Motion Layer Extraction in the Presence of Occlusion using Graph Cut

Jiangjian Xiao Mubarak Shah

Computer Vision Lab, School of Computer Science
University of Central Florida, Orlando, Florida 32816, USA
{jxiao, shah}@cs.ucf.edu

Abstract

Extracting layers from video is very important for video representation, analysis, compression, and recognition. Assuming that a scene can be approximately described by multiple planar regions, this paper describes a robust novel approach to automatically extract a set of affine transformations induced by these regions, and accurately segment the scene into several motion layers. First, a number of seed regions are determined by using two frame correspondences. Then the seed regions are expanded and refined using the level set representation and employing graph cut method. Next, these initial regions are merged into several initial layers according to the motion similarity. Third, after exploiting the occlusion order constraint on multiple frames the robust layer extraction is obtained by graph cut algorithm, and the occlusions between the overlapping layers are explicitly determined. Several examples are demonstrated in the experiments to show that our approach is effective and robust.

1 Introduction

Automatic extraction of layers from video sequence has broad applications, such as video compression and coding, recognition, etc. Once, a motion segmentation is achieved, a video sequence can be efficiently represented by different layers. The major steps of motion segmentation consists of: (1) determining the layer descriptions, which include the number of layers and the motion parameters for each layer; (2) assigning each pixel in the image sequence to the corresponding layer and identifying the occluded pixels.

In an earlier work, Wang and Adelson proposed the use of optical flow to estimate the motion layers, where each layer correspond to a smooth motion field [17]. Ayer and Sawhney combined MDL and MLE in an EM framework to estimate the number of layers and the motion model parameters for each layer [1]. Several other approaches used MAP or MLE for estimation of model parameters assuming different constraints and motion models [18, 22, 7, 16, 12].

Another class of motion segmentation approaches is to group the pixels in a region by using linear subspace constraints. In [6, 5], Ke and Kanade first expanded the seed regions into the initial layers by using k-connected components. After enforcing a low dimensional linear affine subspace constraint on multiple frames, they clustered these initial layers into several groups, and then assigned the image pixels to these layers. Zelnik-Manor and Irani also used the homography subspace for planar scenes to extract a specific layer and to register the images based on this layer [21].

Recently, graph cut [2, 8, 9, 3] was proposed to successfully minimize energy functions for various computer vision problems, such as stereo, image segmentation, image restoration, texture synthesis [10], etc. After formulating these different problems into graph cut framework, an optimal solution of energy minimization problem can be obtained in a polynomial time. In the motion segmentation area, Shi and Malik first used the normalized graph cut to extract layers from a video sequence [14]. However, since they grouped pixels based on the affinity of motion profile, a local measurement, their method ignored the global constraints and appeared unstable for the noisy image sequences. Wills et al proposed the use of graph cut to extract layers between two wide baseline images [19]. After employing the RANSAC technique, they clustered the correspondences into several initial layers, then performed the dense pixel assignment via graph cut for these two images.

After extensively reviewing the previous work on motion segmentation, we did not find any work in the literature, which deals with the explicit detection of the occluded pixels that do not belong to any layer. However, the occlusion problem has been widely studied for a long time in stereo algorithm [8, 9, 4]. The accurate detection of the occluded areas is very important to improve the dense disparity maps and the quality of 3D reconstruction. Similarly, in motion segmentation area, this occlusion problem is essential to detect the discontinuities between the overlapping layers and improve the quality of the layer boundaries.

In this paper, we propose a novel approach to extract accurate layer representations from a video sequence and ex-

plicitly determine occlusions between the overlapping layers. Our algorithm is implemented in two stages. In the first stage, we determine seed correspondences over a short video clip (3-5 frames). Then, we gradually expand each seed region from an initial rectangular patch of fixed dimensions into an enlarged support region of an arbitrary shape. This is achieved using a graph cut approach integrated with the level set representation. After that, we employ two-step merging process to obtain a layer description of the video clip. In the second stage, according the observation of that the occlusion area is increasing with the temporal order, we introduce the occlusion order constraint over multiple frames segmentation. After applying this constraint on the graph cut algorithm, we obtain a stable and accurate video segmentation in terms of layers and their 2D motion parameters. At the same time, the occluded pixels between overlapping layers are correctly identified, which greatly improve the quality of the layer boundaries.

The paper is organized as follows. Section 2 addresses how to extract layer descriptions from short video clips. Section 3 deals with the use of the occlusion order constraint and a multi-frame graph cut algorithm for obtaining precise layer segmentation in the presence of occlusion. In Section 4, we demonstrate several results obtained by our method.

2 Layer descriptions extraction

In our approach, the first stage is to extract the layer descriptions from video sequence, which include the number of layers and the motion parameters for each layer. In this stage, we first detect the robust seed correspondences over a short video clip. Next, using the previous shape prior of the seed region, the region's front is gradually propagated along the normal direction using bi-partitioning graph-cut algorithm integrated with the level set representation. Third, we use a two-step merging process to merge the seed regions into several groups, such that each group belongs to a single motion field.

