f, \Sigma_2^f), N(\mu_2^f, \Sigma_2^f), \ldots\ N(\mu_{K_f}^f, \Sigma_{K_f}^f)\}$. The probability that one pixel belongs to the foreground layer is accordingly expressed as

$$p(\alpha_{\mathbf{x}}^t = 1) = \sum_{k=1}^{K_f} w_k^f N\big(I_{\mathbf{x}}^t \big| \mu_k^f, \Sigma_k^f\big), \tag{18}$$

where w_k^f is the computed weight corresponding to the kth component of the GMM and K_f is the total number of the clusters. The definition difference between (17) and (18) is due to the use of local and global color models.

Finally, the local color statistics measure is defined as:

$$L_g(\alpha_{\mathbf{x}}^t) = \frac{\log p(\alpha_{\mathbf{x}}^t)}{\log p(\alpha_{\mathbf{x}}^t = 0) + \log p(\alpha_{\mathbf{x}}^t = 1)}, \tag{19}$$

where the denominator is for normalization. Fig. 8g shows the local color statistics measure map. A large value reflects high confidence that a pixel belongs to the foreground layer. Our model is different from the one in [15] because we do not assume rotational camera motion. There also does not exist a panoramic background image that can be estimated beforehand.

5.1.3 Depth/Motion Consistency Measure

Depth/motion information is also essential in our method for identifying foreground pixels. We combine disparity consistency error $\overline{\mathcal{D}}$, which is defined in (13) and (15), with the motion consistency error $\mathcal{M}$, described in Section 4.4, to express another likelihood:

$$L_m(\alpha_{\mathbf{x}}^t = 0) = \max\left\{\frac{\mathcal{M}^t(\mathbf{x})}{\mathcal{M}^t(\mathbf{x}) + \delta_m}, \frac{\overline{\mathcal{D}}^t(\mathbf{x})}{\overline{\mathcal{D}}^t(\mathbf{x}) + \delta_d}\right\}, \qquad L_m(\alpha_{\mathbf{x}}^t = 1) = 1 - L_m^t\big(\alpha_{\mathbf{x}}^t = 0\big). \tag{20}$$

Here, δ_d and δ_m are two thresholds and are set to $0.2(D_{\max} - D_{\min})$ and $8 \sim 10$, respectively, where $[D_{\min}, D_{\max}]$ is the approximated disparity range of the scene. Equation (20) implies that only when both $\overline{\mathcal{D}}$ and $\mathcal{M}$ are small are we confident that the pixel is in the background layer.

5.2 Smoothness Terms

With the data cost defined in (11), we optimize the following function with the consideration of the spatial and temporal smoothness:

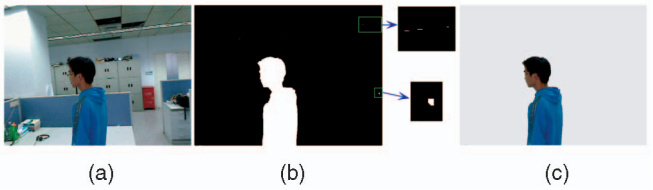


Fig. 9. Removing foreground outliers. (a) One video frame. (b) The computed foreground mask that contains a few isolated regions. The small ones are outliers. (c) The extracted foreground layer.

$$E_B(\alpha) = \sum_{t=1}^{n} \sum_{\mathbf{x} \in I^t} \left(E_d(\alpha_{\mathbf{x}}^t) + \lambda_{\mathcal{S}} \sum_{\mathbf{y} \in N(\mathbf{x})} E_s(\alpha_{\mathbf{x}}^t, \alpha_{\mathbf{y}}^t) + \lambda_{\mathcal{T}} \mathcal{G}(\alpha_{\mathbf{x}}^t) \right), \tag{21}$$

where $E_s(\alpha_{\mathbf{x}}^t, \alpha_{\mathbf{y}}^t)$ and $\mathcal{G}(\alpha_{\mathbf{x}}^t)$ are the spatial and temporal smoothness terms, respectively, which are described as follows: $\lambda_{\mathcal{S}}$ and $\lambda_{\mathcal{T}}$ are weights.

5.2.1 Spatial Smoothness

Strong edges of the background image could mistake the bilayer segmentation. We attenuate them when enforcing the spatial smoothness.

When computing the appearance consistency error $\overline{\mathcal{A}}^t$ using median filtering, it at the same time finds the matched pixel for $\mathbf{x}$ from a warped image. Suppose it is pixel $\mathbf{x}^*$ in $\hat{I}^{t^*,t}$. We denote by $g^{t^*,t}(\mathbf{x}^*)$ and $g^t(\mathbf{x})$ the corresponding gradients in $I^{t^*,t}$ and I^t, respectively. We set $g^{t^*,t}(\mathbf{x}^*) = 0$ if it cannot be computed due to the lack of the matched pixel after warping. Then, we define the following function, which is similar to the one in [35], to attenuate the background contrast:

$$d_a^t(\mathbf{x}) = \|g^t(\mathbf{x})\|^2 \cdot \frac{1 - e^{-\frac{\overline{\mathcal{A}}^t(\mathbf{x})^2}{2 \cdot 5 \cdot 5}}}{1 + \left(\frac{g^{t^*,t}(\mathbf{x}^*)}{5}\right)^2 e^{-\frac{\overline{\mathcal{A}}^t(\mathbf{x})^2}{2 \cdot 10 \cdot 10}}}.$$

It means if $\overline{\mathcal{A}}^t(\mathbf{x})$ is small, it is quite possible that pixel $\mathbf{x}$ is in the background layer, and thus, the contrast should be attenuated more significantly.

Finally, the spatial smoothness term is written as

$$E_s(\alpha_{\mathbf{x}}^t, \alpha_{\mathbf{y}}^t) = |\alpha_{\mathbf{x}}^t - \alpha_{\mathbf{y}}^t| \cdot \exp\left(-\beta d_a^t(\mathbf{x}, \mathbf{y})\right), \tag{22}$$

where $\mathbf{x}$ and $\mathbf{y}$ are neighboring pixels, and β is a robust parameter that weights the color contrast. It is set to $(2\langle \|I_{\mathbf{x}} - I_{\mathbf{y}}\|^2 \rangle)^{-1}$, where $\langle \cdot \rangle$ denotes the expectation operator, same as the one defined in [35]. $E_s(\alpha_{\mathbf{x}}, \alpha_{\mathbf{y}})$ indicates that if both $\mathbf{x}$ and $\mathbf{y}$ are in the foreground (or background) layer, E_s has zero value. Otherwise, the smoothness strength is adaptively adjusted according to the color contrast.

