# Occlusion-Aware Motion Layer Extraction under Large inter-Frame Motions

Feng Xu, and Qionghai Dai, *Senior Member, IEEE*

***Abstract*—Extracting motion layers from videos is an important task for video representation, analysis and compression. For videos with large inter-frame motions, motion layer extraction is challenging in two respects: the estimation of large disparity motions, and the awareness of large occluded regions. In this paper, we propose an effective method for motion layer extraction under large disparity motions. To robustly estimate large displacement motions, we have developed an efficient voting-based method which estimates planar homographies from sparse feature matches. To handle occlusions, we first integrate color and motion consistency into a Markov random field framework to achieve per-pixel assignment with occlusion detection. Then, we perform motion-color segmentation and an earth mover's distance-based comparison to determine motion labels for occluded pixels. Experimental results show that our proposed method achieves good performance in automatically extracting multiple moving objects under large disparity motions while maintaining a low computational cost.**

***Index Terms*—Earth mover's distance, Markov random field, motion segmentation, occlusion determination.**

## I. Introduction

MOTION layer extraction (also called motion segmentation) is an important research topic in computer vision. It aims to group pixels by their motions into layers. Motion layer extraction contains two major steps: (1) determining the layer descriptions, which includes the number of layers and the motion parameters for each layer, and (2) assigning each pixel to the correct layer whether the pixel is occluded or not.

Motion layer extraction has many applications. Object-based video compression relies on motion layer extraction to encode each moving object separately. Motion layer extraction can also be used when estimating depth maps for stereo to provide prior knowledge to disambiguate depth discontinuities along objects boundaries. Motion layer extraction can also be used in object recognition, video retrieval, and motion analysis.



This work was supported by the National Basic Research Project of China (Project Number (973 Program), No.2010CB731804.

Feng Xu and Qionghai Dai are with the TNList and Department of Automation, Tsinghua University, Beijing, China, 100086 (e-mail: xufeng2003@gmail.com; qhdai@mail.tsinghua.edu.cn).

In early works, Wang and Adelson [1] used optical flow to decompose images into motion layers, where each layer presented a smooth motion field. Weiss [2] extended this approach to handle flexible motion fields by using regularized radial basis functions (RBFs). Several other approaches formulate layer extraction as a maximum likelihood estimation (MLE) or maximum a posteriori probability (MAP) estimation, assuming different constraints and motion models [3-7].

Besides these methods, some researchers have focused on the problem of foreground extraction [8-11]: Zhang et al. [8] used appearance and structural consistency constraints to model the background. An estimated dense motion field and bi-layer segmentation results are iteratively refined. Huang et al. [9] used Markov random fields (MRFs) to integrate both spatial and temporal coherence to maintain continuity of segmentation. In their further work [10], they estimated a motion vector field and a foreground segmentation mask by maximizing the conditional joint probability density function of these two elements. In [11], Criminisi el al. fused motion, color and contrast cues to infer layers based on a motion vs. non-motion classifier. However, all these methods segment the frames into a background and a foreground layer only. When there are several objects in the scene with different motions, more layers are needed.

The methods described in [12-14] do not suffer the two-layer limitation. Ke and Kanade [12] expand seed regions into k-connected components. After enforcing a low-dimensional linear affine subspace constraint, they obtained initial layer models and assigned over-segmented regions to correct layers. Kumar et al. [13] obtained an initial estimate of their model using an efficient loopy belief propagation algorithm. Given the initial estimate, the shape of the segments, along with the layering, are learnt by minimizing an objective function using αβ-swap and α-expansion algorithms. Xiao and Shah [14] first established seed regions by using two-frame correspondences. After merging initial regions into several initial layers, they exploited the occlusion order constraint on multiple frames and used graph cuts to perform robust layer extraction. These methods achieve multi-layer motion segmentation results. However, these methods are based on several video frames (at least 3 to 5 frames), and not on two adjacent frames. As more frames are involved, these methods are hard to apply to real-time applications and stereo-based applications.

To overcome the discussed drawbacks of the above methods, we would like to segment more general scenes containing multiple moving objects, based solely on two adjacent frames.

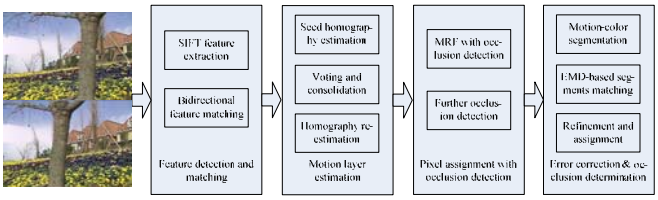



Fig. 1. Flowchart of the proposed framework.

Feature-based segmentation methods (of which a comprehensive survey can be found in [15]) are proposed to achieve this. These works use feature matching (also called correspondences) in adjacent frames to present motion information. Furthermore, by finding correspondences between frames with large disparity, Wills et al. [16] proposed a method which can process videos with large inter-frame motions. Their method achieves state-of-the-art performance in segmenting videos with large inter-frame motions. However, it still has two drawbacks. First, it uses a large number of correspondences (including outliers) to estimate the motion layers. As the number of correspondences is large (about 10%-20% of the number of pixels), the estimation is time consuming. Second, occluded pixels are not assigned to any motion layer. Hence, the segmentation result is not complete. Other works in the literature propose techniques to handle the occlusion problem. Xiao and Shah [14] first proposed occlusion ordering constraints to detect occlusion. However, these constraints are not valid when the captured object is too thin or moving too fast. Also, this method can only detect occlusion and not assign occluded pixels to motion layers.

In this paper, we propose an effective feature-based motion layer extraction method which overcomes the two drawbacks that exist in segmenting subsequent frames with large disparity motions. The main procedures of our proposed method are described as follows. First, by using the scale-invariant feature transform (SIFT) [17], feature points are extracted for adjacent frames and matched with a bi-directional feature matching method. As our correspondences are sparser (less than 0.1% of the number of pixels) than the previous work [16], lower computation complexity is guaranteed (see Section VII). Second, perspective projections are estimated by our proposed voting-based method. These model the camera motions in the video shot. Fast and accurate motion layer estimation is achieved as the relationship among correspondences is explored during the estimation. Third, we incorporate occlusion detection into an MRF framework to assign pixels to their correct motion layers or to the occlusion layer. In this procedure, color and motion consistency are efficiently integrated to perform complete occlusion detection. Finally, the detected occluded regions are over-segmented in our motion-color segmentation method, and an earth mover's distance [18] (EMD)-based method is developed to assign segments to the correct motion layers and correct erroneous assignments for visible pixels. As the EMD-based comparison integrates the global color information of a segment, accurate error correction and occlusion determination are realized. A flowchart of the proposed framework is shown in Fig. 1.

The rest of the paper is organized as follows: we first introduce our method for feature extraction and matching in Section II. In Section III, we detail how to estimate motion layers from sparse correspondences. Section IV describes how pixels are assigned to layers, and how occlusions are detected using an MRF framework. In Section V, we propose an EMD-based method for error correction and occlusion determination. In Section VI, we formulate the computational complexity of our method. Finally, experimental results are shown in Section VII and conclusions are drawn in Section VIII.

## II. Feature Extraction and Matching

As we use correspondences to represent motion information, accurate feature extraction and an effective feature descriptor are crucial for our application. In previous approaches [14,16], Harris corners and Förstner features are used to determine correspondences between adjacent frames. We adopt the SIFT method [17], which applies a difference of Gaussian function to identify feature points and calculates a descriptor for each feature point by using image gradients around a radius of the feature. SIFT has two appealing advantages. First, SIFT can locate feature points with sub-pixel accuracy. Second, as demonstrated in [19], the SIFT descriptor is robust, and maintains distinction under rotation, blurring, scale change, and illumination change.

Besides feature extraction, feature matching also plays an important role in motion extraction. We employ a bi-directional feature matching method based on Lowe's work [17]. We use the $L_2$ distance between two feature descriptors as the distance between the two features. To find the corresponding feature in frame $t$ for a feature in frame $t$+1, two conditions are tested: first, whether the matched feature has the smallest $L_2$; second, whether the distance between the two features is less than $T$ percent of the second closest feature by $L_2$. If these two conditions are satisfied, a preliminary match is established. Then, to perform bi-directional feature matching, we switch frames $t$ and $t$+1 and perform feature matching again. If two features match each other bi-directionally, then correspondence between these two features is established. Compared to Lowe's method, bi-directional feature matching further discards some incorrect matches. It also guarantees that matching results in opposite directions are consistent. This property will be used in the following approach to occlusion detection.