2.1 Determining robust seed correspondences

In order to correctly extract the layer descriptions, we consider a short video clip $\mathcal{L}$ instead of only two consecutive frames. The reason is that if the motion between two consecutive frames is too small, the motion parameters of different layers are not distinct to extract. Therefore, we use an average pixel flow $\bar{\nu}$ of the seed correspondences as a measurement to decide the number of frames in the video clip $\mathcal{L}$. If $\bar{\nu}$ between I_n and I_1 is greater than some threshold (i.e. 3 pixels), the number of frames $\mathcal{L}$ is set to n.

In our approach, first we detect the Harris corners in the first frame, then we use KLT tracking algorithm [15] or our matching algorithm [20] to track the corners over this short period using a 17×17 window support. Since the Harris corners are located in the textured areas, we can obtain reliable affine transformations for the seed regions, and skip the non-textured areas, where the motion parameter estimation is unreliable.

2.2 Expanding seed regions

Once the seed correspondences are determined between frames I_1 and I_n, we consider a 17×17 window patch around each seed corner as an initial layer, which corresponds to a planar patch in the scene. This way, we get a number of initial layers, and each layer is supported by a small patch with the corresponding affine transformation. Nevertheless, the affine motion parameters estimated using the small patches may over-fit the pixels inside the region, and may not correctly represent the global motion of a larger region. Particularly, when the corner is located at the boundary of two true layers, the over-fitting may introduce a serious distortion on this patch after applying the affine transformation.

One straightforward solution is to simply extend the region by including neighboring pixels which are consistent with the affine transformation. Such pixels can be determined by applying threshold to the SSD (Sum of Squared Differences) computed between the original and warped windows. However, this scheme has two problems: First, the resulting expanded region may not be compact and smooth. Second, the new patch may include the pixels from multiple layers, and may not be consistent with a single planar patch in the scene. Fig. 1.b shows one sample result obtained by using this simple scheme. The seed region is originated from the seed on the rotating ball (Fig. 1.a). After expanding the boundary and partitioning by applying a simple threshold, the region is not that smooth, and it also includes the pixels from the other layers.

In order to deal with these problems, we propose a novel approach to gradually expand the seed region by identifying the correct supporting pixels by using the bi-partitioning graph cut method and employing the level set representation. We introduce a smoothness energy term, which can maintain the partitions piecewisely smooth and naturally solve the first problem. Then, using level set representation, the contour of the seed region is gradually evolved by propagating the region's front along its normal direction.

A weighted graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$ is defined by a set of nodes $\mathcal{V}$ (image pixels) and a set of directed edges $\mathcal{E}$ which connect these nodes as shown in Fig. 2. In this graph, there are two distinct nodes s and t, called the source and sink respectively. The edges connected to the source or sink are called

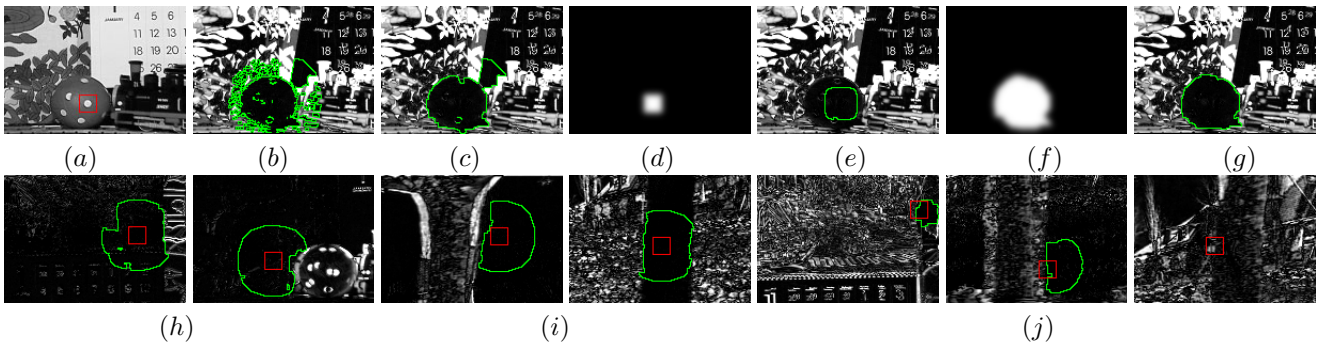


Figure 1: Region expansion process. Top: A procedure for expansion of an initial 17×17 seed region (a) to a large support region. (b) The simple expansion and partitioning. (c) bi-partitioning without the level set representation. (d) and (f) respectively are the expansions of the seed region during the first and fourth iterations using the level set representation. (e) and (g) are the results obtained after the graph cut partitioning. Bottom: Several results from the mobile-calendar (h) and flower-garden sequences (i), where the new region can have an arbitrary compact contour. The last three images (j) show three poor seed regions detected by employing the coverage threshold. *Note:* The red box is the initial seed region. The green contours are obtained after using bi-partitioning algorithm.

t-links, such as (s, p) and (p, t). The edges connected to two neighboring pixel nodes are called n-links, which have two directions, such as (p, q) and (q, p).