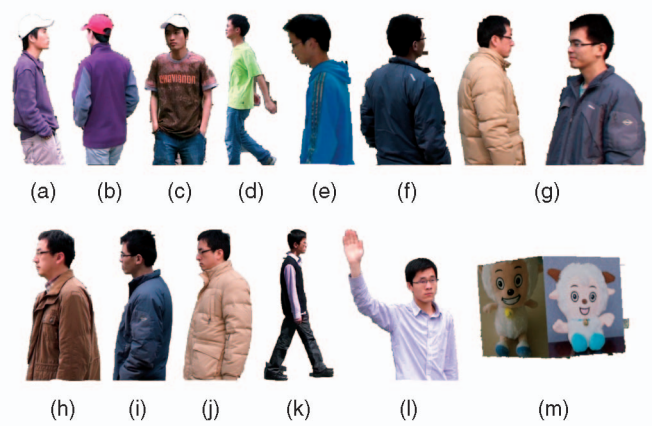


Fig. 10. All coarse foreground regions used in the examples listed in Table 2. They are used in the foreground color modeling (Section 5.1).

5.2.2 Temporal Consistency

This term is defined bidirectionally as

$$\mathcal{G}(\alpha_{\mathbf{x}}^t) = \mathcal{G}^{t,t+1}(\alpha_{\mathbf{x}}^t) + \mathcal{G}^{t,t-1}(\alpha_{\mathbf{x}}^t). \tag{23}$$

Let pixel $\mathbf{x}'$ in I^{t+1} be the corresponding one of $\mathbf{x}$ in I^t according to the estimated motion vector. $\mathcal{G}^{t,t+1}(\alpha_{\mathbf{x}}^t)$ is expressed as

$$\mathcal{G}^{t,t+1}(\alpha_{\mathbf{x}}^t) = |\alpha_{\mathbf{x}}^t - \alpha_{\mathbf{x}'}^{t+1}| \cdot w_{flow}^{t,t+1}(\mathbf{x}),$$

where $|\alpha_{\mathbf{x}}^t - \alpha_{\mathbf{x}'}^{t+1}| = 1$ if the α labels are different. Or else, no temporal smoothness penalty will be enforced. $w_{flow}^{t,t+1}(\mathbf{x})$ penalizes the label inconsistency based on the measure of forward and backward optical flow (9) and the pixel color difference. It is given by

$$w_{flow}^{t,t+1}(\mathbf{x}) = \exp\left(-\frac{\|\mathbf{d}^{t,t+1}(\mathbf{x}) + \mathbf{d}^{t+1,t}(\mathbf{x}')\|^2}{\delta_{flow}^2}\right) \cdot \exp\left(-\frac{\|I^t(\mathbf{x}) - I^{t+1}(\mathbf{x}')\|^2}{\delta_{color}^2}\right), \tag{24}$$

TABLE 2
The Statistics of the Tested Sequences

Sequences	Tree	Stair	Path	Wall	Indoor	Forward	Two-Men	Static-Man	Occluded-Man	Rotation	Shadow	Waving-Arm	Cube
Frames	150	200	120	153	220	121	86	111	91	130	81	71	140
Resolution	720×576	720×576	720×576	960×540	640×480	960×540	960×540	960×540	960×540	960×540	960×540	960×540	960×540

TABLE 3
The Parameters

β_α	η_d	η_o	η_s	β_s	β_o	β_w	β_A	$\lambda_{\mathcal{S}}$	$\lambda_{\mathcal{T}}$	δ_d	δ_m	δ_c
10^4	900	400	4	100	210	600	10^4	0.5	0.3	$0.2(D_{\max} - D_{\min})$	$8 \sim 10$	$5 \sim 13$

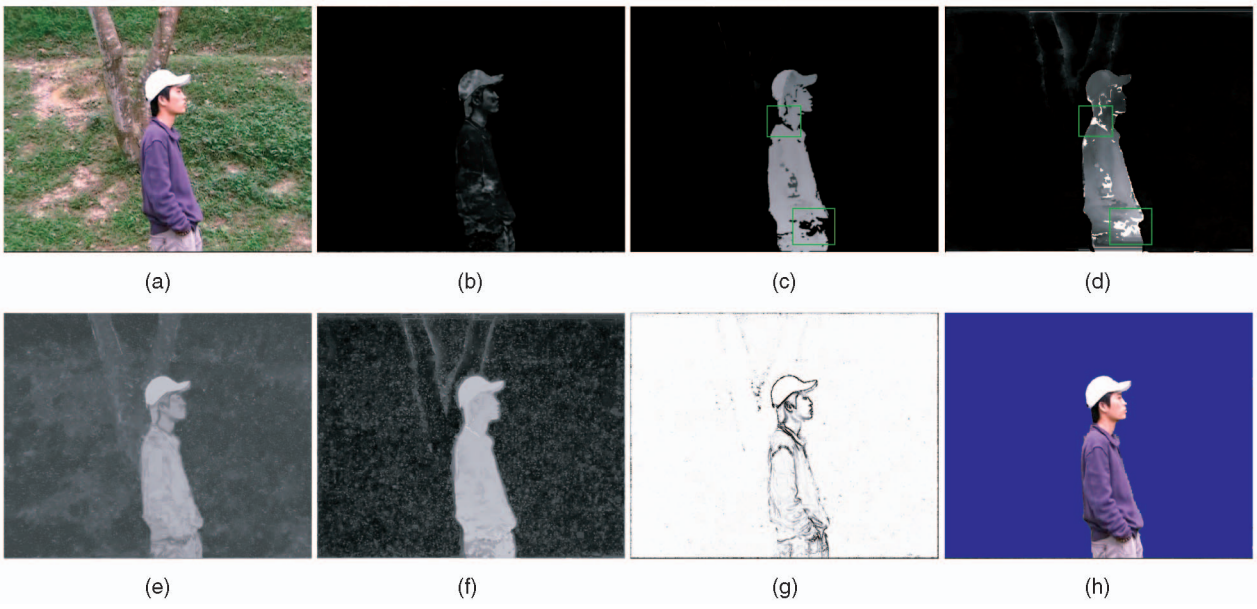


Fig. 11. Bilayer segmentation. (a) One video frame. (b) The appearance consistency map $\overline{\mathcal{A}}$. (c) The disparity consistency map $\overline{\mathcal{D}}$. (d) The track consistency map $\mathcal{M}$. (e) The likelihood L_g with the local color statistics model. (f) The computed data costs for all pixels. (g) The background attenuation map. (h) The extracted foreground image.

where $\delta_{flow} = 1.4$ and $\delta_{color} = 15$. $\mathcal{G}^{t,t-1}(\alpha_{\mathbf{x}}^t)$ is defined similarly.

