Video Object Segmentation by Hypergraph Cut

Yuchi Huang Qingshan Liu Dimitris Metaxas
Department of Computer Science, Rutgers University
617 Bowser Road, Piscataway, NJ 08854

Abstract

In this paper, we present a new framework of video object segmentation, in which we formulate the task of extracting prominent objects from a scene as the problem of hypergraph cut. We initially over-segment each frame in the sequence, and take the over-segmented image patches as the vertices in the graph. Different from the traditional pairwise graph structure, we build a novel graph structure, hypergraph*, to represent the complex spatio-temporal neighborhood relationship among the patches. We assign each patch with several attributes that are computed from the optical flow and the appearance-based motion profile, and the vertices with the same attribute value is connected by a hyperedge. Through all the hyperedges, not only the complex non-pairwise relationships between the patches are described, but also their merits are integrated together organically. The task of video object segmentation is equivalent to the hypergraph partition, which can be solved by the hypergraph cut algorithm. The effectiveness of the proposed method is demonstrated by extensive experiments on nature scenes.*

1. Introduction

Video object segmentation is a hot topic in the communities of computer vision and pattern recognition, due to its potential applications in background substitution, video tracking, general object recognition, and content-based video retrieval. Compared to the object segmentation in static images, temporal correlation between consecutive frames, i.e., motion cues, will alleviate the difficulties in video object segmentation. Prior works can be divided into two categories. The first category aims at detecting objects in videos mainly from input motion itself. Representative work is layered motion segmentation [9] [13] [18]. They assume fixed number of layers and near-planar parametric motion models for each layer, and then employ some reasoning scheme to obtain the motion parameters for each layer. The segmentation results are obtained by assigning each pixel to one layer. When a non-textured region is presented in the scene, layered segmentation methods may not provide satisfactory results due to only using motion cues. The methods in [6] [10] [15] also belong to this category. They predefine explicit geometric models of the motion, and use them to infer the occluded boundaries of objects. When the motion of the data deviates from the predefined models, the performances of these methods will be degenerated.

The second category of approaches attempts to segment video objects with spatio-temporal information. In [4], the mean shift strategy is employed to hierarchically cluster pixels of 3D space-time video stack, which are mapped to 7-dimensional feature points, i.e., three color components and 4 motion components. [17] first uses the appearance cue as a guide to detect and match interest points in two images, and then based on these points, the motion parameters of layers are estimated by the RANSAC algorithm [5]. The method in [2] begins with a layered parametric flow model, and the objects are extracted and tracked by both the region information (provided by appearance and motion coherence) and the boundary information (provided by the result of the active contour). Recently, a complicated method is introduced to detect and group object boundaries by integrating appearance and motion cues [14]. This approach starts from over-segmented images, and then motion cues estimated from segments and fragments are fed to learned local classifiers. Finally the boundaries of objects are obtained by a global inference model. Different from the above methods, Shi and Malik [11] have proposed a pairwise graph based model to describe the spatio-temporal relations in the 3D video data and have employed the spectral clustering analysis to solve the video segmentation problem, which is beautiful and has achieved promising results.

However, in many real world problems, it is not complete to represent the relations among a set of objects as simple graphs based on a single similarity function. For example, based on affinity functions computed from different features, we may build different pairwise graphs. To combine these representations, one may consider a weighted similarity measure using all the features, but simply taking their weighted sum as the new affinity function may lead to the loss of some information which is crucial to the cluster-

978-1-4244-3991-1/09/$25.00 ©2009 IEEE

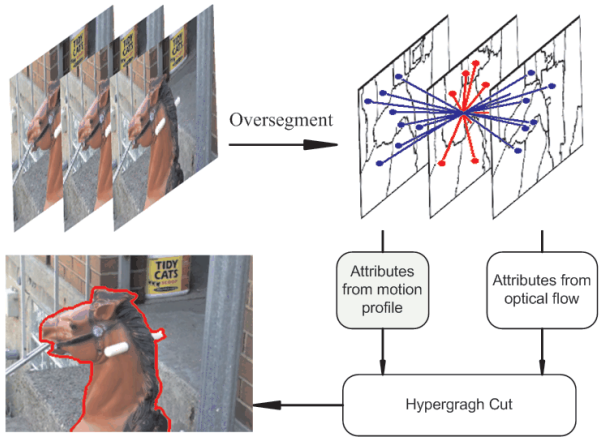


Figure 1. Illustration of our framework.

ing task. On the other hand, sometimes one may consider the relationship among three or more data points to determine if they belong to the same cluster. For example, we may compute the probability that one object and its 'neighbors' are in the same category. This representation for data sets with higher order relationships is a hypergraph, which is defined on a set of vertices and a set of weighted hyperedges. Such hypergraph structure is introduced in [1], but it still needs to be transferred to a weighted simple graph and use the normalized cut algorithm to solve the clustering problems.

