Unsupervised Spatio-Temporal Segmentation with Sparse Spectral Clustering

Mahsa Ghafarianzadeh[1]
masa@gwu.edu
Matthew B. Blaschko[2]
matthew.blaschko@inria.fr
Gabe Sibley[1]
gsibley@gwu.edu

[1] Computer Science Department
The George Washington University
Washington DC, USA

[2] École Centrale Paris
INRIA Saclay
Châtenay-Malabry, France

Abstract

Spatio-temporal cues are powerful sources of information for segmentation in videos. In this work we present an efficient and simple technique for spatio-temporal segmentation that is based on a low-rank spectral clustering algorithm. The complexity of graph-based spatio-temporal segmentation is dominated by the size of the graph, which is proportional to the number of pixels in a video sequence. In contrast to other works, we avoid oversegmenting the images into super-pixels and instead generalize a simple graph based image segmentation. Our graph construction encodes appearance and motion information with temporal links based on optical flow. For large scale data sets naïve graph construction is computationally and memory intensive, and has only been achieved previously using a high power compute cluster. We make feasible for the first time large scale graph-based spatio-temporal segmentation on a *single core* by exploiting the sparsity structure of the problem and a low rank factorization that has strong approximation guarantees. We empirically demonstrate that constructing the low rank approximation using a subset of pixels (30%-50%) achieves performance exceeding the state-of-the-art on the Hopkins 155 dataset, while enabling the graph to fit in core memory.

1 Introduction

The analysis of videos is a central task in computer vision. Videos encode useful information in appearance, motion cues, and temporal continuity, which can be leveraged to improve performance on a wide range of applications. Spatio-temporal segmentation is defined as the problem of partitioning a sequence of images into coherent regions using motion and appearance cues. This is an essential step for different tasks such as tracking, object detection and video analysis. Spatio-temporal segmentation is a challenging problem because there are different factors that may change throughout the sequence such as camera motion, scale, illumination, viewpoint, deformation of objects, occlusion, etc.

In this paper, we improve on a family of methods that generalize graph based image segmentation to the spatio-temporal case. Such methods build a graph based on an affinity

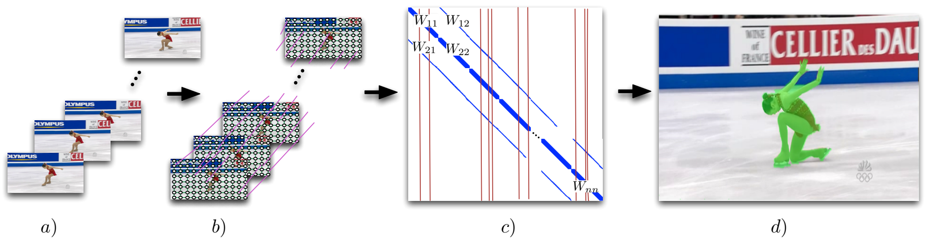

Figure 1: Approach overview: given a sequence of images a) we construct a graph b) connecting pixels in a neighborhood and also to their temporal correspondences. This is represented as a very large sparse matrix, from which we select a random subset of columns (which correspond to pixels) – only the randomly selected pixels are used for graph and matrix construction. d) we employ spectral clustering segmentation based on efficient and accurate low rank factorization based on the Nyström method to approximate the graph Laplacian.

matrix with connectivity between neighboring pixels, as well as temporal links computed using optical flow. While such techniques have been applied previously, they have either been based on a sparse set of pixels, super-pixel segmentation, or have made use of extensive computing power to exhaustively compute a dense graph based segmentation. The central contribution of this paper is to introduce a set of strategies that enable us to compute a dense graph based segmentation using a single processor and fitting in core memory. This is done primarily through two innovations: (i) we exploit the sparsity structure of the spatio-temporal graph, and (ii) we make use of an efficient and accurate low rank factorization based on the Nyström method to approximate the graph Laplacian in a spectral clustering approach.

We randomly sample a subset of pixels in the sequence of images and build a sparse approximation of the full affinity matrix. The sampled pixels are the nodes of the similarity matrix and edges encode appearance and motion cues. Next we perform spectral clustering on this subset similarity matrix as it provides optimal global solutions and use the solution to segment all the pixels. Our results show that not all of the pixels contain meaningful information about images, and just a subset of pixels can be a good representation of the entire scene.