5.3 Overall Computation

In the bilayer segmentation step, to reduce the complexity in computing the data cost for each frame, we only select 20-30 frames, with the interval of 5 frames between each, to perform 3D warping. The minimum interval between the reference frame and sampling frames is generally set to $10 \sim 20$. After the data and smoothness terms are computed, we apply graph cuts to compute the α maps by minimizing $E_B(\alpha)$. Each time we solve for 10 maps sequentially.

Due to inevitable estimation error and image noise, the extracted foreground may contain multiple isolated regions, as shown in Fig. 9b. Most of them are small outliers. Therefore, suppose there are m moving objects, we first select n ($n \leq m$) largest segments, each containing at least 400 pixels. Then, we further select the segments from the remaining ones (containing at least 400 pixels), whose smallest distance to these n segments is less than a threshold. This simple strategy works well empirically.

After computing α in the first pass, we go back to Step 2 in Table 1 and further refine the motion and depth maps as described in Section 4. Two overall passes are sufficient to achieve high-quality bilayer segmentation.

The final step is to recompute the background images and the depth maps based on the α estimate using the method of Bhat et al. [4] (Step 5 in Table 1). We also update the motion vectors for the background pixels in accordance with the pixel correspondences established using the depth information. It is found that the depth computed with the multiview geometry constraint is in general more accurate than the optical flow estimate. We will show in Fig. 12m how the final refinement is useful.

An optional function is provided in our system to refine the foreground boundary by matting [2] such that the extracted layers are usable in other video composition applications.

6 Experimental Results

We experimented with several challenging examples where the videos[2] are taken by a handheld camera and strong vibration exists. Table 2 lists the number of frames and resolutions.

In our implementation, the optical flow is estimated in the grayscale channel (the scale range is $[0, 255]$). Appearance consistency error (12) uses the YUV color space. Other computation is performed in the RGB color space. Our experiments are conducted on a desktop PC with a 4-core Xeon 2.0 GHz CPU. The parameter setting is easy, as shown in Table 3 and described in Section 5.

Fig. 10 contains the computed foreground regions for the foreground color distribution modeling in all examples (described in Section 5.1). Each sequence uses only one such region, which can be quickly obtained by GrabCut [32]. The results do not need to be very accurate for the purpose of color distribution computation.

The processing time depends on the number of frames and image resolution. For a sequence with resolution 720×576 (pixels), the depth estimation (Step 2.1 of Table 1) in the first iteration spends 3 minutes for each frame. In the second iteration, depth refinement runs much faster (a few seconds for each frame) because only depths around the foreground boundaries need to be updated. In each iteration, the motion estimation (Step 2.2 of Table 1) uses 1 minute on average to process one frame. The computation time for bilayer segmentation is about 30 seconds per frame, where over 95 percent of it is spent on the data cost computation.

6.1 Workthrough Example

We use the example shown in Fig. 11 to illustrate how our method works. In the first iteration, the estimated appearance consistency error map $\overline{\mathcal{A}}$ and the disparity consistency error

2. The supplemental video can be found at http://www.cad.zju.edu.cn/home/gfzhang/projects/segmentation/motioncut/.

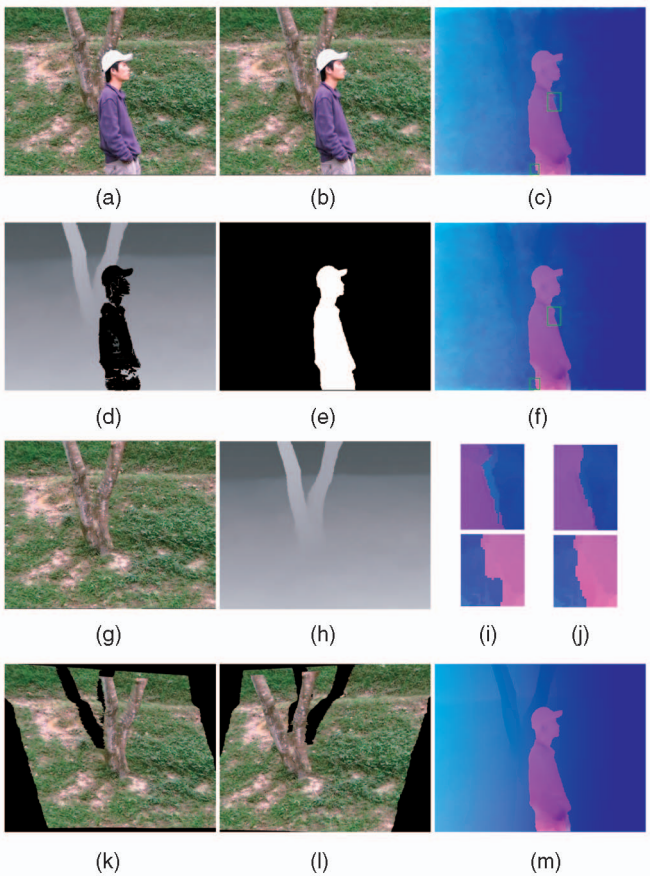


Fig. 12. Motion/depth refinement. (a)-(b) Two consecutive frames. (c) The initial optical flow estimate with all zero αs. (d) The initial depth map of (a). (e) The computed α map in the second iteration. (f) The accordingly updated optical flow. (g) The final background image. (h) The final background depth map. (i) Magnified regions of (c). (j) Magnified regions of (f). (k)-(l) Synthesized novel views. (m) The final optical flow field.

map $\overline{\mathcal{D}}$ (linearly scaled for illustration in Figs. 11b and 11c, respectively) contain very small values for the background pixels. Although we do not perform segmentation here, large-value pixels coarsely shape the foreground. So, these maps provide essential information for bilayer segmentation. Nevertheless, there are errors, as highlighted in the green rectangles, which mistake the background depths as the foreground's. This is why we introduce the motion consistency measure $\mathcal{M}$, as illustrated in Fig. 11d, to gather confidence from another perspective. Different errors seldom arise in the same place.