In our bi-directional feature matching, if $T$ is set to a large value (i.e. greater than 80), many correspondences are established; however, many may be outliers. If $T$ is set to a small value (i.e. smaller than 60), the number of correspondence decreases dramatically, but nearly all are inliers. As a consequence, by the inclusion of parameter $T$ in our bi-directional feature matching, we can achieve a trade-off between the number of feature-point pairs and the accuracy of the correspondences. In our application, $T$ is set to a conservative value of 60. Notice that, in our method, the

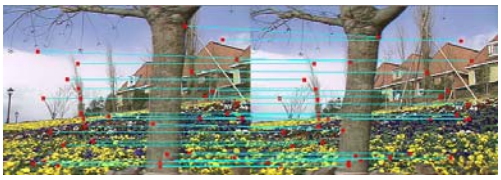


Fig. 2. Correspondences for frame 1 and frame 15 of *Garden* (*T*=60)

performance is not very sensitive to the parameter *T*. In all our experiments, setting *T* to a value between 55 and 67 will result in very similar performance. Fig. 2 presents an example of a feature matching results. Due to the fast camera panning, frames 1 and 15 of the *Garden* test sequence exhibit a large disparity motion. Even in such a case, our method establishes accurate correspondences. The matched features are quite sparse for this value of *T*. Consequently, our following computations are fast. SIFT is used to extract features; however, SURF [20] is also applicable. While SURF is faster than SIFT, the resulting segmentation accuracy varies more widely as parameter *T* changes. Given this choice, we use SIFT to preserve robustness to different scenes and maintain segmentation accuracy.

## III. Voting-based Motion Layer Estimation

In the previous section, we established sparse correspondences between adjacent frames. In this section, given these correspondences, we develop an efficient voting-based method to estimate motion layers for all moving objects and the background respectively. In the literature, tensor voting has previously been used for motion segmentation. In these methods, pixels are first represented by tensor points in 4D [21] or 5D [22] spaces. Then, in the voting process, all points collect votes from their neighbors. Tensor voting-based methods need to calculate the motion of each pixel, and voting is performed between tensor points. As our method is based on sparse correspondences, pixels vote to seed and decide homographies. Thus, our method focuses on sparse correspondences and has a different voting mechanism. The performance of these two methods is compared in Section VII. Another method which produces comparable results is the RANSAC method [23] used by Wills et al. [16]. Compared to RANSAC, the contribution of our method lies in reducing the computational cost significantly. This will be detailed in Section VI which will analyze the time complexity of the two methods.

### *A. Planar Homography*

A homography describes the relationship between projections of a 3D plane in two images. In particular, for two adjacent video frames exhibiting motion, points on the plane map to different image coordinates. The positions are constrained by a homography as follows:

$$\vec{x} = H \cdot \vec{x}' \tag{1}$$

where $\vec{x}$ and $\vec{x}'$ are the homogeneous coordinates of the points in the two frames and $H$ is the homography which constraints all points to the plane. As $H$ is a matrix which contains eight degrees of freedom (the matrix totals nine entries, but one entry defines an ambiguous scale factor), we need at least four pairs of corresponding points on the plane surface to determine $H$. In this work, we assume that each scene object has at least one surface plane that is captured in both images, or that the object can be treated as a plane when it is far from the camera. In these situations, the 3D motion caused by the object and the camera can be modeled as a homography.

### *B. Homography Estimation*

Though the correspondences between adjacent frames are established, we do not know how many motion layers exist in the scene and which correspondences belong to which layer. These are the major barriers to estimate homographies for all layers. We propose a voting mechanism to overcome these barriers as follows.

*1) Seed Homography Estimation*: We observe that features belonging to the same object have a strong motion relationship with each other. This property is used to guide the estimation of homographies. In our method, we use one correspondence and its five nearest neighbors in the 2D motion space to estimate a seed homography (seed *H*). As the six features have a high probability of belonging to one object, the seed *H* may be close to the true *H*. If we used only four correspondences, then three collinear points in either frame would result in a configuration with no unique solution. As our feature matching method (see Section II) generates a set of sparse correspondences, calculating seed *H*s for all correspondences is not computationally expensive. In [16], the RANSAC method randomly selects correspondences to perform the estimation. Therefore, features from different layers are likely to be sampled to estimate one *H*. Although incorrect *H*s are pruned in following steps, much computation has already taken place. Our proposed method avoids this incorrect estimation and saves computation.

*2) Voting and Consolidation:* Though the above seed estimation avoids incorrect estimation to a certain extent, some incorrect seed *H*s may still emerge when the six correspondences do not belong to the same layer. Also, some seed *H*s may redundantly present one layer if they are estimated from features on the same layer. Therefore, the estimated seed *H*s need to be further refined. To this end, we first test whether a correspondence satisfies *H* as follows:

$$\left\|\vec{x}_k - H_l \cdot \vec{x}'_k\right\|_2 \begin{cases} \le \tau & C_k \text{ satisfies } H_l \\ > \tau & C_k \text{ does not satisfy } H_l \end{cases} \tag{2}$$

where $\vec{x}_k$ and $\vec{x}'_k$ are the homogeneous coordinates of a correspondence $C_k$ in two frames. If $C_k$ satisfies homograph $H_l$, we give one vote to $H_l$ from $C_k$. The parameter $\tau$ is an inlier threshold which specifies the tolerated noise level. This parameter depends on the quality of the input images and is set to a large value for noisy images (7 for the *Car* sequence in our experiments) or a small value for other images (3 for other sequences in our experiments). After testing all correspondences for all seed *H*s, the ballot box for each *H* contains a certain number of votes. Looking further at these

votes, two properties can be observed: first, only one vote from a certain correspondence can be found in any one ballot box; second, one correspondence may vote in many ballot boxes if it satisfies the associated *H*s. After the voting procedure, we begin to consolidate *H*s using the method introduced in [16]. If two *H*s contain many votes from the same correspondences (more than 75% of the total number), we consolidate the two ballot boxes into one. After consolidation, the number of ballot boxes approaches the number of real motion layers in the scene.

*3) Homography Re-estimation:* We can now identify which correspondence belongs to which ballot box. We achieve this by counting the votes of all ballot boxes. For one correspondence, if a ballot box has the maximum number of votes, we assume it belongs to this ballot box. If one correspondence votes less than 5 times, it is considered an outlier and is deleted from the correspondence list.

Next, we want to remove outlying ballot boxes. If a ballot box has a small number of correspondences (again less than 5), the associated *H* may be estimated from features on different layers. In this case, the ballot box should be pruned from the result. After the pruning procedure, we create a number of motion layers equal to the number of remaining ballot boxes. For each layer, we re-estimate *H* from all its correspondences. The new obtained *H*s are used to describe the motion layers and will be used in the following occlusion detection procedure.

## IV. Pixel Assignment with Occlusion Detection

In the previous section, we explained how homographies are determined for motion layers. The aim of this section is to assign all pixels to their correct motion layers. Previous works [14,16] achieve this based on the assumption that the appearance of objects remains the same between images. However, this assumption is not valid for occluded pixels as they only appear in one of the two frames. To handle this problem, our method integrates two occlusion detection techniques into an MRF framework for joint pixel assignment.

The first technique is based on the color constancy assumption. A constant error threshold is manually predetermined for the occlusion layer. If the error in assigning a pixel to each estimated motion layer is higher than a threshold, the pixel is treated as an occluded pixel. The second technique is based on the motion constancy assumption. We first warp pixels forward from frame *t* to frame *t*+1, then warp frame *t*+1 back again. If one pixel cannot be warped back to its original coordinates after the bi-directional warping, it is deemed to be an occluded pixel. In our method, we integrate these two techniques to achieve complete occlusion detection. Furthermore, as we add an occlusion layer to the MRF structure, continual occlusion detection and continual motion layer assignment are performed simultaneously.