2 Related work

A number of video segmentation techniques has been proposed in recent years. Existing video segmentation approaches may differ in the assumptions they make and can be divided into different categories. One category include layered motion segmentation techniques [12, 15, 17] which assume that a scene can be represented with a number of overlapping layers, and each layer has a motion model that can be obtained by some reasoning process and therefore is capable of capturing moving objects in the scene. One drawback is that they may not perform very well on non-textured regions. However, they are capable of segmenting objects when there is occlusion in the scene due to their global motion model.

Another category of video segmentation techniques which is based on sparse number of locations with point trajectories [1, 13]. For example in [13], a robust subspace separation method has been developed that uses Agglomerative Lossy Compression (ALC) for motion segmentation. The advantage of this method is that it can successfully deal with corrupted or incomplete trajectories where most of other methods in this category fail. In [1], long term point trajectories based on dense optical flow are clustered into temporally coherent moving segments. However, a major drawback is that these techniques only consider up to 4% of the pixels in the video which is not enough for segmenting the entire video including non-moving objects. Hence, compared to dense video segmentation, the number of labeled points obtained by these sparse methods is considerably smaller.

There are also spatio-temporal segmentation methods which represent the video in a spatio-temporal domain and generalize from image segmentation. For example, in segmentation by hyper-graph cuts [8], first the image is over-segmented into regions, then the regions are taken as the nodes of a hyper-graph that represents the relationship among regions and finally the hyper-graph is partitioned to get the video segmentation. Also in hierarchical graph-based segmentation [7], a volumetric video graph is over-segmented into spatio-temporal regions, and then a region graph is constructed by merging the similar regions obtained from previous over-segmentation. This process is repeated iteratively to create a hierarchical segmentation of the video. Hierarchical mean shift analysis [4], is another method that maps pixels of a video stack into a 7D feature space (three color, two motion and two position component) and then uses mean shift repeatedly to achieve hierarchical clustering of the points in the video. In [9], the concept of spatio-temporal closure over a graph of temporal super-pixels has been introduced which is an extension of closure detection techniques in 2D images. Motion aware hierarchical image segmentation (MAHIS) [6], is another technique that obtains super-pixels from different variety of pixel-based affinities such as brightness,color and texture and then in the next step obtains video segmentation from a variety of super-pixel-based affinities. Another recent work [2], finds temporally consistent super-pixels in a video using a generative probabilistic model.

Spectral clustering has been used for image and video segmentation, and our work is closely related to [5, 14]. We aim to partition a graph that encodes appearance and motion similarity in a sequence of images and obtain robust perceptual group which are coherent in time. However, spectral clustering is computationally expensive which makes it often impractical for large scale data set such as spatio-temporal data. In [5], the Nyström method has been applied as a numerical solution for speeding up the spectral clustering. They consider the video as a space time volume and partitioned the volume into coherent segments. In [16], they gained more computational power by parallelism in GPUs and perform video segmentation more efficiently. We instead use an algorithm proposed by [10], which is a time and space efficient spectral clustering algorithm for estimating the rank k approximation of normalized Laplacian. This forms the core of our method to perform spatio-temporal segmentation in memory on a single CPU.

3 Approach

3.1 Creating the Graph

Given an image I, we create a graph $G = (V, E, W)$, where the graph nodes V are the pixels in the image and are connected by edge E if they are within distance r from each other. As discussed in [3], in general choosing a smaller r makes the segmentation faster and choosing a larger r helps with propagating local grouping cues across larger neighborhood which is useful for detecting faint contours in the image. W measures the similarity of pixels connected by an edge. We define W as the following:

$$W_{ij} = \exp \frac{-d^2(s_i, s_j)}{\sigma_i \sigma_j}$$

where $W_{ij} = 0$ for i =j, and s_i denotes pixel color and $d(s_i, s_j)$ is the Euclidean distance. σ is a local scaling parameter [18] which takes into account the local statistics of the neighborhood around pixels i and j. Local scaling parameter is defined by:

$$\sigma_i = d(s_i, s_k)$$

where s_k is the K'th neighbor of pixel i. In order to extend this to video, we make use of optical flow and add temporal motion information to the graph. We use optical flow [11] to compute the motion vectors between frames. Then we connect pixel (x, y) in frame t to its 9 neighbors along the backward flow (u, v) in frame $t-1$, *e.g.* $(x + u(x,y) + \delta_x, y + v(x,y) + \delta_y)$ for $\delta_x, \delta_y \in \{-1, 0, 1\}$. The similarity matrix for the video is a sparse symmetric block diagonal matrix of the size n = *number of frames* $\times$ *number of pixels in one frame*.