The problem of expanding the seed region can be easily formulated into the graph cut framework as a bi-partitioning problem of a vertex set. In this framework [2], we seek the labelling function f by minimizing the energy

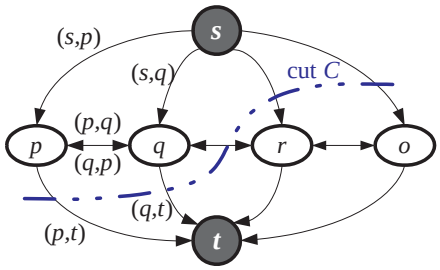
$$\begin{aligned} E &= E_{smooth}(f) + E_{data}(f) \\ &= \sum_{(p,q)\in\mathcal{N}} V(p,q) + \sum_{p\in\mathcal{P}} D_p(f_p), \end{aligned} \quad (1)$$

where E_{smooth} is a piecewise smoothness term, the E_{data} is a data error term, $\mathcal{P}$ is the set of pixels in the image, $\mathcal{N}$ is a 4-neighbor system, f_p is the label of a pixel p, $D_p(f_p)$ is data penalty function, and $V(p, q)$ is smooth penalty function[1]. In this bi-partitioning problem, the label f_p of the pixel p is assigned either 0 or 1. If $f_p = 0$, the pixel p is supporting this seed region, otherwise, this pixel is not supporting the region.

In graph $\mathcal{G}$, after setting the weights of t-links to $D_p(0)$ on the source side and $D_p(1)$ on the sink side, and the weights of n-links to $V(p, q)$, we can compute the minimum cut $\mathcal{C}$ using the standard graph cut algorithm and obtain piecewise smooth partition of the supporting region.

However, the partitioning using graph cut cannot guarantee the gradual expansion or shrinking of a region along the normal direction as shown in Fig. 1.c, where some pixels not belonging to this region are also included. Since the contour information of the initial seed region is not integrated in the function given in equation 1, the graph cut algorithm cannot correctly evolve the region contour along the normal direction. In order to solve this problem, we

[1]This is a simpler version of Eq. 4.



edge	weight
(s,p)	$D_p(0)$
(p,t)	$D_p(1)$
(p,q) (q,p)	V(p,q)

Figure 2: An example of a graph $\mathcal{G}$ for 1D image. Nodes p, q, r, and o are the pixels in the image. After computing minimum cut $\mathcal{C}$, the nodes are partitioned into supporting pixels (source) and un-supporting pixels (sink). The weights of the links are listed in the table on the right.

use the contour of the seed region as a prior to compute the level set representation of this region. Then, after adjusting the weights of the t-links for pixels on the region boundary in graph $\mathcal{G}$, we use the graph cut algorithm to gradually expand the seed region. The detailed process is described as follows:

- *Step 1*: Construct a mask β of the original seed region, which has a value in [0, 1], where the inside pixels of the region are marked by 1 and the others are marked by 0. Then, compute a level set v by simply convolving the region mask β with a Gaussian kernel as: $v = G * \beta$, where G is the Gaussian kernel.
  For each pixel i, the v_i inside of the seed region has a high constant value, and the v_i outside of the region falls down along the contour normal direction until $v_i = 0$. Therefore, we obtain an implicit surface for this contour evolution, which can be represented by

level set [11, 13]. Here we propose another approach to evolve the region contour by integrating the level set representation into graph cut method as the next two steps.

- *Step 2*: Warp the second image using the corresponding affine transformation, and compute SSD between the warped image and the first frame. Construct a graph $\mathcal{G}$ for the pixel with $v_i > 0$. Compute data penalty D_p according to the computed SSD and smoothness penalty $V(p,q)$ for each link in this graph.
- *Step 3*: Use the level set v as a weight to change t-links of each pixel at the sink side, then compute the minimum cut $\mathcal{C}$.
  The weights of the pixels inside the region are almost not changed, while the weight (p,t) will decrease when the pixel p is away from the boundary. As a result, the minimum cut $\mathcal{C}$ is most likely to cut the outside pixels, and label them as the un-supporting pixels for this region. This way, the seed region will gradually propagate from the center to outside.
- *Step 4*: Use the new computed region as the seed region to compute a new affine transformation by minimizing the image residue inside the region, then goto *Step 1* to do the next iteration. If the new region is shrinking and cannot cover 75% area of the original seed region (coverage threshold), it is discarded as a poor initial layer.

After a few iterations of the above steps, the front of the seed region will either expand or shrink along the normal direction of the contour.

Fig. 1 shows a detailed process for seed region expansion. Fig. 1.d shows the level set representation obtained from the initial seed region (Fig. 1).a. Fig. 1.e and 1.g are the partitioning results after the first and fourth iterations. In the second row of Fig. 1, we show several results for seed region expansion of the mobile-calendar and flower-garden sequences. The last three images (Fig. 1.j) show that we can identify the poor seed regions using the coverage threshold. Most of these poor seed regions are located at the boundary of multiple layers.

2.3 Two-step region merging process

After expanding the regions, each good seed region becomes an initial layer. Most of these layers may share the same affine transformation. Therefore, we design a two-step merging algorithm to merge these layers to obtain the layer descriptions.

In the first step, we only merge the layers which overlap with each other. Given two regions R_1 and R_2, we test if the number of overlapping pixels are more than half of pixels in the smaller region. If this is true, we compute the SSD by warping the first region, R_1, using the transformation H_2 of the second region R_2. Using this SSD as the measure, we can detect how many pixels support H_2 using the graph cut algorithm. If the majority (say 80%) of pixels of R_1 support R_2, we merge these two regions and recompute the motion parameters using the merged pixels. Then, using bi-partitioning graph cut algorithm, we prune the un-supporting pixels from the new region.