### *A. MRF Framework*

The problem of assigning each pixel to the correct motion layer or the occlusion layer can be formulated as determining a function $l$ that maps each pixel to a unique motion label from the label set $L=\{1,...,m,m+1\}$, where $1,...,m$ represent all motion layers and $m+1$ represents the occlusion layer. We describe function $l$ for frame $t$ by minimizing the following energy function:

$$E(l,I^t,I^{t+1}) = E_{data}(l,I^t,I^{t+1}) + \lambda E_{smooth}(l,I^t) \quad (3)$$

where $I^t$ is the intensity of frame $t$. This energy function has two terms with a penalizing factor $\lambda$ between them. The data term addresses the reconstruction error by the following formulation:

$$E_{data}(l,I^t,I^{t+1}) = \sum_i \begin{cases} [I^t(i) - I^{t+1}(M(l(i),i))]^2 & \text{if } l \in \{1,...,m\} \\ d & \text{if } l = m+1 \end{cases} \quad (4)$$

where $I^t(i)$ denotes the intensity of pixel $i$ in frame $t$ and $M(l(i),i)$ returns the new label for pixel $i$ in frame $t+1$ under the influence of motion $l(i)$. $d$ is a constant parameter modeling the reconstruction error for occluded pixels. If $l$ is in the range $\{1,...,m\}$, the difference in intensity is used to model the reconstruction error. However, if $l$ equals to $m+1$, a constant parameter $d$ is used. For one pixel, if the reconstruction error caused by assigning it to any real motion layer is greater than the constant $d$, the pixel is given the occlusion label. Thus, if $d$ is set too large, some occluded pixels may be incorrectly given a motion label. However, if $d$ is set too small, visible pixels may be assigned to the occlusion layer. The result may be sensitive to $d$, but our method can overcome this drawback. We include a special step (illustrated in Section V) to handle detected occluded pixels, and so we choose a small value of $d$ (0.12 after normalizing pixel intensity to [0, 1]). Even though some visible pixels are still incorrectly given the occlusion label, they will be reassigned with correct motion labels in the final result.

Following [16], the second term of the energy function is the smoothness prior:

$$E_{smooth}(l,I^t) = \sum_i \sum_{j \in N(i)} s_{ij}(I^t)\left[1 - \delta_{l(i)l(j)}\right]. \quad (5)$$

Here $s_{ij}(I^t)$ is the similarity between two pixels $i$ and $j$ in frame $t$. $\delta$ equals 1 when its arguments are equal; otherwise, it equals 0. $N(i)$ represents the neighborhood of pixel $i$, which contains all pixels in an image block centered at $i$. We use a bilateral filter [24] to define the similarity between pixels. A bilateral filter combines geometric proximity and photometric similarity between two pixels. To use this property in our measurement of similarity, we set the weight in our bilateral filter as the similarity between two pixels. This similarity is defined as follows:

$$s_{ij}\left(I^t\right) = \Phi_{ij}^X(I^t)\Phi_{ij}^C(I^t). \quad (6)$$

The first term $\Phi_{ij}^X(I^t)$ is the closeness function and the second term $\Phi_{ij}^C(I^t)$ is the photometric similarity function. In our case, these two functions are Gaussian in the Euclidean distance between their arguments [24]. More specifically, $\Phi_{ij}^X(I^t)$ has the following formulation:

$$\Phi_{ij}^X(I^t) = \exp\left[-\frac{D(i,j)^2}{\sigma_X}\right] \quad (7)$$

where $D(i,j)$ is the Euclidean distance between pixel $i$ and $j$, and $\sigma_X$ is the bandwidth. The formulation of $\Phi_{ij}^C(I^t)$ is analogous:

$$\Phi_{ij}^C(I^t) = \exp\left[-\frac{(I^t(i)-I^t(j))^2}{\sigma_C}\right] \quad (8)$$

where $I^t(i)$ is the intensity value of pixel $i$ and $\sigma_C$ is the bandwidth. Notice that $\sigma_X$ and $\sigma_C$ are set to achieve the desired combination of pixel locations and pixel values respectively ($\sigma_X$=2 and $\sigma_C$=1 in our experiments). In equation (5), $s_{ij}(I^t)$ penalizes the discontinuity assignment of motion. As motion segmentation results for rigid objects should be piecewise constant, this definition is reasonable. From the formulation of the energy function (3), we see that the data term ensures the assignment is consistent with the motion and occlusion in the scene, and the smoothness term guarantees the piecewise constant property.

The problem of minimizing this energy function can be treated as a metric labeling problem. As demonstrated by Kleinberg and Tardos [25], this kind of problems corresponds to finding the maximum a posteriori labeling of a class of Markov random field. α-β swap is used to solve this problem in Boykov, Veksler and Zabih's (BVZ's) method [26]. As our smoothness term is not metric, we cannot use α-expansion to further speed up this process. BVZ's method can find a solution with an error at most twice that of the optimal solution in polynomial time. Each iteration of the algorithm constructs a graph, and finds the minimum cut partition in the graph to assign new labels to pixels.

### *B. Further Occlusion Detection*

The previous procedure assigns occluded pixels with the occlusion label based on color consistency. However, motion consistency can also be used to detect occlusion [16]. The proposed bi-directional feature matching guarantees that obtained feature motions are consistent in opposite directions. If the motions of pixels do not have this property, then the pixels are likely to occur in only one frame and should be assigned to the occlusion layer. Based on this observation, we run the BVZ algorithm twice for adjacent frames. The first round assigns motions from pixels in frame $t$ to frame $t$+1 (to describe the forward motion), and the second round assigns motions from pixels in frame $t$+1 to frame $t$ (to describe the backward motion). For pixel $i$ in frame $t$, if the corresponding pixel $i'$ in frame $t$+1 obtained by the forward motion maps back to $i$ or its neighbors by the backward motion, pixel $i$ is considered a visible pixel; otherwise, pixel $i$ is an occluded pixel. This is demonstrated by the following formulation:

$$i\begin{cases} \text{is a visible pixel} & \text{if } M_B\left(l'(i'),i'\right)=i \\ \text{is an occluded pixel} & \text{if } M_B\left(l'(i'),i'\right)\neq i \end{cases} \quad (9)$$

where

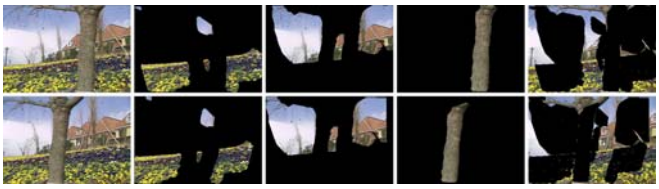


Fig. 3. Pixel assignment results for frames 1 and 15 of the *Garden* sequence. Column 1: two original frames, Columns 2-4: three motion layers, Column 5: occlusion layer.

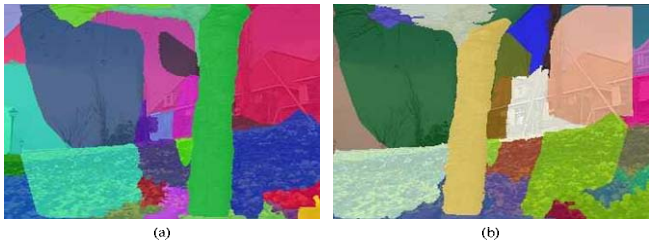
$$i' = M_F\left(l(i),i\right). \quad (10)$$

Here, the unprimed symbols refer to frame $t$ and the primed symbols refer to frame $t$+1, while $M$ is the same as in equation (4) and the subscripts $F$ and $B$ denote the forward and backward motion. When new occluded pixels are detected, we change their motion labels to $m$+1 to represent occlusion. An example of the pixel assignment is shown in Fig. 3. We can see that the assignment of pixels is consistent with the motion of the video, and is piecewise constant. However, there are still two drawbacks. First, some incorrect assignments still exist in the results. For example, the sky region is assigned to the "garden" layer (Column 2 in Fig.3). This is because the data term in equation (4) is based on the intensity difference of two pixels, which leads to poor performance in regions with little texture. Second, the occlusion layer incorrectly contains some visible pixels, which is caused by parameter $d$ having a small value. In the following section, based on the color information, incorrect motion labels of visible regions will be corrected and occluded regions will be assigned with correct motion labels.