In this paper, we propose a novel framework of video object segmentation based on *hypergraph*. Inspired by [14], we first over-segment the images by the appearance information, and we take the over-segmented image patches as the vertices of the graph for further clustering. The relationship between the image patches becomes complex due to the coupling of spatio-temporal information, while forcibly squeezing the complex relationship into pairwise will lead to the loss of information. To deal with this issue, we present to use the hypergraph [19] to model the relationship between the over-segmented image patches. We describe the over-segmented patches in spatio-temporal domain with the optical flow and the appearance based motion profile. The hypergraph is presented to integrated them together closely. Different from simple graph, the relationship between vertices of the hypergraph is non-pairwise hyperedge connection (we denote the traditional graph with pairwise relationship as simple graph for simplicity). Graph vertices which have the same attribute value can be connected by a hyperedge. Through all the hyperedges, the complex non-pairwise relationship between image patches is described. We take the task of attribute assignment as a problem of binary classification. We perform the spectral analysis on two different motion cues respectively, and produce several attributes for each patch by some representative spectral eigenvectors. Finally, we use the hypergraph cut algorithm to obtain global optimal segmentation of video objects under a variety of conditions, as evidenced by extensive experiments.

The rest paper is organized as follows: The proposed framework is introduced in Section 2; We address the hyperedge computation in Section 3, and we present the hypergraph cut algorithm in Section 4; Experiments are reported in Section 5, and followed by the conclusions finally.

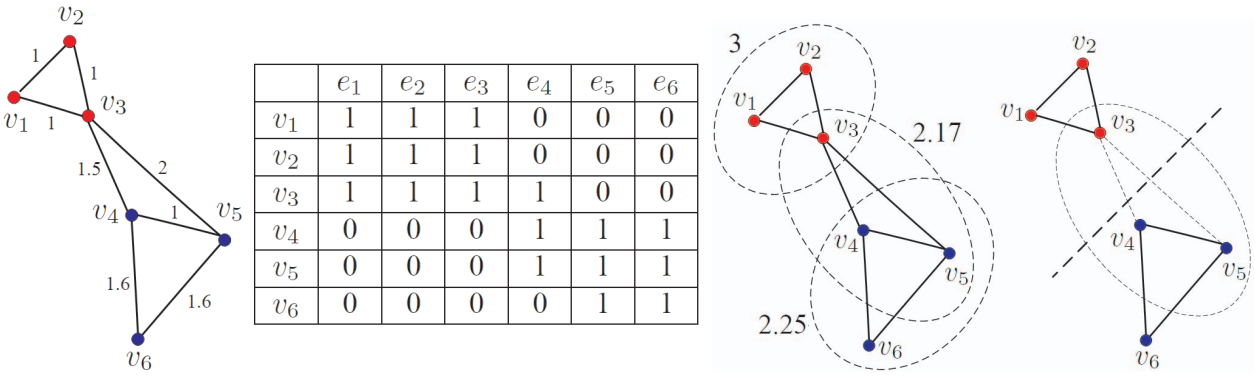
2. Overview of the proposed Framework

Video object segmentation can be regarded as clustering the image pixels or patches in the spatio-temporal domain. Graph model is demonstrated to be a good tool for data clustering, including image and video segmentation [11] [12]. In a simple graph, the data points are generally taken as the vertices, and the similarity between two data point is connected as an edge. However, for video object segmentation, the relationship among the pixels or patches may be far more complicated than the pairwise relationship due to the coupling of spatio-temporal information. Within a simple graph, these non-pairwise relationships should be squeezed to pairwise ones enforcedly, so that some useful information may be lost. In this section, we present to use the hypergraph to describe the complex spatio-temporal structure of video sequences. Before we overview the hypergraph based framework, we first introduce the concept of the hypergraph.