3.2 Partitioning the Graph

3.2.1 Spectral Clustering

Spectral clustering has been widely used for partitioning the similarity graphs. Given the similarity matrix of n points,W, first the normalized similarity matrix $L = D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$ is computed, where D is the diagonal degree matrix defined as $D_{ii} = \sum_{j=1}^{n} W_{ij}$. Then it finds the k largest eigenvectors of L and form $V = [v_1, ..., v_k] \in R^{n \times k}$. Next V will be re-normalized row-wise to have unit length, U, and then k-means is applied to cluster the rows of U. Finally, each pixel s_i will be assigned to the corresponding cluster for row i of U. However, the $O(n^3)$ complexity of eigen decomposition of L and also storing the similarity matrix often makes it expensive and impractical when n is large.

3.2.2 Nyström Method

The Nyström method used by [5] is a popular technique to tackle the mentioned problem by approximating a low-rank matrix of W, since we only need a number of eigenvectors. Given a symmetric matrix $W \in R^{n \times n}$, $m \ll n$ columns of the matrix will be sampled. For simplicity, we assume that these sampled columns are the first m columns of W. Matrix W can be represented as follows:

$$W = \begin{pmatrix} A & B \\ B^T & C \end{pmatrix}$$

where $A \in R^{m \times m}$, $B \in R^{n-m \times m}$, and $C \in R^{n-m \times n-m}$. The Nyström approximation of W is defined by:

$$\hat{W} = \begin{pmatrix} A \\ B^T \end{pmatrix} A^{-1} \begin{pmatrix} A & B \end{pmatrix}$$

Then if A is positive definite, we can define $S = A + A^{-\frac{1}{2}} B B^T A^{-\frac{1}{2}}$ and perform singular value decomposition (SVD) on S to get $S = U_S \Lambda_S U_S{}^T$. The approximated $\hat{L}$ then can be defined as $\hat{L} = V \Lambda V^T$ and $V^T V = I$, where

$$V = \begin{pmatrix} A \\ B^T \end{pmatrix} A^{\frac{-1}{2}} U_S \Lambda_S{}^{-\frac{1}{2}}$$

But if A is indefinite, a more complicated process is required. We define $Z = \bar{U_S}\Lambda^{\frac{1}{2}}$ and $\bar{U_S}^T = \left[\begin{array}{cc} U_S^T & \Lambda_S^{-1}U_S^T B \end{array} \right]$ so $\hat{L} = ZZ^T$. If singular value decomposition of $ZZ^T = F\Sigma F^T$, then $V = ZF\Sigma^{-\frac{1}{2}}$ has the orthogonalized eigenvectors of $\hat{L}$. This algorithm is expensive for large scale data sets, since the time complexity of this algorithm is $O(nm^2 + m^3)$ and it requires keeping A and $\begin{pmatrix} A \\ B^T \end{pmatrix}$ in the memory. Hence, we are using a time and space efficient spectral clustering via column sampling [10], that is similar to Nyström method, but with a further rank-k approximation of the normalized Laplacian using the sampled sub-matrix of the similarity matrix. This algorithm has shown promising results, since it reduces the time and space complexity of Nyström method and also it is able to recover all the degree information of the selected points. Here we briefly review the algorithm proposed in [10]. First we form the sampled sub-matrix $A \in R^{m\times m}$ and compute the degree matrix D_* for A. Next, we define $M_* \in R^{m\times m}$ as

$$M_* = D_*^{-\frac{1}{2}} A D_*^{-\frac{1}{2}}$$

and $S_* \in R^{m\times m}$ based on the rank-k approximation $M_{*,k}$ of M_* as

$$S_* = D_*^{-\frac{1}{2}} M_{*,k}^{-1} D_*^{-\frac{1}{2}}$$

$M_{*,k}$ is computed by performing eigen decomposition of M_* and find the k largest eigenvalues Λ and the corresponding eigenvectors V. Finally the approximated $\hat{W}$ is obtained by

$$\hat{W} = \begin{pmatrix} A \\ B^T \end{pmatrix} S_* \begin{pmatrix} A \\ B^T \end{pmatrix}^T$$

By having $M_{*,k} = V\Lambda V^T$ and replacing it in above formulas, we get $\hat{W} = Q\Lambda Q^T$, where

$$Q = \begin{pmatrix} A \\ B^T \end{pmatrix} D_*^{-\frac{1}{2}} V \Lambda^{-1}$$

Lastly, rank-k approximation of L is obtained by following

$$\hat{L}_k = \hat{D}^{-\frac{1}{2}} \hat{A} \hat{D}^{-\frac{1}{2}} = U\Lambda U^T$$

where $U = \hat{D}^{-\frac{1}{2}} Q$ and $\hat{D}$ is the degree matrix of $\hat{W}$. The time complexity of this algorithm is $O(nmk)$ and there is no need to store large similarity matrix W or its sampled columns in the memory. Also we are using the proposed inexpensive algorithm by [10] to orthogonalized estimated eigenvectors U.