If only a few pixels of R_1 support H_2, we repeat the process by warping R_2 using the transformation H_1 of R_1. In order to achieve large merged regions, we iterate the whole process a few times (typically 3 to 4) to make sure the merging process converges. As a result, only a few large regions remain, and some of non-overlapping regions may still share a single motion transformation. In the second merging step, we merge these non-overlapping regions using a similar process. Fig. 3 shows the results for the mobile-calendar sequence.

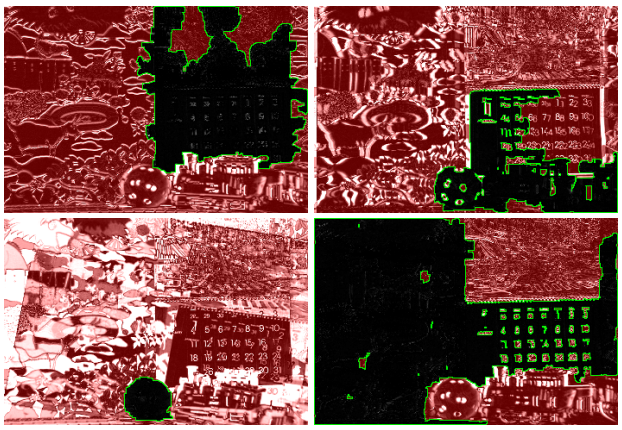


Figure 3: Four layers of the mobile-calendar sequence, which are corresponding to the calendar, train, ball, and wall respectively. The green contour is the region boundary, and non-supporting pixels are marked by red. *Note:* The non-textured areas may belong to several layers due to their ambiguities, such as the white paper at the lower part of the calendar in the mobile-calendar sequence.

3 Multi-frame layer segmentation in presence of occlusion via graph cut

In this section, we will address the following problem: Given the extracted layer descriptions, how to compute an accurate layer segmentation in presence of occlusion using *multiple* frames of this short video clip. In this paper, we propose an approach to explicitly identify the occluded pixels, where every occluded pixel is assigned a new occlusion label, f_{oc}. First, we will state the occlusion order constraint.

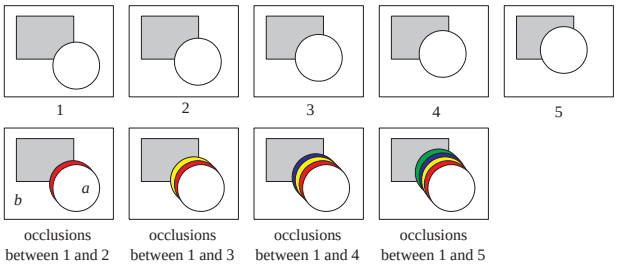


Figure 4: The occlusion order in a short video clip containing five consecutive frames (the first image is the reference image). The top row is the five-frame sequence, where a solid circle is moving along the left-top direction. The bottom images show that the occlusions (color areas) between the first frame and the other frames are increasing with time.

Then, we embed this constraint into the labelling function, which can be minimized using the multi-frame graph cut algorithm.

3.1 Occlusion order constraint

With the intention of computing an accurate motion layer segmentation of a video clip, first we analyze the occlusion process over a temporal domain. Fig. 4 shows the occlusion has a temporal order for a linearly moving object. It is obvious that occlusion area is increasing with the temporal order. During a short period (3-5 frames), this observation is not violated if the object is not too thin or not moving too fast. Therefore, based on this assumption, we state the occlusion order constraint as follows:

- *Rule 1*: During a short period, if one pixel is occluded between frames 1 and j, this pixel will also be occluded between frames 1 and $(j+1)$.
- *Rule 2*: If the pixel p is assigned a label f_p between frames 1 and j, then pixel p should be assigned either f_p or f_{oc} between frames 1 and k, where $k > j$; and pixel p should be assigned f_p between frames 1 and k, where $k < j$.

According to this occlusion order constraint, only pixels at the same image coordinates in two consecutive frame pairs can influence each other, such as the frame pairs (1,2) and (1,3).

Now, this multi-frame motion segmentation problem can be formulated as an energy minimization problem of the following function:

$$E = \sum_{j=1}^{n-1}(E_{smooth}(f) + E_{data}(f) + E_{occ}(f)) + \sum_{j=1}^{n-2} E_{order}(f),$$

where j is frame number, and n is the total number of frames. Compared to Eq. 1, there are two additional terms

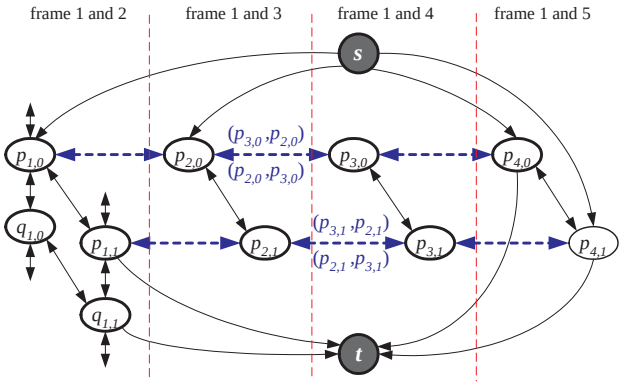


Figure 5: This graph is constructed by five consecutive frames, which have four image pairs related to the reference image. The red lines separate each pair of images into one block. The blue n-links are introduced to maintain the occlusion order constraint. *Note: Only part of the nodes and links are drawn.*

in this equation. The first one is $E_{occ}(f)$, which is used to impose the occlusion penalties for the occluded pixels between frames 1 and $(j+1)$. The second one is $E_{order}(f)$, which is used to impose occlusion order penalties for maintaining the occlusion order constraint on each consecutive pair of image pairs.