2.1. Concept of HyperGraph

The key difference between the hypergraph and the simple graph in that the former uses a subset of the vertices as an edge, i.e., a *hyperedge* connecting more than two vertices [19]. Let V represent a finite set of vertices and E a family of subsets of V such that $\bigcup_{e\in E} = V$, $G = (V, E, w)$ is called a hypergraph with the vertex set V and the hyperedge set E, and each hyperedge e is assigned a positive weight $w(e)$. For a vertex $v \in V$, its degree is defined to be $d(v) = \sum_{\{e\in E|v\in e\}} w(e)$. For a hyperedge $e \in E$, its degree is defined by $\delta(e) = |e|$. Let us use D_v,D_e, and W to denote the diagonal matrices of the vetex degrees, the hyperedge degrees, and the hyperedge weights respectively. The hypergraph G can be represented by a $|V| \times |E|$ matrix H which $h(v, e) = 1$ if $v_H \in e_H$ and 0 otherwise. According to the definition of H, $d(v) = \sum_{e\in E} w(e)h(v, e)$ and $\delta(e) = \sum_{v\in V} h(v, e)$. See details in [19].

Figure 2 shows an example to explain how to construct such a hypergraph. $v_1, v_2, ..., v_6$ are six points in a 2-D space and their interrelationships could be represented as a simple graph, in which pairwise distances between every vertex and its neighbors are marked on the corresponding edges. Consider that each vertex and its two-nearest neighbors form a hyperedge, a vertex-hyperedge matrix H could be given as Fig 2(b). For example, the hyperedge e_4 is composed of vertex v_4 and its two nearest neighbors v_3 and



	e_1	e_2	e_3	e_4	e_5	e_6
v_1	1	1	1	0	0	0
v_2	1	1	1	0	0	0
v_3	1	1	1	1	0	0
v_4	0	0	0	1	1	1
v_5	0	0	0	1	1	1
v_6	0	0	0	0	1	1

Figure 2. (a): A simple graph representing the interrelationships among six points in the 2-D space. Pairwise distances between every vertex and its neighbors are marked on the corresponding edges. (b) The H matrix representing the relationship between the vertices set and the hyperedge sets. The entry (v_i, e_j) is set to 1 if e_j contains v_i, or 0 otherwise. (c): The hypergraph which illustrates the complete relationships in the matrix H. The hyperedge weight is defined as the sum of reciprocals of all the pairwise distances in a hyperedge. (d) A hypergraph partition which is made on e_4.

v_5. Among all the hyperedges constructed in this example, e_1, e_2, e_3 correspond with the vertices subset $\{v_1, v_2, v_3\}$ and e_5, e_6 correspond with the vertices subset $\{v_4, v_5, v_6\}$ (Fig 2(c)). To measure the affinity among vertices in each hyperedge, we can define the hyperedge weight as the sum of reciprocals of all pairwise distances in a hyperedge.

In order to bipartition the hypergraph in Fig 2(c)), intuitively the hyperedges with the smallest affinity should be remove, and at the same time the hyperedges with larger affinities should be kept as many as possible. Since e_4 has the smallest hyperedge weight, a hypergraph cut could be made on it (Fig 2(d)) and classify $v_1, v_2, ..., v_6$ to two groups. How to obtain an optimal hypergraph partition for practical applications will be addressed in Section 4.

2.2. HyperGraph based Framework of Video Object Segmentation

In this paper, we develop a video object segmentation framework based on the hypergraph, shown in Fig 1. There contains three main components: the selection of the vertices, the hyperedge computation, and the hypergraph partition.

Inspired by [14], we initially over-segment the sequential images into small patches with consistent local appearance and motion information, as shown in Fig 3. Using the pixel values in the LUV color space, we get a 3-D features (l, u, v) for each pixel in the image sequence. With this feature, we adopt a multi-scale graph decomposition method [3] to do over-segmentation, for its ability to capture both the local and middle range relationship of image intensities, and its linear time complexity. This over-segmentation provides a good preparation for high-level reasoning of spatial-temporal relationship among the patches. We take these over-segmented patches as the vertices of the graph.

The computation of the hyperedges is actually equivalent to generating some attributes of the image patches. We treat the task of attribute assignment as a problem of binary classification according to different criteria. We first perform the spectral analysis in the spatio-temporal domain on two different motion cues, i.e., the optical flow and the appearance based motion profile, respectively. Then we cluster the data into two classes (2-way cut) on each spectral eigenvector respectively. Some representative 2-way cut results are finally selected to indicate the attributes of the patches. By analyzing the 2-way cut results, we assign different weights to different hyperedges. The details are described in Section 3.

After we obtain the vertices and the hyperedges, the hypergraph is built. We will use the hypergraph cut to partition the video into different objects, which will be addressed in Section 4.

Figure 3. A frame of oversegmentation results extracted from the rocking-horse sequence used in [14].

3. Hyperedge Computation

As mentioned above, the hyperedge is used to connect the vertices with same attribute value, so the task of hyper-

edges computation is actually to assign attributes for each image patch in spatio-temporal domain. In this section, we present to use spectral analysis for the attribute assignment. Before this, we will introduce how to represent the over-segmented patches in the spatio-temporal domain, and finally we discuss how to assign weights to the hyperedges.