## V. Error Correction and Occlusion Determination

In the previous section, we achieved smooth pixel assignment as well as occlusion detection. However, motion labels for occluded pixels are still unknown. In this section, we determine the motion labels for occluded pixels (occlusion determination) to perform complete motion layer extraction. We also refine motions for visible pixels (called error correction). These are jointly achieved by our EMD-based method.

First, we propose a motion-color segmentation method to segment each frame, in which visible regions and occluded regions are segmented respectively. For visible regions, segmentation is based on the motion labels of pixels. Neighboring pixels with the same motion label are grouped into one segment and vice-versa. The formed segments are called visible segments. For occluded regions, an over-segmentation is performed by using pixel color information to generate occluded segments. Following motion-color segmentation, we need to assign each pixel to its correct motion layer for each segment. Second, we compute the distance (EMD, which will be explained in part B) between segments by comparing statistical quantities. As these quantities are calculated on the intensities and gradients of all pixels in a segment, they carry more information about the segment. Third, two tasks are jointly performed based on the distances between segments. One is to refine the motion labels for visible segments, and the



(a) (b)

Fig. 4. Motion-color segmentation results for frames 1 and 15 of the *Garden* sequence.

other is to assign occluded segments with correct motion labels. An adjacency constraint is used in the two tasks. Following this, the segments are merged according to their labels to form the final motion segmentation result.

### *A. Motion-color segmentation*

To achieve motion-color segmentation, we use a graph-based method [27]. This method has several advantages for our application. First, this method needs only similarity between pixels to achieve segmentation. Therefore, by properly defining the similarity, we can perform color segmentation in occluded regions and use motion labels to segment visible regions. Second, this method can guarantee that pixels from different moving objects will not be assigned to one layer. Third, this method is efficient with computational complexity $O(n \log n)$, where $n$ is the number of pixels in the image. This allows the algorithm to be used in real-time applications. Finally, this method captures the perceptually important non-local properties of an image, and so objects with a high variability in intensity can also be extracted.

The similarity (ranging from 0 to 1) between neighboring pixels (on a 4-neighbor system) is obtained by the following rules:

**Rule 1.** If two pixels are both occluded, their similarity will be $s_{ij}(I^t)$ (Equation (6)).

**Rule 2.** If two visible pixels own the same motion label, their similarity will be 1; otherwise, it will be 0.

**Rule 3.** If one pixel is occluded while the other is visible, their similarity will be 0.

Due to this assignment of similarity, the segmentation result is determined by motion labels in visible regions and by color in occluded regions. The formed segments can be classified into two groups: one contains visible segments which have known motion labels (which may be incorrect), and the other contains occluded segments whose motions will be estimated in the following steps. Fig. 4 illustrates the segmentation result. We can see that no pixels from different motion layers are segmented into one segment. Determining the motion labels for all segments is equal to determining the motion labels for all pixels.

### *B. EMD-based Segment Comparison*

In this subsection, we will define a distance between two segments to describe their textural similarity. This is achieved by the proposed EMD-based method. Earth-mover's distance is a mathematical measure of the distance between two distributions or two histograms and can be successfully used to measure the similarity between two images in image and video retrieval (refer to [28-30] for details of the algorithm). EMD achieves good performance in distance measurement for two reasons: first, it relates neighbor bins in a histogram, which the Euclidean distance cannot guarantee; second, EMD can automatically explore the neighborhood relationship by moving the "earth" in a bin to its correct "cave" with the global minimum cost. In our application, we use EMD to measure the distance between two image segments. We calculate three histograms for each segment: one based on color, and two based on horizontal and vertical gradients respectively. By integrating the EMDs of these three distributions, we obtain the distance of two segments. In particular, we calculate two distance matrices. One stores the distances between every pair of visible segments, which we call the *V-V Matrix*. The other stores the distances from every occluded segment to every visible segment, which we call the *O-V Matrix*.

### *C. Refinement and Assignment*

The refinement for one visible segment is achieved by comparing the minimum intra-class distance and the minimum inter-class distance. For the visible segment $i$, the minimum intra-class distance and the minimum inter-class distance are defined as follows:

$$D_{\text{intra}}^{\min}(v_i) = \min_{v_j \in \Omega_i} \left[ D(v_i, v_j) \right], \quad \Omega_i = \{ v_k \mid v_k \text{ has the same motion label with } v_i \} \tag{11}$$

$$D_{\text{inter}}^{\min}(v_i) = \min_{v_j \in \Psi_i} \left[ D(v_i, v_j) \right], \quad \Psi_i = \{ v_k \mid v_k \text{ has a different motion label with } v_i \} \tag{12}$$

where $v_i$ denotes the visible segment $i$ and $D(v_i, v_j)$ is the distance between visible segments $i$ and $j$. These two distances are found in the *V-V Matrix*. If the ratio of $D_{\text{intra}}^{\min}(v_i)$ to $D_{\text{inter}}^{\min}(v_i)$ is greater than a threshold (1.7 in our experiments), the motion label of the segment $i$ is thought to be incorrect, and is replaced by the motion label of the segment which has the minimum inter-class distance with segment $i$.

After refining the motions for visible segments, the assignment for occluded segments is performed. If an occluded segment belongs to an object with a visible segment, the motion label of the visible segment is assigned to the occluded segment. Thus, our task is to find a correct visible segment for each occluded segment. Closer inspection reveals that the desired visible segment, which at first glance should be the most similar visible segment, is actually restricted by an adjacency constraint: the occluded segment should be connected to the desired visible segment. If not, the segments are likely to be on different objects. In our method, we first assign each occluded segment with the motion label of its most similar visible segment (by searching the elements in the *O-V* Matrix). Then, to impose the adjacency constraint upon the algorithm, the following test is performed. For one occluded segment, we test whether it is adjacent to the most similar visible segment or has

a path composed of segments with the same motion to the most similar segment. If neither is true, we reassign the occluded segment with the motion label of the second similar segment. As changing the motion label for one occluded segment may affect the test of others, we iteratively perform the test for all occluded segments until convergence. Videos are synthesized to illustrate the procedures of the error correction and occlusion determination. These are available from http://media.au.tsinghua.ed u.cn/xufeng.jsp.

## VI. Computational Complexity

Like Sharma and Paliwal's work [31], we analyze the computational complexity of our method in this section. Let $n$ be the number of pixels in one frame, $f$ be the number of SIFT features detected in one frame, $c$ be the number of correspondences, $y$ be the number of layers and $g$ be the number of segments after motion-color segmentation. The estimated computational complexity is shown in Table I. Notice that $f$, $c$, $y$ and $g$ are always small numbers in our application, so the computational complexity of our whole method can be estimated by $O(n\ logn)$. This complexity is attributed to the motion-color segmentation step. However, for common image resolutions, motion-color segmentation only takes a small part of the total computation time (see Section VII.B). In most situations, the most time consuming step is motion layer estimation (similar to in Wills' method). In our method, we propose a voting-based method which dramatically reduces the time complexity of this step. Before showing the experimental comparison, we theoretically explain our improvement as follows.

In terms of omega, the complexity of the proposed voting-based method and the RANSAC method used in [16] is formulated in (13):

$$Complexity = O\left(T_{iter}\left(C_{estimate}(k) + c \cdot C_{fitting}\right)\right) \tag{13}$$

where $T_{iter}$ is the number of iterations, $k$ is the number of correspondences used in computing the parameters of a homography, $C_{estimate}(k)$ is the cost of this computation, $c$ is the number of correspondences in one frame and $C_{fitting}$ is the cost associated with computing the fit of a single correspondence.

In our voting-based method, $T_{iter}$ equals $c$ as illustrated in Section III.B. For RANSAC, $T_{iter}$ has the following formulation:

$$T_{iter} = \frac{\log \varepsilon}{\log\left(1 - \left(\frac{c_I}{c}\right)^k\right)} \tag{14}$$

where $\varepsilon$ is a probability threshold (often called an alarm rate, which is the probability of failure of RANSAC), and $c_I$ is the number of correspondences belonging to layer $I$. Taking the *Garden* sequence as an example, in our voting-based method, $c$ equals 34, so $T_{iter}$ equals 34. In RANSAC, as there are three layers in the scene, for one layer, $c_I / c$ is approximately 1/3.