The likelihood map L_g computed with the local color statistics model in the first iteration is shown in Fig. 11e. Combining the terms defined in (11), the data cost is computed as shown in Fig. 11f (linearly scaled for illustration). Except for some isolated noise, the foreground object has reasonable confidence to be correctly labeled. The background tree trunk has salient boundary and thus needs to be attenuated. The smoothness cost map is shown in Fig. 11g, where the background edges are suppressed. With the computed data and smoothness terms, a high-quality foreground image can be extracted, as shown in Fig. 11h.

We also show the motion/depth maps estimated in the two iterations. With the two consecutive frames shown in Figs. 12a and 12b, the initial optical flow and depth maps for Figs. 12a are computed (shown in Figs. 12c and 12d, respectively). Note that in the first pass, zero-αs provide no information, making the flow and depth estimates erroneous. In the second iteration, with the updated α shown in Fig. 12e, the flow result is refined (shown in Fig. 12f). Visual artifacts around the object boundary are reduced, as highlighted in the green rectangles.

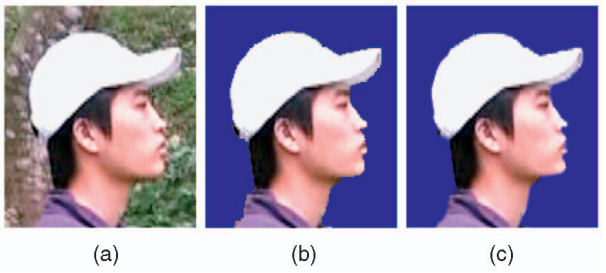


Fig. 13. Boundary refinement. (a) Part of an input frame. (b) Binary segmentation result. (c) Fractional boundary after matting.

With the resulting α map in Fig. 12e, we then infer the final background image, as shown in Fig. 12g. The depth maps are also completed, as shown in Fig. 12h. The optical flow field is updated in accordance with the layer and depth information (Fig. 12m). To verify the quality of the recovered depth, we synthesize new images from different views by 3D warping and show them in Figs. 12k and 12l. Each pixel is a vertex and the surface discontinues when the disparity contrast is large (for example, along the tree trunk boundary). The results accurately preserve object shapes and the order of occlusion. Our system also allows refining the layer boundary, as shown in Fig. 13c, by matting.

6.2 More Open Area Examples

We show more examples to demonstrate the effectiveness of the proposed method. In the "Stair" sequence shown in Fig. 14, both the background color and depth are complex with thin handrail and lamp posts. Our method successfully extracts the dynamic foreground, and computes the optical flow and depth maps. To insert the extracted foreground image to another video, we assign planar depth values to the object[3] and align the camera poses between the source and target frames. Fig. 15 shows a few frames selected from the rendered video. It is visually compelling due to the high-quality bilayer segmentation. Note that naively pasting the walking man to the target video in a frame-by-frame manner results in serious drift problem.

To evaluate how good the depth and motion estimates are, we show more view interpolation results. Figs. 16a, 16b, and 16c include the 85th, 90th and 95th frames, respectively. We warp frames 85 and 95 onto frame 90, and linearly blend them. To handle occlusion, the warping is performed for the foreground and background layers, respectively. The layer results are merged to synthesize the novel view shown in Fig. 16d. The difference map between the interpolated result and the ground truth image is shown in Fig. 16g, which has near-zero values for most pixels and small error only on the object boundary.

3. It is because the depth of the dynamic objects cannot be recovered with multiview geometry.

Fig. 14. "Stair" example. (a) One input frame. (b) The extracted foreground image. (c) The computed optical flow map. (d) The estimated background image. (e) The recovered background depth map.

With the estimated depth and motion, we produce the "slow shot" visual effect, which "densifies" the video by synthesizing frames in between originally consecutive ones. In our experiments, in between any two frames t and $t+1$, we insert four or nine more frames by view interpolation. Please refer to our supplemental video, which can be found at http://www.cad.zju.edu.cn/home/gfzhang/projects/segmentation/motioncut/, for the complete sequences and more examples.

6.3 Indoor Example

To further demonstrate the robustness of the proposed method, we use a Web camera (Logitech Quick-Cam Pro 9000) to capture an indoor sequence (shown in Fig. 17) in low light. This sequence contains strong noise and luminance variation among frames. In addition, most of the regions are textureless, which is very challenging for the motion and depth estimation. Our method is robust to these problems. The complete result is included in our supplementary video, which can be found at http://www.cad.zju.edu.cn/home/gfzhang/ projects/segmentation/motioncut/.

Fig. 15. Inserting the extracted foreground layer into another video.

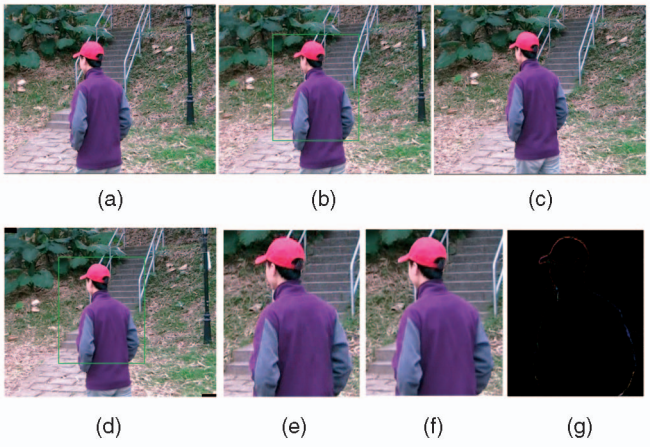


Fig. 16. View interpolation. (a)-(c) Frames 85, 90, and 95 of the "Stair" sequence. (d) Interpolated frame 90, using the motion and depth information of frames 85 and 95. The absolutely black pixels are the missing ones. (e) Close-up of (b). (f) Close-up of (d). (g) Difference image of (e) and (f).