3.1. Computing Motion Cues

We use the optical flow and the appearance based motion profile to describe the over-segmented patches in the spatio-temporal domain. The Lucas-Kanade optical flow method[7] is adopted to obtain the translations (x, y) of each pixel, and we indicate each pixel with the motion intensity $z = \sqrt{x^2 + y^2}$ and the motion direction $o = arctan(\frac{x}{y})$. We assume that pixels in the same patch have a similar motion, and then the motion of a patch can be estimated, $f^o = (u, d)$, by computing the weighted average of all the pixel motions in a patch:

$$u = \frac{1}{N}\sum_i \omega_i z_i, d = \frac{1}{N}\sum_i \omega_i o_i, \quad (1)$$

where N is the total number of pixels in a region, and w_i is the weight generated from a low-pass 2-D Gaussian centered on the centroid of the patch. u, d are the motion intensity and the motion angle of the patch, respectively. Since the motion of the pixels near the patch boundaries may be disturbed by other neighborhood patches, we discard the pixels near the boundaries (3 pixels to the boundaries).

Besides the optical flow, we also apply the appearance based motion profile to describe the over-segmented patches, inspired by the idea in [11]. Based on a reasonable assumption that the pixels in one patch have the same movement and color components and remain stable between consecutive frames too, the motion profile is defined as a measure of the probability distribution of image velocity to every patch based on appearance information. Let $I^t(X_i)$ denote the vector containing all the (l, u, v) pixel values of patch i centered at X, and denote $P_i(dx)$ as the probability of the image patch i at time t corresponding to another image patch $I^{t+1}(X_i + dx)$ at $t + 1$:

$$P_i(dx) = \frac{S_i(dx)}{\sum_{dx} S_i(dx)} \quad (2)$$

where $S_i(dx)$ denotes the similarity between $I^t(X_i)$ and $I^{t+1}(X_i + dx)$, which is based on on the SSD difference between $I^t(X_i)$ and $I^{t+1}(X_i + dx)$:

$$S_i(dx) = exp(-SSD(I^t(X_i), I^{t+1}(X_i + dx))). \quad (3)$$

3.2. Spectral Analysis for Hyperedge Computation

The idea of spectral analysis is based on an affinity matrix A, where $A(i, j)$ is the similarity between sample i and j [12] [8] [16]. Based on the affinity matrix, the Laplacian matrix can be defined as $L = D^{-\frac{1}{2}}(D - A)D^{-\frac{1}{2}}$, where D is the diagonal matrix $D(i, i) = \sum_j A(i, j)$. Then unsupervised data clustering can be achieved by doing eigenvalue decomposition of the Laplacian matrix. The popular way is to use the k-means method on the first several eigenvectors associated with the smallest non-zero eigenvalues [8] to get the final clustering result.

To set up the hyperedges, we perform the spectral analysis on the optical flow and the appearance based motion profile respectively. As in [12] [8] [16], only local neighbors are taken into account for the similarity computation. We defined two patches to be *spatial-temporal* neighbors if 1) in the same frame they are 8-connected or both their centroids fall into a ball of radius R, or 2) in the adjacent frames (± 1 frame in the paper) their regions are overlapped or 8-connected, as illustrated in Fig 1.

Denote the affinity matrices of the optical flow as A^o and the motion profile as A^p respectively. For the motion profile, we define the similarity between two neighbor patches i and j is defined as:

$$A^p(i, j) = e^{-\frac{dis(i,j)}{\sigma^p}},\ dis(i, j) = 1 - \sum_{dx} P_i(dx)P_j(dx), \quad (4)$$

where $dis(i, j)$ is defined as the distance between two patches i and j, and σ^p is constant computed as the standard deviation of $dis(i, j)$.

Based on the optical flow, the similarity metric between two neighbor patches i and j is defined as:

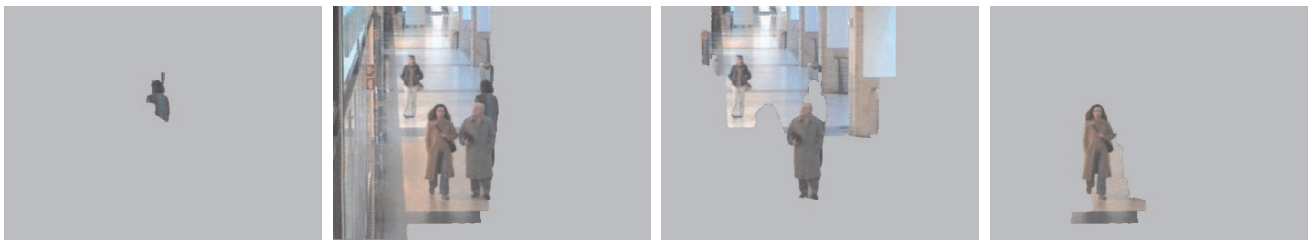
$$A^o(i, j) = e^{-\frac{\|f_i^m - f_j^m\|_2}{\sigma^o}}, \quad (5)$$

where σ^o is a constant computed as the standard deviation of $\|f_i^o - f_j^o\|_2$.