TABLE I Computational Complexity of the Algorithm

| Steps of the proposed method | | Complexity |
|---|---|---|
| Feature detection and matching | Feature detection | $O(n)$ |
| | Feature matching | $O(f^2)$ |
| Motion layer estimation | Seed $H$ estimation | $O(c)$ |
| | Voting and consolidation | $O(c^2)$ |
| | $H$ re-estimation | $O(y)$ |
| Pixel assignment with occlusion detection | MRF with occlusion detection | $O(n)$ |
| | Further occlusion detection | $O(n)$ |
| Error correction & occlusion determination | Motion color segmentation | $O(n\ logn)$ |
| | EMD based segments comparison | $O(n+g^2)$ |
| | Refinement and Assignment | $O(g)$ |
| Total estimated | | $O(n\ logn+ f^2+ c^2+ y + g^2)$ |

$\varepsilon$ is set to 0.01 to guarantee the success of the RANSAC method. As $k$ equals 4 in the homography estimation, we obtain $T_{iter}$ equal to 370. Notice that $C_{estimate}(6)$ in our method is close to $C_{estimate}(4)$ in RANSAC, so $\left(C_{estimate}(k) + c \cdot C_{fitting}\right)$ is similar between the two methods and the total cost of our method is 34/370 of the cost of the RANSAC method. In RANSAC, the time complexity is $O(c)$ for a nonzero $\varepsilon$ and a fixed $c_I / c$. However, for multi-layer estimation, $c_I / c$ is always small, and so $T_{iter}$ is usually not a small number. On the other hand, the complexity of our voting based method is $O(c^2)$. When handling sparse correspondences, $T_{iter}$ is always much smaller than that of the RANSAC method. Furthermore, we use sparse correspondences to estimate motion layers. RANSAC processes many more correspondences ($c$=20939 in Wills' method). As $c \cdot C_{fitting}$ becomes larger, the total cost becomes much larger than our method.

## VII. Experimental Results

We apply our method to videos containing different types of object classes (such as vehicles, humans and nature scenes), foreground motions (translation and similarity transforms), camera motions (static and translating) and occlusions (object-caused and camera-caused). In Section VII.A, we compare our segmentation results with state-of-the-art methods. Computational cost is analyzed in Section VII.B. In Section VII.C, we demonstrate the performance of specific procedures in our proposed method.

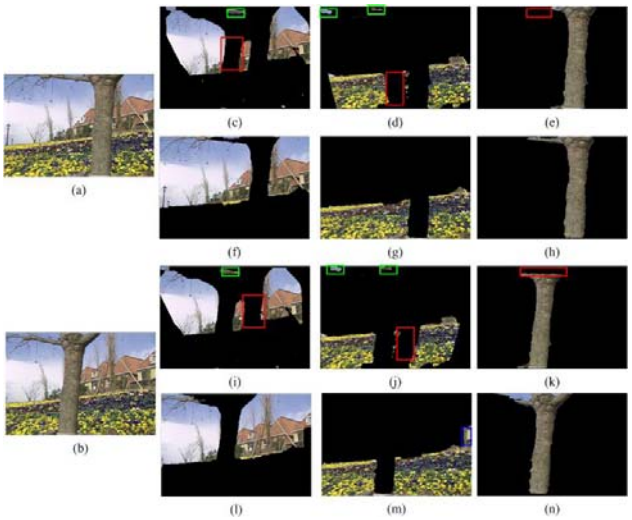



Fig. 5. Segmentation results for *Garden*: (a-b) Frames 1 and 15 of the sequence. (c-e) Three layers of Wills' method for frame 1. (f-h) Three layers of our method for frame 1. (i-k) Three layers of Wills' method for frame 15. (l-n) Three layers of our method for frame 15.

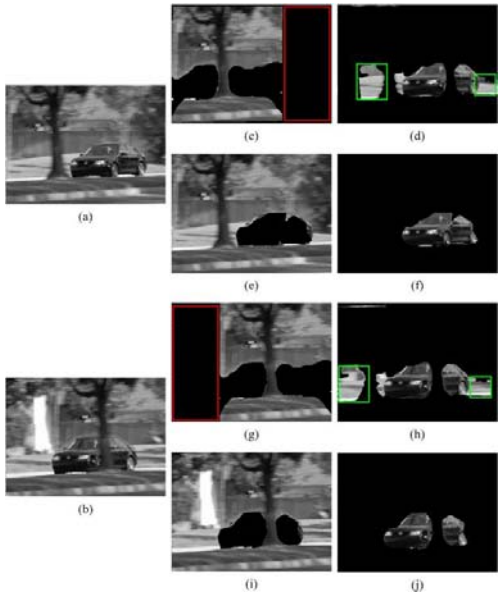



Fig. 6. Segmentation results for two frames of *Car*. (a-b) Two frames of the sequence. (c-d) Two layers of Wills' method for the first frame. (e-f) Two layers of our method for the first frame. (g-h) Two layers of Wills' method for the second frame. (i-j) Two layers of our method for the second frame.

### *A. Segmentation Result*

As our method aims to segment moving objects from two frames with large disparity motions, we compare it to the method proposed by Wills et al. [16], which has the same objective and is the current state-of-the-art method.

We first use two frames of the *Garden* sequence in our experiments. To test the performance under large disparity motions, we choose frames 1 and 15. In this sequence, the camera moves horizontally while the scene remains unchanged. The tree is the nearest object in the scene, and has the largest disparity caused by camera motion. The garden has the second largest disparity. Since the sky and the house are far from the camera, they form the third motion layer in the scene. In Fig. 5, some occluded regions (highlighted by red rectangles) are not assigned to any motion labels by Wills' method, but are correctly segmented and labeled by our method. Our method can segment the whole frame rather than only visible pixels. In addition, as the MRF does not perform well in untextured regions, Wills' results contain some incorrect segments (highlighted by green rectangles). Our proposed method does not suffer from this problem due to our EMD-based method. Our proposed method also has some limitations. If the over-segmented regions of one motion layer have different appearances, the EMD-based method cannot always assign them to their correct layers (marked by blue rectangles). Note that the following figures in this subsection use these rectangles to mark regions of interest.

The *Car* sequence was used in Wills' work [16] to evaluate performance. For comparison, the same two frames are used to test our method. In the first frame, the car is to the side of the tree. In the second frame, the tree occludes the middle of the car. The low quality of the images and the occlusion are the major challenges for this segmentation. As shown in Fig. 6, Wills' method cannot correctly distinguish the background and the moving car. This is mainly caused by the low quality of the frames. In these frames, mismatching does not cause salient penalization in the energy function, and so this is a difficult problem for the MRF framework. However, in our method, since the EMD-based method reassigns visible segments to correct motion layers, this difficulty can be overcome to a certain extent. This experiment shows that the EMD-based method can not only handle the occlusion problem, but also effectively refine the results in visible regions. The disadvantage of the proposed method lies in the object boundary. As the quality of the images is poor, this drawback is apparent in the result.

A third comparison is performed on two frames of the *Akko&Kayo* sequence. Two people walk towards each other, one holding a balloon and the other holding a box with two holes through which the background is visible. It is difficult to correctly segment these two small holes and assign them to the background layer. As shown in Fig. 7, Wills' method fails in these regions due to the lack of error correction and occlusion handling mechanisms, while our proposed algorithm handles this problem correctly. However, our method also introduces some artifacts. The balloon in the first frame is not assigned to the correct motion layer because it is occluded in the second frame (denoted by the blue rectangle). Its appearance is totally different from the other segments which belong to the same layer.