7 Special Case Discussion

We show a few examples that are generally regarded as special and challenging in bilayer segmentation and provide more discussion in this section.

7.1 Rotational Camera Motion

The first type of example we discuss is those captured by static or rotating-only cameras. In this case, the depth cannot be recovered due to the lack of the multiview stereo constraint. Our system, however, still works because view warping and background estimation can be relied on to solve for bilayer segmentation. No change is needed in our system except for constant depth initialization for all pixels such that the depth consistency measure in (20) does not really function.

We show in Fig. 18 a video example taken by a rotating camera (the complete sequence included in the supplemental video, which can be found at http://www.cad. zju.edu.cn/home/gfzhang/projects/segmentation/motioncut/). Constant depth is used. The computed foreground image and

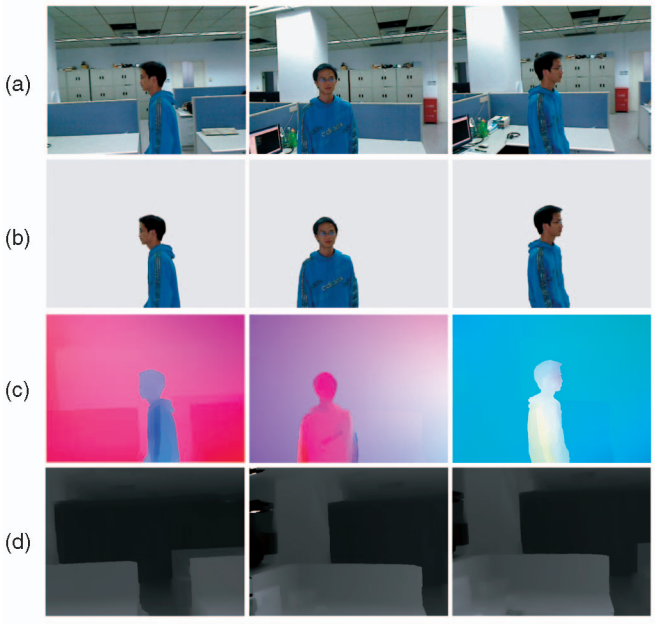


Fig. 17. An indoor example. (a) Original frames. (b) Extracted foreground images. (c) Optical flow fields. (d) Background depth maps.

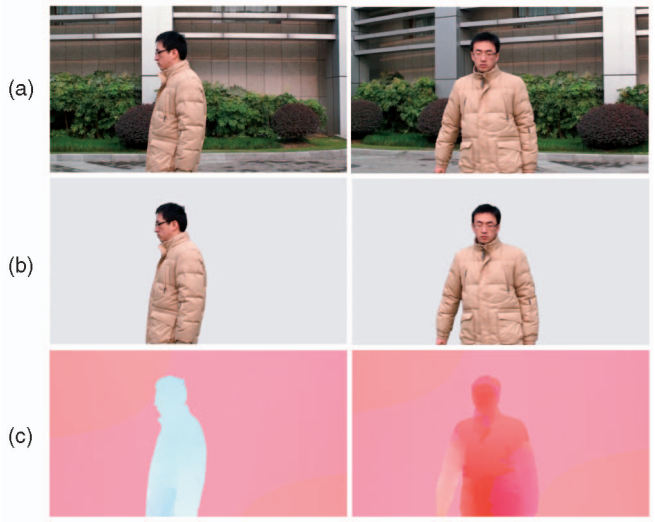


Fig. 18. Video taken by a rotating camera. (a) Two selected frames. (b) Extracted foreground images. (c) Computed optical flow maps.

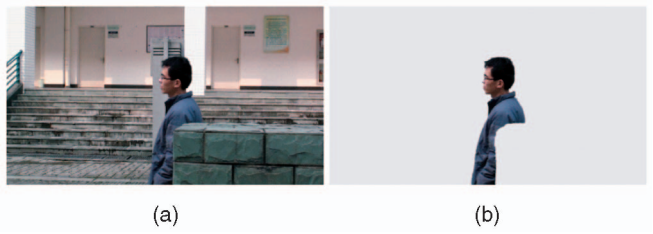


Fig. 20. "Occluded-Man" example. (a) One selected frame. (b) The extracted foreground image.

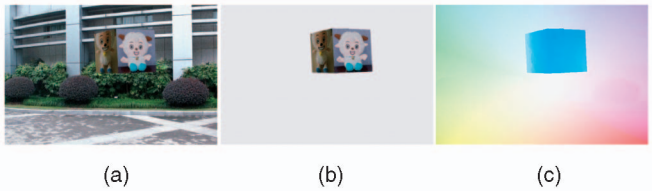


Fig. 21. "Cube" example. (a) One frame from a video. (b) Extracted foreground cube. (c) Computed optical flow map.

flow field are shown in Figs. 18b and 18c, respectively. It is actually an easier example than others by disregarding the depth terms.

7.2 Occasionally Static Foreground

In general, bilayer segmentation assumes that the foreground object moves with the static background scene. All previous methods cannot handle the scenario where the foreground object stays stationary in quite a few continuous frames if the background image (or its color distribution) is unknown. Our system also cannot perfectly handle long-time static foreground as the background information (i.e., color, motion, and disparity) may mistakenly include that from the foreground layer, and accordingly mislead bilayer segmentation. Our method however allows stationary foreground in dozens of successive frames. Fig. 19 shows an example where a man stands still in the first 50 frames and then begins to walk. Our system successfully extracts him.

7.3 Occlusion

It is inherently difficult to compute accurate motion for occluded objects. Fortunately, it generally does not affect our bilayer segmentation because the background subtraction, local color statistics, and depth/motion consistency measures work together in the voting-like scheme, which is robust to occasional motion estimation error. We show a few examples in this paper, which include the partial occlusion by the wall (Fig. 20) and the occlusion between foreground objects (Fig. 22).

Our system is not restricted to near-planar foreground extraction. Fig. 21 shows a sequence containing a moving 3D cube. Our method segments it out, and obtains the dense motion field.