Based on A^o and A^p, we can compute the corresponding Laplacian matrix of L^o and L^p and their eigenvectors associated with the first k smallest non-zero eigenvalues respectively. Each of these eigenvectors may lead to a meaningful but not optimal 2-way cut result. Fig 4 shows some examples, where the patches without the gray mask are regarded as the vertices having the *attribute* value 1 and the patches with the gray mask having the *attribute* 0. A hyperedge can be formed by those vertices with same attribute values. With all the hyperedges, the complex relationship between the image patches can be represented by the hypergraph completely.

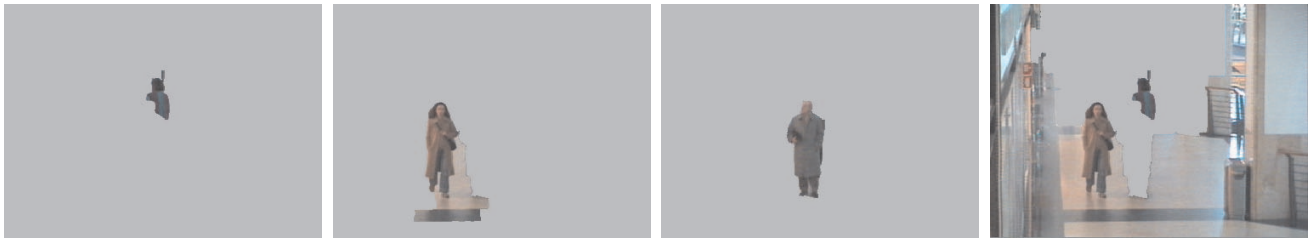
3.3. Hyperedge Weights

According to [12], the eigenvectors of the smallest k non-zero eigenvalues can be used for clustering. Then a nature idea is to choose the first k eigenvectors to compute the hyperedges, and weight those heyperedges with their



(1) (2) (3) (4)

Figure 4. Four binary partition results got by the first 4 eigenvectors computed from motion profile (for one frame of the sequence *WalkByShop1cor.mpg*, CAVIAR database.).



(1) 0.3506 (2) 0.2403 (3) 0.2378 (4) 0.0986

Figure 5. 4 binary partition results with largest hyperedge weights (for one frame of *WalkByShop1cor.mpg*). Obviously that the heperedge got from the 1st and 4th frames have a good description of objects we want to segment according to their importance. The computed hyperedge weights are shown below those binary images.

corresponding reciprocals of the eigenvalues. In our experiments, we find that the eigenvalues of the first k eigenvectors are very close and may not absolutely reflect the importance of the corresponding eigenvectors. In order to emphasize more important hyperedges which contain moving objects, larger weights should be assigned to them.

We impose the weights to the hyperedges from two different cues, w_H^o and w_H^p, by the following equations:

$$w_H^o = c^o \| f_1^o - f_0^o \|_2 \tag{6}$$

$$w_H^p = c^p dis(1,0) \tag{7}$$

where c^o and c^p are constant, and $dis(1,0)$ means the dissimilarity between two regions in the binary image with value 1 and 0, based on the first motion feature; f_1^o and f_0^o means the weighted motion intensity and direction of two regions in the binary image with value 1 and 0. Based on above definition, a larger weight is assigned to the binary frame whose two segmented regions have distinct appearance (motion) distributions.

In practice, we select the first 5 hyperedges with larger weights computed from appearance and motion respectively; and then proper c^p and c^o are chosen to let $\sum_{i=1}^{5} w_H^p(i) = 1$ and $\sum_{i=1}^{5} w_H^o(i) = 1$. In 5, we show the corresponding weight values under the binary attribute images. It is obvious that more meaningful attributes are assigned larger weights in our algorithm.