Two frames of the *Jeep* sequence are used in our fourth comparison. In this sequence, a jeep is traveling in a desert while the camera pans. The jeep is near to the camera in both frames and so the jeep cannot be treated as a plane. As the planarity assumption is violated, our performance is limited. From Fig. 8, we see that Wills' method presents many incorrectly segmented regions (denoted by green and red rectangles) while our proposed method achieves better performance (though the boundary is not that clear). This

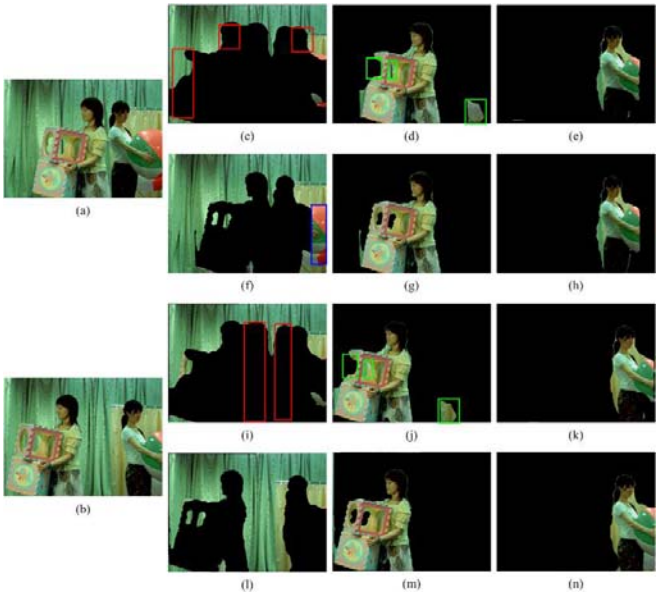


Fig. 7. Segmentation results for *Akko&Kayo*: (a-b) Frames 48 and 58 of the sequence. (c-e) Three layers of Wills' method for frame 48. (f-h) Three layers of our method for frame 48. (i-k) Three layers of Wills' method for frame 58. (l-n) Three layers of our method for frame 58.

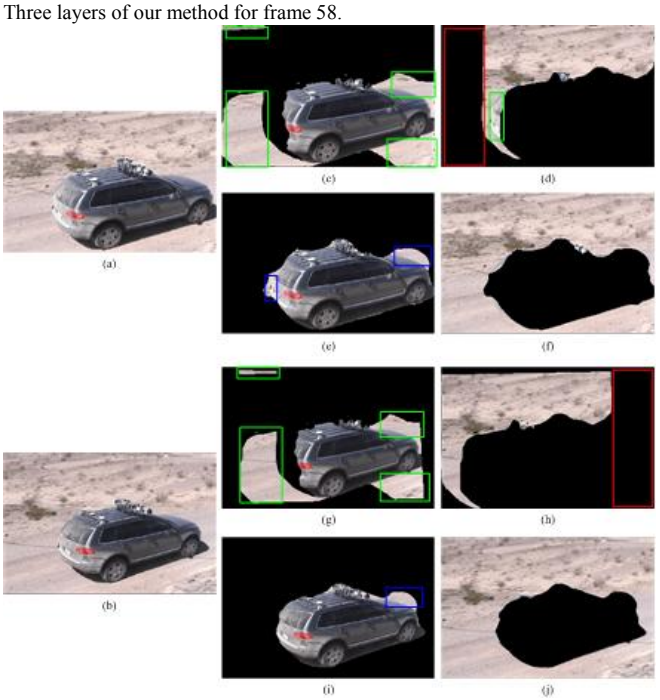


Fig. 8. Segmentation results for *Jeep*: (a-b) Frames 193 and 199 of the sequence. (c-d) Two layers of Wills' method for frame 193. (e-f) Two layers of our method for frame 193. (g-h) Two layers of Wills' method for frame 199. (i-j) Two layers of our method for frame 199.

improvement is achieved by the EMD-based method which propagates correct labels between visible pixels. As a consequence, to a certain extent, the EMD-based comparison can extend our method to situations which do not satisfy the planarity assumption.

In Table II, segmentation errors are calculated and compared for all the above experiments. The segmentation error for one frame is defined as the number of incorrectly segmented pixels divided by the total number of pixels. The ground truth is

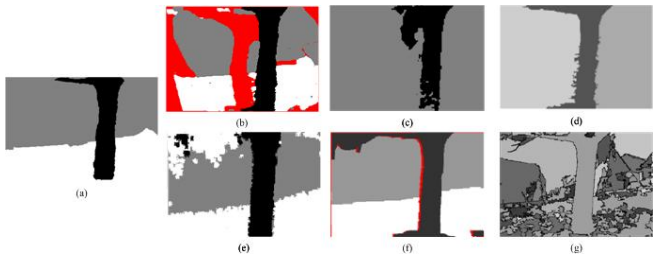


Fig. 9. Comparison on the *Garden* sequence with six other methods. (a) Result of our method. (b) Result of Wills et al. [16]. (c) Result of Wong and Spetsakis [32]. (d) Result of Min and Medioni [22]. (e) Result of Babu et al. [33]. (f) Result of Xiao and Shah [14]. (g) Result of Brendel and Todorovic[34]. Note: (c-g) are reproduced from papers [32], [22], [33], [14] and [34] respectively.

obtained manually. From Table II, we can see that our proposed method outperforms Wills' method. The error in our method is only 12.1% of the error in Wills' method on average.

From the above comparison, we see that the proposed method achieves more accurate segmentation than Wills' method. This is mainly attributed to two factors. The first factor is the procedure of pixel assignment, which segments visible pixels and simultaneously detects occluded pixels with piecewise smoothness. The second factor is the EMD-based method which not only determines the motion labels for occluded pixels but also fixes incorrect assignments in visible pixels.

As mentioned in the introduction, many other methods exist for motion segmentation. Although these methods do not focus on large disparity motions, they attempt to solve similar problems to ours. Thus, we choose five state-of-the-art methods to compare with Wills' method and our method. Notice that all their results are achieved by using either more than two frames, or two frames with small inter-frame motions. This is a different problem from that which we attempt to solve. Thus, we only present a qualitative comparison. Fig. 9 shows a comparison on the standard *Garden* sequence. The color red denotes occluded pixels. We can clearly see that our result provides more precise boundaries than the results shown in (b - e). The result shown in (f) achieves a good segmentation result. However, this method requires a video clip as input. It detects occluded pixels but does not assign occluded pixels to any motion layer. As a consequence, if input frames contain a large disparity, such as the frames used in our experiment, there will be large red regions in their result. The result shown in (g) successfully delineates the entire tree. However, it fails to track the textured surface of the flowers as a whole, because the mean shift algorithm used in this technique is very unstable in that area.

### *B. Computational Cost*

Computational cost is shown in Table III, where *FN* denotes the number of correspondences (the ratio of this number to the total number of pixels is shown in brackets), *CT_F*, *CT _M*, *CT_P* and *CT_EO* are the *cputimes* for the four steps of our method (feature extraction and matching, motion layer estimation, pixel assignment and error correction, and occlusion determination). *CT_W* is the *cputime* for the whole algorithm. All experiments for both methods are performed on

TABLE II
COMPARISON OF THE SEGMENTATION ERRORS BETWEEN WILLS' METHOD AND THE PROPOSED METHOD

| | *Garden* (352×288) | | *Car* (320×240) | | *Akko&Kayo* (384×288) | | *Jeep* (288×192) | |
|---|---|---|---|---|---|---|---|---|
| | First frame | Second frame | First frame | Second frame | First frame | Second frame | First frame | Second frame |
| Wills' method | 0.2756 | 0.3339 | 0.4092 | 0.377 | 0.2921 | 0.2952 | 0.3778 | 0.4347 |
| Our method | 0.0288 | 0.0468 | 0.0189 | 0.0182 | 0.0912 | 0.0482 | 0.0384 | 0.0224 |

TABLE III
COMPARISON OF THE COMPLEXITY PERFORMANCES BETWEEN WILLS' METHOD AND THE PROPOSED METHOD

| | *Garden* (352×288) | | *Car* (320×240) | | *Akko&Kayo* (384×288) | | *Jeep* (288×192) | |
|---|---|---|---|---|---|---|---|---|
| | Wills' method | Our method | Wills' method | Our method | Wills' method | Our method | Wills' method | Our method |
| *FN* | 20939(20.7%) | 34(0.034%) | 6426(8.37%) | 52(0.068%) | 11968(10.8%) | 64(0.058%) | 12159(22.0%) | 53(0.096%) |
| *CT_F* | 466.0s | 0.01563s | 50.75s | 0.03125s | 143.3s | 0.07813s | 170.2s | 0.03125 |
| *CT_M* | 263.7s | 0.8594s | 77.75s | 0.1875s | 176.5s | 0.1719s | 422.9s | 0.1563 |
| *CT_P* | 18.67s | 27.34s | 13.58s | 24.14s | 22.83s | 36.83s | 12.48s | 17.25s |
| *CT_EO* | - | 0.1250 | - | 0.0625s | - | 0.1406s | - | 0.1250s |
| *CT_W* | **748.4s** | **28.34s** | **142.1s** | **24.42s** | **342.7s** | **37.22s** | **605.6s** | **17.563s** |

a 3.66-GHz Pentium 4 computer with 2.5 GB RAM using an unoptimized Matlab 2006b implementation. In our experiments, we use one process and one thread to run each of the two algorithms. *cputime* is a MATLAB keyword that returns the CPU time in seconds for any process.