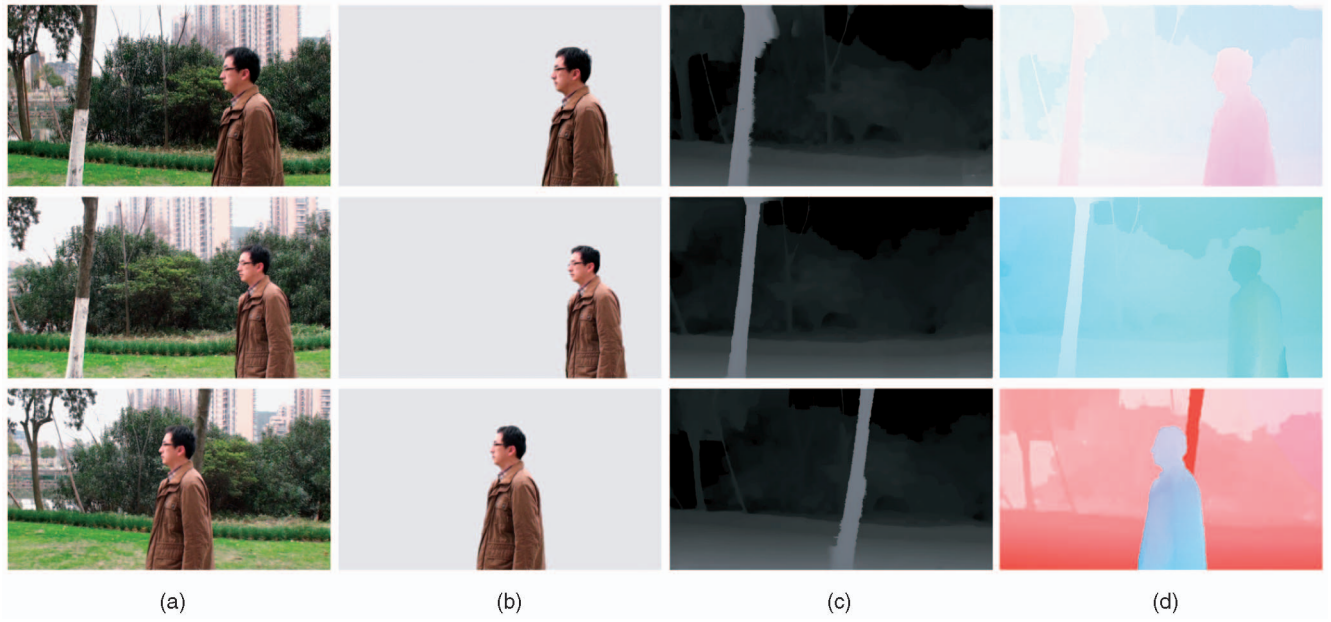


Fig. 19. "Static-Man" example. The man stays stationary in the first 50 frames, and walks in the remaining frames. (a) Selected frames (the 1, 50 and 100 frames). (b) Extracted foreground. (c) Background depth maps. (d) Optical flow estimates.

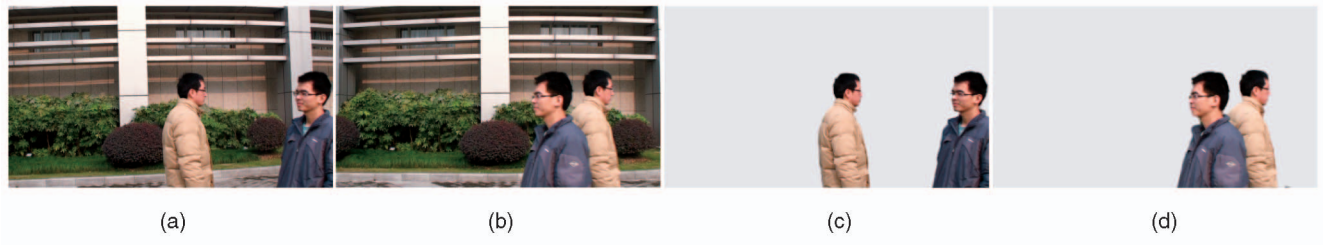


Fig. 22. "Two-Men" example. (a)-(b) Two selected frames. (c)-(d) Extracted foreground images.

Fig. 23. "Shadow" example. (a)-(b) Selected frames. (c)-(d) The extracted foreground images. The shadow is recognized as part of the foreground.

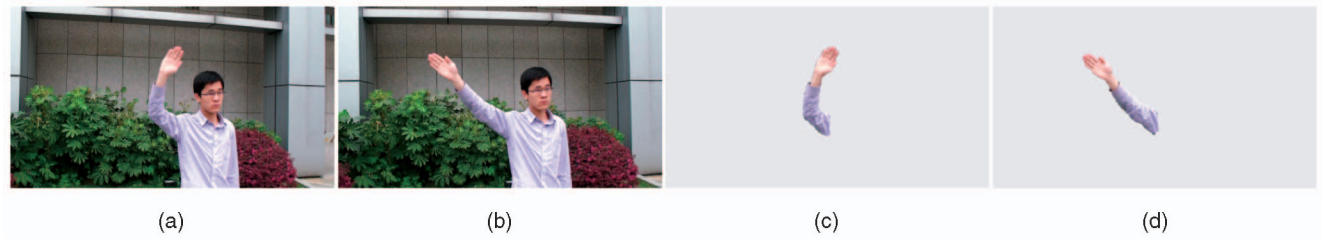


Fig. 24. "Waving-Arm" example. (a)-(b) Selected frames. (c)-(d) The extracted foreground images. The static body, but not the waving arm, is segmented into the background layer.

7.4 Multiple Foreground Objects

Although our system is mainly developed for extracting one foreground object, it can be employed to handle multiple moving objects with the procedure described in Section 5.3. Fig. 22 shows an example that contains two persons at different depths. Our extracted foreground images are shown in Figs. 22c and 22d. Note that if there are many foreground objects with small sizes, it will be difficult for our system to distinguish them from outliers.

7.5 Limitations

The current system still has the following limitations: First, if the background layer does not have sufficient features or is extremely textureless, the camera parameters and motion/depth estimation could be difficult. Second, our system assumes the background is static to satisfy the photoconsistency constraint. Pixels with significant appearance change due to illumination variation, reflection, or shadow will be recognized as foreground accordingly. Fig. 23 shows an example where the sequence contains a walking person with his shadow on the ground. Our method classifies the shadow as part of the foreground layer. Similarly, unmoving objects will be regarded as background. An example is the static person in Fig. 24. Only his waving arm is segmented into the foreground layer.