4. Video Object Segmentation by Hypergraph Cut

Given a vertex subset $S \subset V$, denote S^c as the compliment of S. The hyperedge boundary $\partial S := \{e \in E | e \cap S \neq \emptyset, e \cap S^c \neq \emptyset\}$ is a set of the hyperedges to partition the hypergraph G into two parts S and S^c [19]. Then a two-way partition for the hypergraph could be defined as:

$$Hcut(S, S^c) := \sum_{e \in \partial S} w(e) \frac{|e \cap S||e \cap S^c|}{\delta(e)} \tag{8}$$

where $\delta(e) = \sum_{v \in V} H(v, e)$ is the degree of the hyperedge e. The definition of the *hypergraph partition* given above can be understood as the *volume* of the hyperedge boundary ∂S. Similar to the *normalized cut* [**?**], the two-way normalized hypergraph partition can be defined to avoid the bias of unbalanced partitioning:

$$NHcut(S, S^c) := Hcut(S, S^c) \left(\frac{1}{vol(S)} + \frac{1}{vol(S^c)} \right)$$

where $vol(S)$ is the volume of S, i.e., $vol(S) = \sum_{v \in S} d(v)$. Similar to the normalized cut, minimizing Equation 4 is an NP-complete problem, and it can be relaxed into a real valued optimization problem [19]. The theoretical solution of this real value problem is the eigenvector associated with the smallest non-zeros eigenvector of the *hypergraph Laplacian matrix* $\Delta = I - D_v^{-\frac{1}{2}} H W D_e^{-1} H^T D_v^{-\frac{1}{2}}$. As in [8], to make a multi-way classification of vertices in a

hypergraph, we take the first several eigenvectors with nonzeros eigenvalues of Δ as the indicators (we take 3 in this paper), and then use a k-means clustering algorithms on the formed eigenspace to get final clustering results.

5. Experiments

5.1. Experimental Protocol

To evaluate the performance of our segmentation method based on the hypergraph cut, we compare it with three clustering approaches based on the simple graph, i.e., the conventional simple graph with pairwise relationship. In these three approaches, we measure the similarity between two over-segmented patches using (1) the optical flow, (2) the motion profile, and (3) both the motion cues. The similarity matrix for (1) and (2) just follow Eq. 5 and Eq. 4. For (3), the similarity is defined as follows:

$$A(i,j) = e^{-\frac{\|f_i^m - f_j^m\|_2}{\sigma^o} - \frac{dis(i,j)}{\sigma^p}}, \quad (9)$$

where σ^o and σ^p are constants. Notice that σ values in Eq 9, Eq. 4 and Eq 5 are all tuned to get the best segmentation results for both the hypergraph based and the simple graph based methods for comparison. Then corresponding Laplacian matrix of these three approaches can be computed accordingly and the k-means algorithm can be performed on the first n eigenvectors with nonzero eigenvalues. In our experiment, we choose $n = 10$ for all these three simple graph based methods.

5.2. Results on Videos under Different Conditions

We first report the experiments on the rocking-horse sequence and the squirrel sequence used in [14]. We choose them because the movement of objects in these two sequences are very subtle and their backgrounds are cluttered and similar to the objects. Fig 6 and 7 show the ground truth frames, the results of three simple graph based methods, and the results of hypergraph cut for these two sequences. To illustrate a distinctive comparison with the ground truth, we plot the red edge of the segmented patches in our results. Compared with the results in [14] and the simple graph based methods, in both sequences our method gives more meaningful segmentation results for the foreground objects, although a few local details are lost in the squirrel sequence. In all these figures, the number of cluster classes is set to 2 (K=2).

We also compare four algorithms on the image sequences in which the video object has complicated movements. The sequence shown in Fig 8 (*Walk1.mpg*, from CAVIAR database) contains a person browsing back and forth and rotating during the course of his movement. In this example, we cluster the scene to two classes (K=2) too. From Fig 8, we can observe that our method can give very accurate segmentation result for the moving objects, in spite of the small perturbation in the left corners of this sequence. However, the simple graph based methods can not completely extract the moving person from background and some unexpected small patches are classified into the moving objects.

In the real world, the video objects may be occluded or interacted with each other during their movements. We also test the proposed method on such examples with occlusion. In Fig 9, four algorithm are compared on a running-car sequence with an artificial occlusion, in which the hypergraph cut extracts the car and the pedestrian from the background accurately, while the simple graph based methods can extract the car or the pedestrian. In the sequence shown in Fig 10 (*WalkByShop1front.mpg*, from CAVIAR database), a couple walk along the corridor, and another person moves to the door of the shop hastily and is occluded by the couple during his moving. When we set K=2, the person with largest velocity of movement is segmented. When we set K=3, K=4 and K=5, three primary moving objects are extracted one by one only with a small patch between the couple, which is caused by the noise of motion estimation. For the simple graph based methods, we give the best case (the best result under different K). For $K > 3$, simple graph based methods usually give very cluttered and not meaningful results. For the simple graph based methods using the motion profile or the optical flow, K =2 can give the most meaningful results, and K =3 can give a good extraction of the couple for the simple graph method using both motion cues.