Table III reveals the following three points. First, the number of correspondences used in our method is far fewer than that of Wills' method (see the first row of the table). Second, our proposed method spends much less time in feature detection, matching and motion layer estimation than Wills' method (see the second and third rows of the table). Third, even though our method has extra procedures for error correction and occlusion determination (which are not included in Wills' method), it still only spends 3%~20% of the total computation time of Wills' method (see the last row of the table).

There are two reasons why our proposed method is faster than Wills' method. First, Wills' method establishes a large number of correspondences (usually 10%-20% of the total number of pixels) to extract as much motion information as possible. However, for the purpose of motion layer estimation, we have shown that a large set of features is not required. Our proposed method establishes fewer accurate correspondences (less than 0.1% of the number of pixels). It achieves lower computational cost without losing accuracy in motion layer estimation. Second, Wills' method uses RANSAC to estimate homographies. As random sampling is applied, many incorrect seed homographies are found, which increases computation time. Our proposed voting-based method uses the relationship among features to guide seed homography estimation, which avoids finding incorrect homographies. This reduces computation time.

### *C. Error Correction and Occlusion Determination*

In our proposed method, the first two steps (feature extraction and matching, and motion layer estimation) contribute to reducing the computational cost of estimating motion layers (demonstrated in the previous subsection). The following two steps (error correction and occlusion determination) focus on achieving a complete and accurate segmentation.

Occlusion detection and determination are crucial to achieve complete motion segmentation. In our proposed method, occlusion detection is performed by pixel assignment, and occlusion determination is achieved by an EMD-based segment comparison. Figures 10 and 11 show the results of occlusion detection and determination. The occlusion layer detected by pixel assignment is shown in (a) and the final three layers are shown in (b)-(d) respectively. We see that the occluded regions have been segmented and assigned to correct motion layers. As we need all occlusions to be detected, the parameter $d$ in (4) is set to a small value of 0.12. As a result, some visible regions may be incorrectly assigned as occlusions. These regions are denoted by red rectangles in Figures 10(a) and 11(a). From the final results shown in (b)-(d), we see that motion labels are correctly determined by the EMD-based method. We see that our proposed method achieves good performance in occlusion

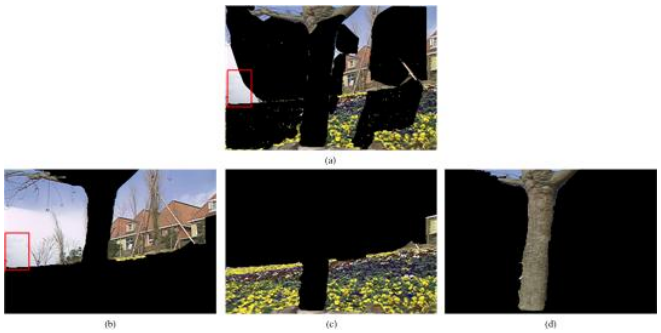


Fig. 10. The result of occlusion detection and determination for frame 15 of *Garden*. (a) The occlusion layer detected by pixel assignment. (b) The "sky and house" layer, (c) the "garden" layer and (d) the "tree" layer after the EMD-based method.

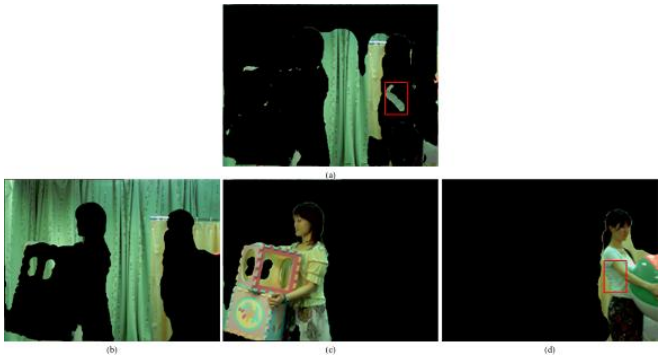


Fig. 11. The result of occlusion detection and determination for frame 58 of *Akko&Kayo*. (a) The occlusion layer detected by pixel assignment. (b) The "background" layer, (c) the "left girl" layer and (d) the "right girl" layer after the EMD-based method.

handling and is robust to incorrect occlusion detection.

Besides occlusion determination, the EMD-based segment comparison plays another important role in our framework. It corrects the labels of visible pixels that are incorrectly assigned by the MRF. This is illustrated in Figures 12 and 13. For one frame of the *Garden* sequence, the "garden" layer and the "sky and house" layer before the EMD-based procedure are shown in Figures 12(a) and 12(b) respectively. The region in the red rectangle is incorrectly assigned to the "garden" layer. Figures 12(c) and 12(d) show the two layers after the EMD-based method. The region in the red rectangle has been correctly reassigned to the "sky and house" layer. This is because the texture of the region is similar with other sky regions. Our EMD-based method assigns it to the correct motion layer. The result shown in Figure 13 is similar. The region in the red rectangle is reassigned to the "background" layer.

Our EMD-based method achieves good performance in error correction and occlusion determination. This is for two reasons. First, the MRF used in pixel assignment processes individual pixels. However, the EMD-based method is based on segments, so more information from segments is used to counter the drawbacks of the MRF. Second, as illustrated before, EMD can efficiently measure the similarity between segments. This benefits the final refinement and ensures the assignment is more accurate.

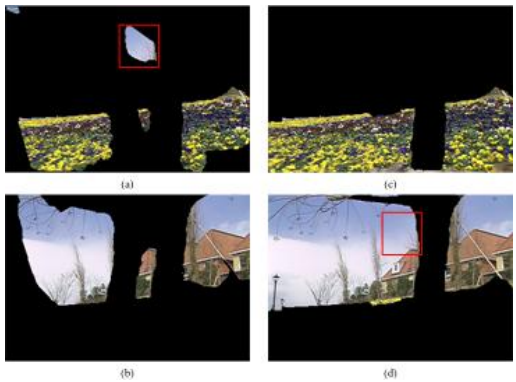


Fig. 12. The result of error correction for frame 1 of the *Garden* sequence. (a) The "garden" layer and (b) the "sky and house" layer before the EMD-based procedure. (c) The "garden" layer and (d) the "sky and house" layer after the EMD-based procedure.

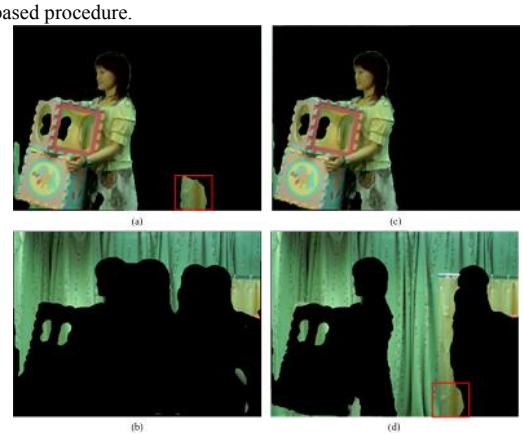


Fig. 13. The result of error correction for frame 58 of the *Akko&Kayo* sequence. (a) The "left girl" layer and (b) the "background" layer before the EMD-based procedure. (c) The "left girl" layer and (d) the "background" layer after the EMD-based procedure.