8 Conclusion and Future Work

In this paper, we have proposed a complete system for high-quality bilayer segmentation, and motion and depth estimation from videos taken by a handheld camera. Our method alternates between two major steps. In the first one, we estimate the camera motion parameters, the optical flow fields, and the depth maps. The visibility, segmentation, and layer information is encoded for handling occlusion and textureless regions. In the second step, we take depth and motion into layer separation. We have proposed a simple voting-like scheme to reliably detect the moving objects. A few postprocessing steps finally compute the motion fields and the background color/depth images.

Presently, our system assumes static background and identifies dynamic regions as foreground. The bilayer segmentation cannot separate multiple moving layers. Part of our future work will be along this line to accomplish robust multilayer segmentation with regard to more diversified occlusion and color constraints.

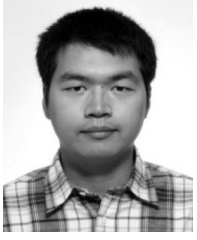
Acknowledgments

The authors would like to thank the associate editor and all of the reviewers for their constructive comments to improve the manuscript. This work is supported by the 973 program of China (No. 2009CB320804), NSF of China (Nos. 60633070 and 60903135), and a grant from the Research Grants Council of the Hong Kong Special Administrative Region (Project No. 412708).

References

[1] S. Ayer and H.S. Sawhney, "Layered Representation of Motion Video Using Robust Maximum-Likelihood Estimation of Mixture Models and Mdl Encoding," *Proc. IEEE Int'l Conf. Computer Vision*, pp. 777-784, 1995.