In Table 1, the average segmentation accuracy and segmentation error are estimated and compared on the experimental frames of all the image sequences. The segmentation accuracy for one frame is defined as the number of 'true positive' pixels (the true positive area) divided by the number of the ground truth pixels(the ground truth area). The segmentation error for one frame is defined as the number of 'false positive' pixels (the false positive area) divided by the number of the ground truth pixels(the ground truth area).

6. Conclusions

In this paper, we proposed a framework of video object segmentation , in which a novel graph structure – hypergraph is used to represent the complex relationship of the images. We first used the multi-scale graph decomposition method to over-segment the images and took the oversegmented image patches as the vertices of the hypergraph. The spectral analysis was performed on two motion cues respectively to set up the hyperedges, and the spatio-temporal information is integrated by the hyperedges. Furthermore, a weighting procedure is discussed to put larger weights on more important hyperedges. In this way, the task of video object segmentation is transferred into a hypergraph partition problem which can be solved by the hypergraph cut

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

Figure 6. Segmentation results for the 8th frame of the rocking-horse sequence. (a) The ground truth, (b) the result by the simple graph based segmentation using optical flow, (c) the result by the simple graph based segmentation using motion profile, (d) the result by the simple graph based segmentation using both motion cues, and (e) the result by the hypergraph cut.

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

Figure 7. Segmentation results for the 4th frame of the squirrel sequence. (a) The ground truth, (b) the result by the simple graph based segmentation using optical flow, (c) the result by the simple graph based segmentation using motion profile, (d) the result by the simple graph based segmentation using both motion cues, and (e) the result by the hypergraph cut.

algorithm. The effectiveness of the proposed method is demonstrated by extensive experiments on nature scenes. Since our algorithm is a open system, in the future work, we will add more motion or appearance cues (such as texture information, the occlusions between frames) into our framework to construct more hyperedges and further improve the accuracy of these results.

References

[1] S. Agarwal, J. Lim, L. Zelnik-Manor, P. Perona, D. Kriegman, and S. Belongie. Beyond pairwise clustering. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2*, pages 838–845, Washington, DC, USA, 2005.

[2] D. Chung, W. J. MacLean, and S. Dickinson. Integrating region and boundary information for improved spatial coherencein object tracking. pages 3, CVPRW '04: Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition Workshop Volume 1, 2004.

[3] T. Cour, F. Benezit, and J. Shi. Spectral segmentation with multiscale graph decomposition. pages II: 1124–1131, CVPR2005.

[4] D. DeMenthon and R. Megret. Spatio-temporal segmentation of video by hierarchical mean shift analysis. CVPR 2002.

[5] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.

[6] S. Lazebnik and J. Ponce. The local projective shape of smooth surfaces and their outlines. *Int. J. Comput. Vision*, 63(1):65–83, 2005.

[7] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision (ijcai). In *Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI '81)*, pages 674–679, April 1981.

[8] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. NIPS2001.

[9] A. S. Ogale, C. Fermuller, and Y. Aloimonos. Motion segmentation using occlusions. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(6):988–992, 2005.

[10] A. Sethi, D. Renaudie, D. Kriegman, and J. Ponce. Curve and surface duals and the recognition of curved 3d objects from their silhouettes. *Int. J. Comput. Vision*, 58(1):73–86, 2004.

[11] J. Shi and J. Malik. Motion segmentation and tracking using normalized cuts. pages 1154–1160, ICCV1998.

[12] J. Shi and J. Malik. Normalized cuts and image segmentation. 22(8):888–905, August PAMI 2000, vol 8.

[13] P. Smith, T. Drummond, and R. Cipolla. Layered motion segmentation and depth ordering by tracking edges. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(4):479–494, 2004.

[14] A. Stein, D. Hoiem, and M. Hebert. Learning to find object boundaries using motion cues. October ICCV 2007.

[15] R. Vaillant and O. Faugeras. Using extremal boundaries for 3-d object modeling. 14(2):157–173, February 1992,PAMI, vol2.

[16] Y. Weiss. Segmentation using eigenvectors: A unifying view. In *ICCV (2)*, pages 975–982, 1999.

[17] J. Wills, S. Agarwal, and S. Belongie. What went where. pages I: 37–44, CVPR2003.