## VIII. Conclusion

In this paper, we propose an efficient method to achieve occlusion-aware motion layer extraction from two frames with large disparity motions. There are two major contributions to our method. First, our voting-based method performs accurate motion layer estimation from sparse correspondences, and requires less computation than current state-of-the-art methods. Second, our proposed MRF framework integrates color and motion consistency to perform complete occlusion detection, and our subsequent EMD-based method assigns occluded pixels to correct motion layers. In attempting to solve the occlusion problem, our method significantly improves segmentation accuracy.

In our proposed method, we assume a planar motion model. When an object with a complex 3D structure is near the camera, this assumption is no longer valid. Although the EMD-based method overcomes this limitation to some extent, the extension of our method to more general motion models is still required.

## Acknowledgment

The authors would like to thank Xilin Chen for helpful discussions and Josh Wills for sharing the code of their method [16].

## REFERENCES


[1] J. Wang and E. Adelson, "Representing Moving Images with Layers", IEEE Trans. Image Processing, vol. 3, pp. 625-638, Sep. 1994.

[2] Y. Weiss, "Smoothness in Layers: Motion Segmentation Using Nonparametric on Homographics", Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 1997.

[3] S. Ayer and H. Sawhney, "Layered Representation of Motion Video using Robust Maximum-Likelihood Estimation of Mixture Models and MDL Encoding", Proc. IEEE Int. Conf. Comput. Vis., 1995.

[4] S. Khan and M. Shah, "Object Based Segmentation of Video Using Color, Motion and Spatial", Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2001.

[5] I. Patras, E. Hendirks, and R. Lagendijk, "Video Segmentation by MAP Labeling of Watershed Segments", IEEE Trans. Pattern Anal. Mach. Intell., vol.23, pp. 326-332, Mar. 2001.

[6] P. Torr, R. Szeliski, and P. Anandan. "An integrated Bayesian approach to layer extraction from image sequences", IEEE Trans. Pattern Anal. Mach. Intell., vol.23, pp.297–303, Mar. 2001.

[7] H. Tao, H. Sawhney, and R. Kumar, "Object Tracking with Bayesian Estimation of Dynamic Layer Representations", IEEE Trans. Pattern Anal. Mach. Intell., vol. 24, pp. 75-89, Jan. 2002.

[8] G.F. Zhang, J.Y. Jia, W. Xiong, T.T. Wong, P.A. Heng, and H.J. Bao, "Moving Object Extraction with a Hand-held Camera", Proc. IEEE Int. Conf. Comput. Vis., 2007.

[9] S.S. Huang, L.C. Fu, and P.Y. Hsiao, "Region-Level Motion-Based Background Modeling and Subtraction Using MRFs", IEEE Trans. Image Process., vol.16, pp.1446-1456, May 2007.

[10] S.S. Huang, L.C. Fu, and P.Y. Hsiao, "Region-Level Motion-Based Foreground Segmentation Under a Bayesian Network", IEEE Trans. Circuits and Syst. Video Technol., vol. 19, pp. 522-532, Apr. 2009.

[11] A. Criminisi, G. Cross, A. Blake, and V. Kolmogorov, "Bilayer segmentation of live video", Proc. IEEE Conf. Comput. Vis. Pattern Recognit. , 2006.

[12] Q. Ke and T. Kanade, "A robust subspace approach to layer extraction", IEEE Workshop on Motion and Video Computing, 2002.

[13] M. P. Kumar, P.H.S. Torr, and A. Zisserman, "Learning Layered Motion Segmentations of Video", Int. J. Comput. Vis., vol. 76, pp. 301-319, Mar. 2008.

[14] J.J. Xiao and M. Shah, "Motion Layer Extraction in the Presence of Occlusion Using Graph Cuts", IEEE Trans. Pattern Anal. Mach. Intell., vol. 27, pp. 1644-59, Oct. 2005.

[15] P.H.S. Torr and A. Zisserman, "Feature Based Methods for Structure and Motion Estimation", Proc. Vision Algorithms Workshop, 1999.

[16] J. Wills, S. Agarwal, and S. Belongie, "A Feature-based Approach for Dense Segmentation and Estimation of Large Disparity Motion", Int. J. Comput. Vis., vol. 68, pp. 125-143, Jun. 2006.

[17] D. Lowe, "Object recognition from local scale-invariant features", Proc. IEEE Int. Conf. Comput. Vis., 1999.

[18] Y. Rubner, C. Tomasi, and L.J. Guibas, "The earth mover's distance as a metric for image retrieval", Int. J. Comput. Vis., vol. 40, pp. 99-121, Nov. 2000.

[19] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors", IEEE Trans. Pattern Anal. Mach. Intell., vol. 27, pp. 1615–1630, Oct. 2005.

[20] H. Bay, T. Tuytelaars, and L.V. Gool, "Surf: Speeded Up Robust Features", Proc. European Conf. Computer Vision, 2006.

[21] M. Nicolescu and G. Medioni, "Layered 4D Representation and Voting for Grouping from Motion", IEEE Trans. Pattern Anal. Mach. Intell., vol. 25, pp. 492-501, Apr. 2003.

[22] C. Min and G. Medioni, "Inferring Segmented Dense Motion Layers Using 5D Tensor Voting", IEEE Trans. Pattern Anal. Mach. Intell., vol. 30, pp. 1589-1602, Sep. 2008.

[23] M. A. Fischler and R. C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography", Communications of the ACM, vol. 24, pp. 381-395, Jun. 1981.

[24] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images", Proc. of IEEE Int. Conf. on Computer Vision 1998.

[25] J. Kleinberg and E. Tardos, "Approximation Algorithms for Classification Problems with Pairwise Relationships: Metric Labeling and Markov Random Fields", J. ACM, vol. 49, pp. 616-630, Sep. 2002.

[26] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts", IEEE Trans. Pattern Anal. Mach. Intell., vol. 23, pp. 1222– 1239, Nov. 2001.

[27] P.F. Felzenszwalb and D.P. Huttenlocher, "Efficient graph-based image segmentation", Int. J. Comput. Vis., vol. 59, pp. 167-181, Sep.2004.

[28] Y. Rubner, L. J. Guibas, and C. Tomasi, "The earth mover's distance, multi-dimensional scaling, and color-based image retrieval", Proc. DARPA Image Understanding Workshop, 1997.

[29] Y. Rubner and C. Tomasi, "Texture-based image retrieval without segmentation", Proc. IEEE Int. Conf. Comput. Vis., 1999.

[30] Y.X. Peng and C.W. Ngo, "EMD-Based Video Clip Retrieval by Many-to-Many matching", International Conference on Image and Video Retrieval, 2005.

[31] A. Sharma and K.K. Paliwal, "Fast principal component analysis using fixed-point algorithm", Pattern Recognition Letters, vol.28, pp.1151-1155, Jul. 2007.

[32] K.Y. Wong and M.E. Spetsakis, "Tracking based motion segmentation under relaxed statistical assumptions", Computer Vision and Image Understanding, vol.101, pp.45-64, Jan. 2006.

[33] R. Venkatesh Babu, K. R. Ramakrishnan, and S. H. Srinivasan, "Video object segmentation: a compressed domain approach", IEEE Trans. Circuits Syst. Video Technol., vol. 14, pp. 462-474, Apr. 2004.

[34] W. Brendel and S. Todorovic, "Video Object Segmentation by Tracking Regions", Proc. IEEE Int. Conf. Comput. Vis., 2009.


**Feng Xu** received a B.E. degree from Tsinghua University, Beijing, China in 2007. He is currently working towards a Ph.D. Degree at Department of Automation, Tsinghua University, Beijing, China.

His research interests include image/video processing, computer vision, and computer graphics.

**Qionghai Dai** (SM'05) received a B.S. degree in mathematics from Shanxi Normal University, China, in 1987, and M.E. and Ph.D. degrees in computer science and automation from Northeastern University, China, in 1994 and 1996 respectively.

Since 1997, he has been with the faculty of Tsinghua University, Beijing, China, and is currently Professor and the Director of the Broadband Networks and Digital Media Laboratory. His research areas include signal processing, broad-band networks, computer vision, and computer graphics.