3.2 Multi-frame motion segmentation

In order to minimize this energy function, we employ the graph cut method and construct a graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$ involving multiple consecutive frames as shown in Fig. 5. To illustrate this, we stack four pairs of image nodes together in this graph, note that each image pair involves the first frame (the reference frame) and one of the other frames, which is consistent with Fig. 4.

In Fig. 5, each image pair is separated by the red dotted lines. In each image pair $(1, j+1)$, $j > 1$, and every pixel p, there is a pair of nodes $p_{j,0}$ and $p_{j,1}$. All of these nodes from the image pixels form a new subset $\mathcal{A} = \mathcal{V} - \{s, t\}$. According to the occlusion order constraint, a set of *order n-links* (blue edges), such as $(p_{3,0}, p_{2,0})$ and $(p_{2,0}, p_{3,0})$, are added in the graph $\mathcal{G}$ to interact with the nodes at the same image coordinates. To simplify the graph $\mathcal{G}$, we only show two nodes from one particular pixel p for each image pair to illustrate these *order n-links*. The detailed sub-graph $\mathcal{G}_{1,2}$ for the first image pair is redrawn in Fig. 6.

Before we describe how to minimize the energy E for the whole graph $\mathcal{G}$, we first discuss the interaction of the nodes in sub-graph $\mathcal{G}_{1,2}$, and then discuss how to assign the weights to these links. To reduce the complexity, we only show three pixel graphs p, q, and r in graph $\mathcal{G}_{1,2}$, where the

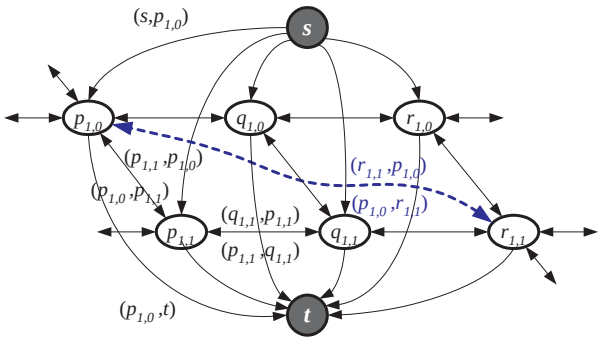


Figure 6: A graph $\mathcal{G}_{1,2}$, where three basic pixel graphs are shown corresponding to pixels p, q, are r respectively. The n-links between neighboring pixels is to enforce the smoothness penalties, such as $(p_{1,1}, q_{1,1})$ and $(q_{1,1}, p_{1,1})$. The new blue n-links are introduced to enforce the symmetric property of the occlusion, such as $(p_{1,0}, r_{1,1})$ and $(r_{1,1}, p_{1,0})$.

pixel graph corresponds to one pixel in the reference image, and is the basic element to construct this whole graph.

In each pixel graph, there are two nodes, such as $p_{1,0}$ and $p_{1,1}$, and one pair of *occlusion* n*-links*, such as $(p_{1,0}, p_{1,1})$ and $(p_{1,1}, p_{1,0})$. If the minimum cut $\mathcal{C}$ cuts the link $(p_{1,0}, p_{1,1})$, the pixel p is occluded. After assigning weights D_{oc}, ∞, $D_p(f_p)$, and $D_p(\alpha)$ on links $(p_{1,0}, p_{1,1})$, $(p_{1,1}, p_{1,0})$, $(p_{1,1}, t)$, and $(s, p_{1,0})$ respectively, Fig. 7 shows three cases of $\mathcal{C}$s after applying the standard α-expansion technique [2, 8] on one independent pixel graph. Let f_p be the original label of a pixel p in the reference image. The pixel, p, will be assigned a new label $f_p^{\mathcal{C}}$ as follows:

$$f_p^{\mathcal{C}} = \begin{cases} \alpha & \text{if } (s, p_{1,0}) \in \mathcal{C}, (s, p_{1,1}) \in \mathcal{C} \text{ (Fig.7.}a\text{)}, \\ f_p & \text{if } (p_{1,0}, t) \in \mathcal{C}, (p_{1,1}, t) \in \mathcal{C} \text{ (Fig.7.}b\text{)}, \\ f_{oc} & \text{if } (p_{1,0}, t) \in \mathcal{C}, (s, p_{1,1}) \in \mathcal{C}, \\ & \quad (p_{1,0}, p_{1,1}) \in \mathcal{C} \text{ (Fig.7.}c\text{)}, \end{cases} \tag{2}$$

where f_{oc} is the occlusion label. In the occlusion case, either $D_p(\alpha)$ or $D_p(f_p)$ of pixel p is greater than the occlusion penalty D_{oc}. It means that it is not suitable to assign either the original label f_p or the new label α to this pixel. Therefore, this pixel is an occluded pixel and is assigned f_{oc}.