[2] X. Bai, J. Wang, D. Simons, and G. Saprio, "Video Snapcut: Robust Video Object Cutout Using Localized Classifiers," *ACM Trans. Graphics,* vol. 28, no. 3, 2009.
[3] S. Baker, D. Scharstein, J.P. Lewis, S. Roth, M.J. Black, and R. Szeliski, "A Database and Evaluation Methodology for Optical Flow," *Proc. IEEE Int'l Conf. Computer Vision,* pp. 1-8, 2007.
[4] P. Bhat, C.L. Zitnick, N. Snavely, A. Agarwala, M. Agrawala, B. Curless, M. Cohen, and S.B. Kang, "Using Photographs to Enhance Videos of a Static Scene," *Rendering Techniques 2007,* J. Kautz and S. Pattanaik, eds., pp. 327-338, A.K. Peters, June 2007.
[5] M.J. Black and P. Anandan, "The Robust Estimation of Multiple Motions: Parametric and Piecewise-Smooth Flow Fields," *Computer Vision and Image Understanding,* vol. 63, no. 1, pp. 75-104, 1996.
[6] Y. Boykov, O. Veksler, and R. Zabih, "Fast Approximate Energy Minimization via Graph Cuts," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 23, no. 11, pp. 1222-1239, Nov. 2001.
[7] M. Brown and D.G. Lowe, "Recognising Panoramas," *Proc. IEEE Int'l Conf. Computer Vision,* pp. 1218-1227, 2003.
[8] T. Brox, C. Bregler, and J. Malik, "Large Displacement Optical Flow," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition,* 2009.
[9] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High Accuracy Optical Flow Estimation Based on a Theory for Warping," *Proc. European Conf. Computer Vision,* vol. 4, pp. 25-36, 2004.
[10] A. Bruhn and J. Weickert, "Towards Ultimate Motion Estimation: Combining Highest Accuracy with Real-Time Performance," *Proc. IEEE Int'l Conf. Computer Vision,* pp. 749-755, 2005.
[11] Y.-Y. Chuang, A. Agarwala, B. Curless, D. Salesin, and R. Szeliski, "Video Matting of Complex Scenes," *ACM Trans. Graphics,* vol. 21, no. 3, pp. 243-248, 2002.
[12] Y.-Y. Chuang, B. Curless, D. Salesin, and R. Szeliski, "A Bayesian Approach to Digital Matting," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition,* vol. 2, pp. 264-271, 2001.
[13] D. Comaniciu and P. Meer, "Mean Shift: A Robust Approach Toward Feature Space Analysis," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 24, no. 5, pp. 603-619, May 2002.
[14] A. Criminisi, G. Cross, A. Blake, and V. Kolmogorov, "Bilayer Segmentation of Live Video," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition,* vol. 1, pp. 53-60, 2006.
[15] Z. Dong, L. Jiang, G. Zhang, Q. Wang, and H. Bao, "Live Video Montage with a Rotating Camera," *Computer Graphics Forum,* vol. 28, no. 7, pp. 1745-1753, 2009.
[16] A.M. Elgammal, D. Harwood, and L.S. Davis, "Non-Parametric Model for Background Subtraction," *Proc. European Conf. Computer Vision,* vol. 2, pp. 751-767, 2000.
[17] R.I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision,* second ed. Cambridge Univ. Press, 2004.
[18] B.K.P. Horn and B.G. Schunck, "Determining Optical Flow," *Artificial Intelligence,* vol. 17, nos. 1-3, pp. 185-203, 1981.
[19] S. Khan and M. Shah, "Object Based Segmentation of Video Using Color, Motion and Spatial Information," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition,* vol. 2, pp. 746-751, 2001.
[20] V. Kolmogorov, A. Criminisi, A. Blake, G. Cross, and C. Rother, "Bi-Layer Segmentation of Binocular Stereo Video," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition,* vol. 2, pp. 407-414, 2005.
[21] M.P. Kumar, P.H.S. Torr, and A. Zisserman, "Learning Layered Motion Segmentation of Video," *Proc. IEEE Int'l Conf. Computer Vision,* pp. 33-40, 2005.
[22] V.S. Lempitsky, S. Roth, and C. Rother, "Fusionflow: Discrete-Continuous Optimization for Optical Flow Estimation," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition,* 2008.
[23] A. Levin, D. Lischinski, and Y. Weiss, "A Closed-Form Solution to Natural Image Matting," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 30, no. 2, pp. 228-242, Feb. 2008.
[24] Y. Li, J. Sun, and H.-Y. Shum, "Video Object Cut and Paste," *ACM Trans. Graphics,* vol. 24, no. 3, pp. 595-600, 2005.
[25] Y. Li, J. Sun, C.-K. Tang, and H.-Y. Shum, "Lazy Snapping," *ACM Trans. Graphics,* vol. 23, no. 3, pp. 303-308, 2004.
[26] C. Liu, W.T. Freeman, E.H. Adelson, and Y. Weiss, "Human-Assisted Motion Annotation," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition,* 2008.
[27] F. Liu and M. Gleicher, "Learning Color and Locality Cues for Moving Object Detection and Segmentation," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition,* 2009.
[28] D.G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *Int'l J. Computer Vision,* vol. 60, no. 2, pp. 91-110, 2004.
[29] B.D. Lucas and T. Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision," *Proc. Int'l Joint Conf. Artificial Intelligence,* pp. 674-679, 1981.
[30] W.R. Mark, L. McMillan, and G. Bishop, "Post-Rendering 3D Warping," *Proc. Symp. Interactive 3D Grahics,* vol. 180, pp. 7-16, 1997.
[31] M.M. Oliveira, B. Bowen, R. McKenna, and Y.-S. Chang, "Fast Digital Image Inpainting," *Proc. Int'l Conf. Visualization Imaging and Image Processing,* pp. 261-266, 2001.
[32] C. Rother, V. Kolmogorov, and A. Blake, "'Grabcut': Interactive Foreground Extraction Using Iterated Graph Cuts," *ACM Trans. Graphics,* vol. 23, no. 3, pp. 309-314, 2004.
[33] Y. Sheikh and M. Shah, "Bayesian Object Detection in Dynamic Scenes," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition,* vol. 1, pp. 74-79, 2005.
[34] J. Sun, Y. Li, and S.B. Kang, "Symmetric Stereo Matching for Occlusion Handling," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition,* vol. 2, pp. 399-406, 2005.
[35] J. Sun, W. Zhang, X. Tang, and H.-Y. Shum, "Background Cut," *Proc. European Conf. Computer Vision,* vol. 2, pp. 628-641, 2006.
[36] R. Szeliski and H.-Y. Shum, "Creating Full View Panoramic Image Mosaics and Environment Maps," *Proc. ACM SIGGRAPH,* pp. 251-258, 1997.
[37] J. Wang, P. Bhat, A. Colburn, M. Agrawala, and M.F. Cohen, "Interactive Video Cutout," *ACM Trans. Graphics,* vol. 24, no. 3, pp. 585-594, 2005.
[38] J. Wang and M.F. Cohen, "An Iterative Optimization Approach for Unified Image Segmentation and Matting," *Proc. IEEE Int'l Conf. Computer Vision,* pp. 936-943, 2005.
[39] J. Wang and M.F. Cohen, "Optimized Color Sampling for Robust Matting," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition,* 2007.
[40] A. Wedel, D. Cremers, T. Pock, and H. Bischof, "Structure- and Motion-Adaptive Regularization for High Accuracy Optic Flow," *Proc. IEEE Int'l Conf. Computer Vision,* 2009.
[41] Y. Weiss and E.H. Adelson, "A Unified Mixture Framework for Motion Segmentation: Incorporating Spatial Coherence and Estimating the Number of Models," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition,* pp. 321-326, 1996.
[42] L. Xu, J. Chen, and J. Jia, "A Segmentation Based Variational Model for Accurate Optical Flow Estimation," *Proc. European Conf. Computer Vision,* vol. 1, pp. 671-684, 2008.
[43] P. Yin, A. Criminisi, J. Winn, and I. Essa, "Tree-Based Classifiers for Bilayer Video Segmentation," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition,* 2007.
[44] G. Zhang, Z. Dong, J. Jia, L. Wan, T.-T. Wong, and H. Bao, "Refilming with Depth-Inferred Videos," *IEEE Trans. Visualization and Computer Graphics,* vol. 15, no. 5, pp. 828-840, Sept./Oct. 2009.
[45] G. Zhang, J. Jia, T.-T. Wong, and H. Bao, "Consistent Depth Maps Recovery from a Video Sequence," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 31, no. 6, pp. 974-988, June 2009.
[46] G. Zhang, J. Jia, W. Xiong, T.-T. Wong, P.-A. Heng, and H. Bao, "Moving Object Extraction with a Hand-Held Camera," *Proc. IEEE Int'l Conf. Computer Vision,* 2007.
[47] G. Zhang, X. Qin, W. Hua, T.-T. Wong, P.-A. Heng, and H. Bao, "Robust Metric Reconstruction from Challenging Video Sequences," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition,* 2007.



Guofeng Zhang received the BS and PhD degrees in computer science from Zhejiang University in 2003 and 2009, respectively. He is currently a postdoctoral researcher at the State Key Laboratory of CAD&CG, Zhejiang University. His research interests include camera tracking, 3D reconstruction, augmented reality, and video segmentation and editing. He is a member of the IEEE.

Jiaya Jia received the PhD degree in computer science from the Hong Kong University of Science and Technology in 2004. He joined the Department of Computer Science and Engineering, Chinese University of Hong Kong, in September 2004, where he is currently an associate professor. His research interests include vision geometry, image/video editing and enhancement, and motion estimation. He has served on the program committees of ICCV, CVPR, ECCV, and ACCV. He served as co-chair of the Interactive Computer Vision Workshop 2007 (in conjunction with ICCV '07). He is a senior member of the IEEE.

Wei Hua received the BS degree in biomedical engineering and the PhD degree in applied mathematics from Zhejiang University in 1996 and 2002, respectively. Currently, he is an associate professor of the State Key Laboratory of CAD&CG at Zhejiang University. His research interests include real-time simulation and rendering, virtual reality, and software engineering.

Hujun Bao received the BS and PhD degrees in applied mathematics from Zhejiang University in 1987 and 1993, respectively. Currently, he is a professor and the director of the State Key Laboratory of CAD&CG at Zhejiang University. His main research interest is computer graphics and computer vision, including real-time rendering technique, geometry computing, virtual reality, and 3D reconstruction.