[18] J. Xiao and M. Shah. Accurate motion layer segmentation and matting. pages 698–703, CVPR2005.

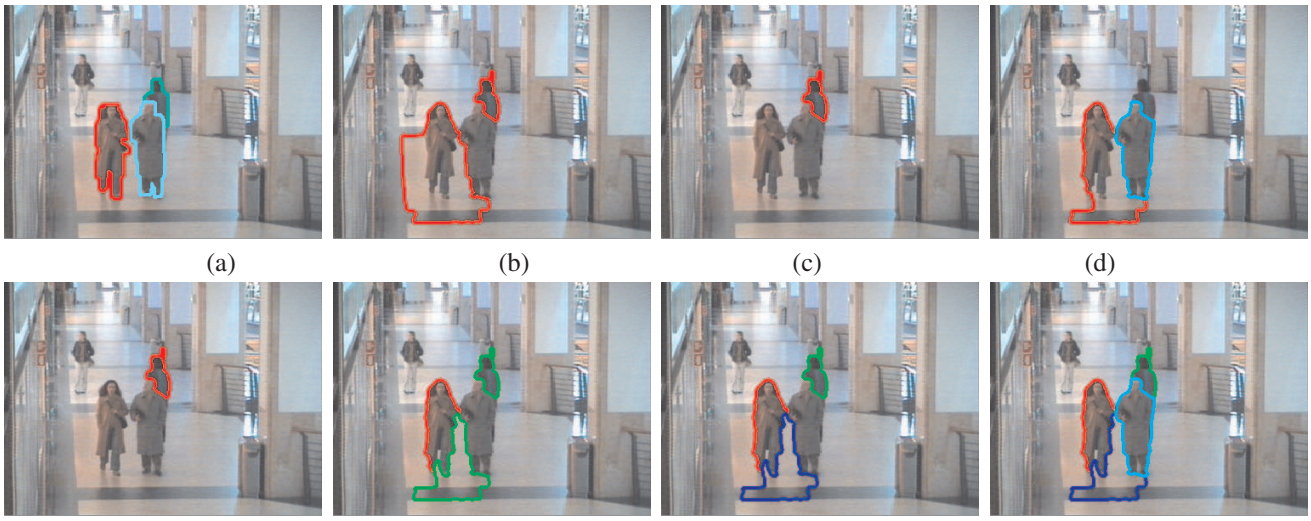
[19] D. Zhou, J. Huang, and B. Schlkopf. Learning with hypergraphs: Clustering, classification, and embedding. pages 1601–1608, NIPS2007.

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

Figure 8. Segmentation results for one frame of *Walk1.mpg*, CAVIAR database. (a) The ground truth, (b) the result by the simple graph based segmentation using optical flow, (c) the result by the simple graph based segmentation using motion profile, (d) the result by the simple graph based segmentation using both motion cues, and (e) the result by the hypergraph cut.

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

Figure 9. Segmentation results for the 16th frame of the car running sequence with occlusion. (a) The ground truth, (b) the result by the simple graph based segmentation using optical flow, (c) the result by the simple graph based segmentation using motion profile, (d) the result by the simple graph based segmentation using both motion cues, and (e) the result by the hypergraph cut.



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

(e) (f) (g) (h)

Figure 10. Segmentation results for one frame of the *WalkByShop1front.mpg*, different colors denote different clusters in each sub-figure. (a) The ground truth, (b) the result by the simple graph based segmentation using optical flow (K=2), (c) the result by the simple graph based segmentation using motion profile (K=2), (d) the result by the simple graph based segmentation using both motion cues (K=3), (e) the result by the hypergraph cut (K=2), (f) the result by the hypergraph cut (K=3), (g) the result by the hypergraph cut (K=4), and (h) the result by the hypergraph cut (K=5).

Sequence Name	MP	OP	MP+OP	Hypergraph Cut
Rocking-horse	0.87/0.02	0.96/0.76	0.96/0.92	0.91/0.02
Squirrel	0.91/0.86	0.89/1.32	0.72/0.02	0.89/0.01
Walk1	0.92/15.3	0.92/6.4	0.92/12.7761	0.94/0.03
car running	0.14/0.03	0.82/0.02	0.82/3.22	0.89/0.03
WalkByShop1front	0.32/0.47	0.56/0.81	0.79/0.66	0.84/0.37

Table 1. Average accuracy/error for all the experimental frames of every sequence, where MP means simple graph method by the motion profile, OP means simple graph method by the optical flow and MP+OP means the simple graph method using both cues. Mention that for *WalkByShop1front.mpg* we only consider the case when K=4.