In graph $\mathcal{G}_{1,2}$, the smoothness energy term $E_{smooth}(f)$ is implemented by the *smoothness* n*-links*, which connect each pair of neighboring pixel graphs such as $(q_{1,1}, p_{1,1})$ and $(p_{1,1}, q_{1,1})$. In order to compute the smoothness penalty term $V(p_{1,i}, q_{1,i})$ of a link $(p_{1,i}, q_{1,i})$, we warp the second image I_2 to obtain the warped image $I_2^{H_{f_i}}$ by applying the motion transformation H_{f_i}, corresponding to label f_i, for each label in the layer descriptions. Here

$$f_i = \begin{cases} \alpha & \text{if } i = 0 \\ f_p & \text{if } i = 1 \end{cases}. \tag{3}$$

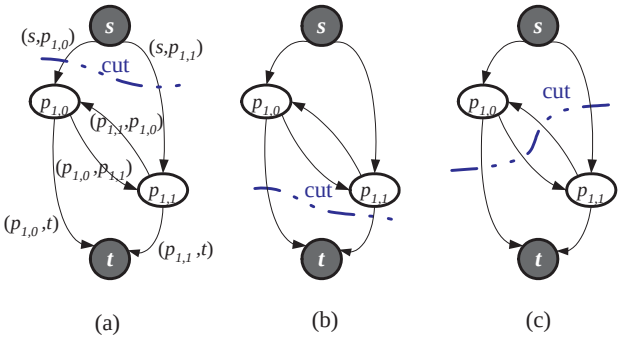


Figure 7: Three possible cases after one α-expansion of an independent pixel graph p. (a) p is assigned to the new label α. (b) p will keep the original label f_p. (c) p is occluded and assigned to the label f_{oc}.

Therefore, the smoothness penalty term $V(p_{1,i}, q_{1,i})$ can be computed as

$$V(p_{j,i}, q_{j,i}) = \begin{cases} 4\lambda & \text{if } \max(|I_1(p) - I_1(q)|, \\ & \quad |I_{(j+1)}^{H_{f_i}}(p) - I_{(j+1)}^{H_{f_i}}(q)|) < 4, \\ 2\lambda & \text{if } 4 \leq \max(|I_1(p) - I_1(q)|, \\ & \quad |I_{(j+1)}^{H_{f_i}}(p) - I_{(j+1)}^{H_{f_i}}(q)|) < 8, \\ \lambda & \text{otherwise}, \end{cases} \tag{4}$$

where λ is an empirical constant, I_1 is the first frame, $I_{(j+1)}^{H_{f_i}}$ is the warped version of $I_{(j+1)}$ obtained by applying transformation H_{f_i}.

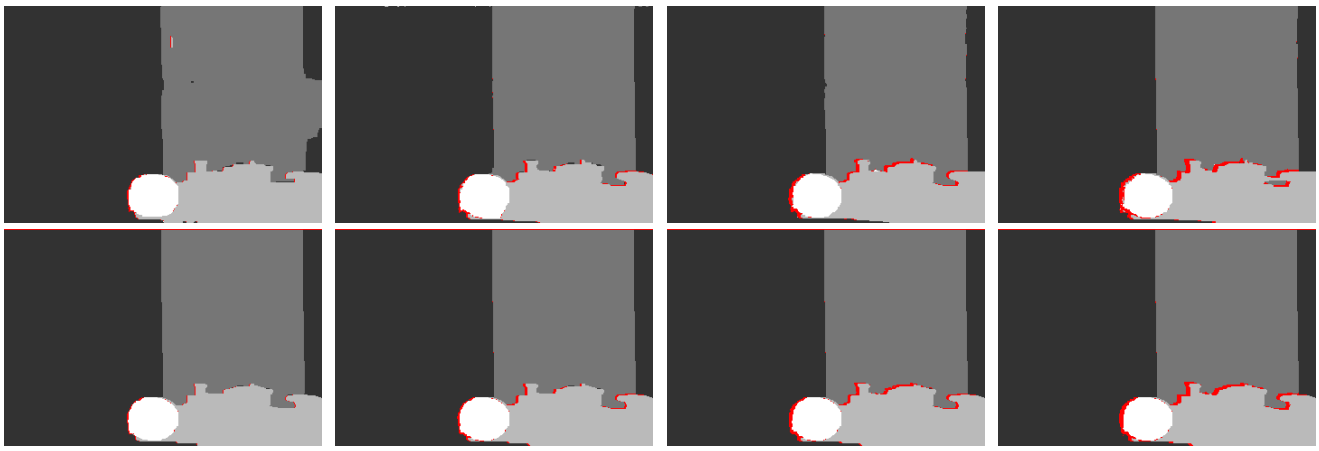
To deal with the symmetric properties of the occlusion, a set of new symmetric occlusion n-links are added to connect the related nodes. In Fig. 6, a pair of *symmetric* n-links are added to connect these two nodes, such as the blue dotted links $(r_{1,1}, p_{1,0})$ and $(p_{1,0}, r_{1,1})$. With the help of these *symmetric* n-links, the occlusion penalties from frame 2 to frame 1 are also specified.

After assigning weights 0, ∞, ∞, and 0 to *order* n-links $(p_{(i+1),0}, p_{i,0})$, $(p_{i,0}, p_{(i+1),0})$, $(p_{(i+1),1}, p_{i,1})$, and $(p_{i,1}, p_{(i+1),1})$ respectively, the occlusion order constraint is fully satisfied, which can be easily verified by assuming the minimum cut position.

Fig. 8 compares the segmentation results obtained using five frames with those obtained using only two frames. Due to the use of multiple frames, the artifacts are removed and the segmentation results are consistent as shown in Fig. 8*b-d*. Moreover, it is obvious that the occluded areas between the overlapping layers increases with the time.

4 Experiments

We have tested our approach on two standard motion sequences, mobile-calendar and flower-garden (Fig. 9), and one additional car-map (Fig. 10) sequence.



(a) $1-2$ (b) $1-3$ (c) $1-4$ (d) $1-5$

Figure 8: Segmentation results of frame 1 in the mobile-calendar sequence. Top: The segmentation results obtained by using only two frames. Bottom: The multi-frame segmentation results obtained by using five frames (1-5). $(a-d)$ Segmentation results between frames 1 and 2-5 respectively. The red pixels in segmented images are the occluded pixels.

Fig. 9 shows the segmentation results for the mobile-calendar and flower-garden sequences. We used five frames to extract the layers for the mobile-calendar sequence, and used three frames to extract the layers for the flower-garden sequence. We also compared our results with the the other methods [6, 5, 17, 1] for these two standard sequences. Since the ground truth of these data are not available, we have to limit our analysis to the qualitative comparisons. Our approach can explicitly determine the occluded pixels, and in our results, the boundaries between overlapping layers are more precise and finer than the previous methods.

We also applied our method to our own sequence with a large occlusion, car-map (Fig. 10), where the car is moving behind the map and the scale of the car is apparently changed. The sequence is taken by a hand-held moving video camera. During some frames, most parts of the car are occluded by the map. Once the car moves behind the map, it is very difficult to compute the correct motion parameters for car layer based on a small region of the car due to the over-fitting problem. Therefore, we use a common tracking technique to predict the motion parameters based on the previous frames. If the region shrinks by some amount (say 20%) and the predicted motion parameters are much different than the new estimated parameters, we keep the predicted parameters to perform the segmentation. The results are shown in Fig. 10.

In all of our experiments, once the layer descriptions are extracted, the average computational time for one frame segmentation is less than 30 seconds. *Note: All of our results are also available at our web site [23].*

5 Conclusions

In this paper, we presented an effective method to extract robust layer descriptions and to perform an accurate layer segmentation for image sequences containing 2-D motion. Our contributions consist of: (1) Initial layer descriptions by integrating the level set representation into the graph cut method to obtain gradually expanding seed regions. (2) Using the occlusion order constraints, we successfully combine multiple frames to compute accurate layer segmentation and explicitly detect the occluded pixels, which have not been done before.

In the future, we will investigate the relationship between the level set and graph cut methods, and unify these two approaches into one framework for different applications.

6 Acknowledgement

We would like to thank Khurram Shafique, Yunjun Zhang, Alper Yilmaz, and Yaser Ajmal for helpful discussions.

References

[1] S. Ayer, H. Sawhney, “Layered representation of motion video using robust maximum-likelihood estimation of mixture models and mdl encoding”, ICCV, 1995.

[2] Y. Boykov, O. Veksler, R. Zabih, “Fast Approximate Energy Minimization via Graph Cuts”, PAMI, 23(11), 2001.

[3] Y. Boykov, V. Kolmogorov, “Computing Geodesics and Minimal Surfaces via Graph Cuts”, ICCV, 2003.

[4] S. Kang, R. Szeliski, J. Chai, ”Handing Occlusions in Dense Multi-view Stereo”, CVPR, 2001.

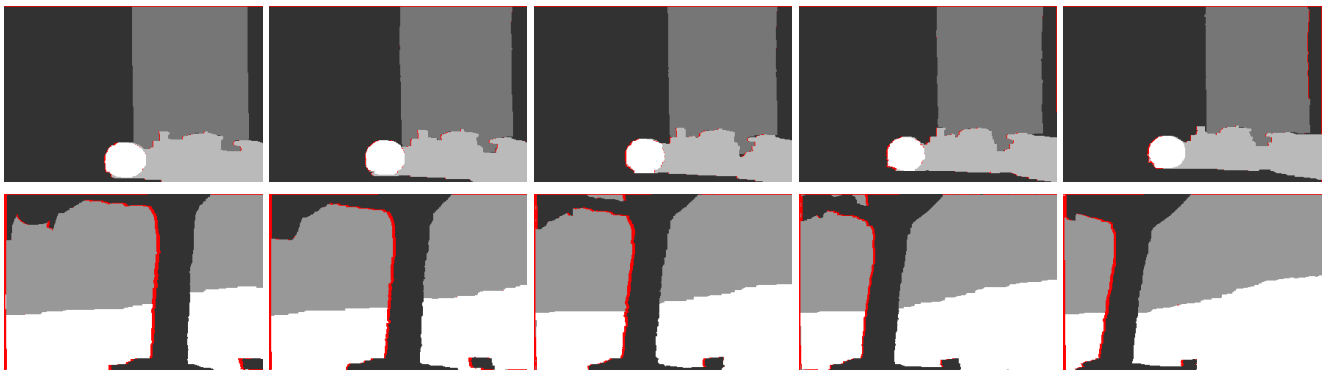


Figure 9: Top: The segmentation results for the mobile-calendar sequence. Bottom: The segmentation results for the flower-garden sequence. The red pixels in segmented images are the occluded pixels.

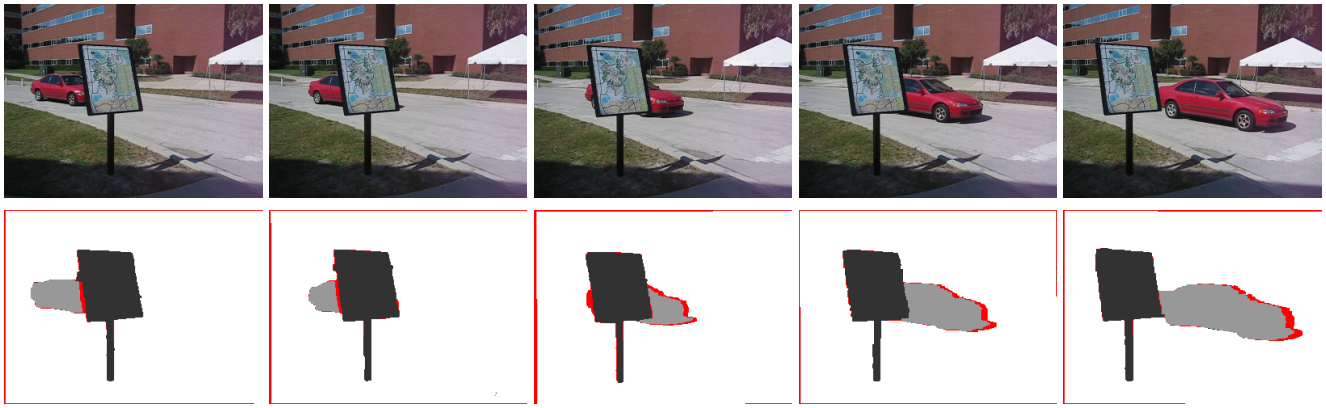


Figure 10: Segmentation results for the car-map sequence. Top: Several frames from the sequence. Bottom: The segmentation results, where the layers are accurately extracted even though the most parts of the moving car are occluded in some frames.

[5] Q. Ke, T. Kanade, "A Subspace Approach to Layer Extraction", CVPR, 2001.

[6] Q. Ke, T. Kanade, "A Robust Subspace Approach to Layer Extraction", IEEE Workshop on Motion and Video Computing, 2002.

[7] S. Khan, M. Shah, "Object Based Segmentation of Video Using Color, Motion and Spatial", ICCV 2001.

[8] V. Kolmogorov and R. Zabih, "Visual Correspondence with Occlusions using Graph Cuts", ICCV, 2001.

[9] V. Kolmogorov and R. Zabih, "Multi-camera Scene Reconstruction via Graph Cut", ECCV, 2002.

[10] V. Kwatra, I. Essa, A. Schdl, G. Turk, A. Bobick, "Graphcut Textures: Image and Video Synthesis Using Graph Cuts", SIGGRAPH, 2003.

[11] S. Osher, R. Fedkiw, "Level Set Methods and Dynamic Implicit Surfaces". The Springer-Verlag Press, 2003.

[12] I. Patras, E. hendirks, R. Lagendijk, "Video Segmentation by MAP Labeling of Watershed Segments". PAMI, 23 (3), 2001.

[13] J. Sethian. "Level Set Methods and Fast Marching Methods". Cambridge University Press, 1999.

[14] J. Shi, J. Malik, "Motion Segmentation and Tracking Using Normalized Cuts", ICCV, 1998.

[15] J. Shi, C. Tomasi, "Good Features to Track", CVPR, 1994.

[16] H. Tao, H. Sawhney, R. Kumar, "Object tracking with bayesian estimation of dynamic layer representations", PAMI, 24(1), 2002.

[17] J. Wang, E. Adelson, "Representing moving images with layers", IEEE Trans. Image Processing, 3 (5), 1994.

[18] Y. Weiss, "Smoothness in Layers: Motion Segmentation using Nonparametric on Homographics", CVPR, 1997.

[19] J. Wills, S. Agarwal, S. Belongie, "What Went Where", CVPR, 2003.

[20] J. Xiao, and M. Shah, "Two-Frame Wide Baseline Matching", ICCV 2003.

[21] L. Zelnik-Manor, M. Irani, "Multi View Subspace Constraints on Homographies", ICCV, 1999.

[22] Y. Zhou, H, Tao. "A Background Layer Model for Object Tracking through Occlusion", ICCV, 2003.

[23] http://www.cs.ucf.edu/∼vision/projects/motion_layer_extraction/.