3.3 Post Processing

After performing spectral clustering on the similarity graph and obtaining the clusters, we first quantize each cluster into 256 bins (16 bins for each channel) and compute the *RGB* histogram. Then we find the Euclidean distance between adjacent clusters and merge them repeatedly if their similarity is more than a threshold τ to achieve the final segmentation.

Table 1: Comparison of dense and sparse segmentation methods

Name	Density	Overall Error	Overall Accuracy	Average Error	Over-segmentation	Extracted Objects
First 10 frames (26 sequences)						
Ours (Sample:50%)	100%	5.06%	94.94%	22.31%	6.12	22
Sundaram *et al*[16]	100%	6.97%	93.03%	19.72%	7.15	20
Brox *et al*[1]	3.34%	7.75%	3.08%	25.01%	0.54	24
Rao *et al*[13] corrupted	2.98%	7.88%	2.74%	24.05%	0.15	26
Rao *et al*[13] incomplete	3.34%	11.20%	2.97%	26.73%	0.54	19
First 50 frames (15 sequences)						
Ours(Sample:50%)	100%	7.63%	92.37%	32.24%	14.9	5
Sundaram *et al*[16]	100%	8.42%	91.58%	28.76%	13.0	5
Brox *et al*[1]	3.27%	7.13%	3.04%	34.76%	0.53	9
Rao *et al*[13] corrupted	1.53%	7.91%	1.41%	42.13%	0.36	8
Rao *et al*[13] incomplete	3.27%	16.42%	2.73%	49.05%	6.07	2
First 200 frames (7 sequences)						
Ours(Sample:50%)	100%	13.24%	86.76%	47.83%	5.3	6
Sundaram *et al*[16]	100%	15.46%	84.54%	45.05%	5.86	3
Brox *et al*[1]	3.43%	7.64%	3.17%	31.14%	3.14	7
Rao *et al*[13] corrupted	0.20%	0.00%	0.20%	74.52%	0.40	1
Rao *et al*[13] incomplete	3.43%	19.33%	2.77%	50.98%	54.57	0
Grundmann *et al*[7]	100%	20.77%	79.23%	40.37%	10.42	0

4 Results

To evaluate the performance of our segmentation method, we run several experiments and provide quantitative and qualitative results. We compare our method against motion segmentation techniques such as [1, 13], which cluster point trajectories in the sequence of images and also other dense video segmentation techniques such as [7, 16]. In all of our experiments, we randomly select a subset sample of size 30%-50% for approximating the similarity matrix. The quality of our segmentation highly relies on the selected number of clusters for k-means and also distance r for computing pixel similarities. Choosing larger numbers for these parameters makes the overall segmentation better. However, the disadvantage is that it creates unsmooth boundaries and captures most of the undesired details in the scene and also increases the size of matrix rapidly. We found that by choosing a distance size of 11-15 and cluster size of 300-500, we achieve reasonable performance.

We use an annotated dataset by [1] for motion segmentation. This dataset includes 26 sequences which is annotated sparse in time and dense in space. We ran our method on three scenarios and show the numerical result in Table 1. The first section shows the segmentation result for the first 10 frames of all 26 available sequences in the dataset, the second section shows the segmentation result for the first 50 frames of 15 selected sequences in the dataset and the third scenario shows the result for the first 200 frames of 7 selected sequences. The density shows the percentage of pixels for which a label is reported. The overall error is the number of mislabeled pixels over the total number of labeled pixels. the overall accuracy is the number of correctly labeled pixels over the total number of pixels. The average error is similar to the overall error but computed separately for each region first and then the average across all the regions is reported. The over-segmentation error indicates the number of clusters that needs to be merged to fit the ground truth regions. Also the number of regions with less than 10% error is reported as the extracted objects.

In this experiment, we aimed to compare our method against the method in [16]. The reason is that in contrast with their method which achieves good performance by using GPUs and heavily depending on parallelization, we exploit numerical algorithms to make our method fast and efficient. The density of both methods is 100% but we achieved comparable errors while using just a subset of pixels (30%-50%) to label all of the pixels as

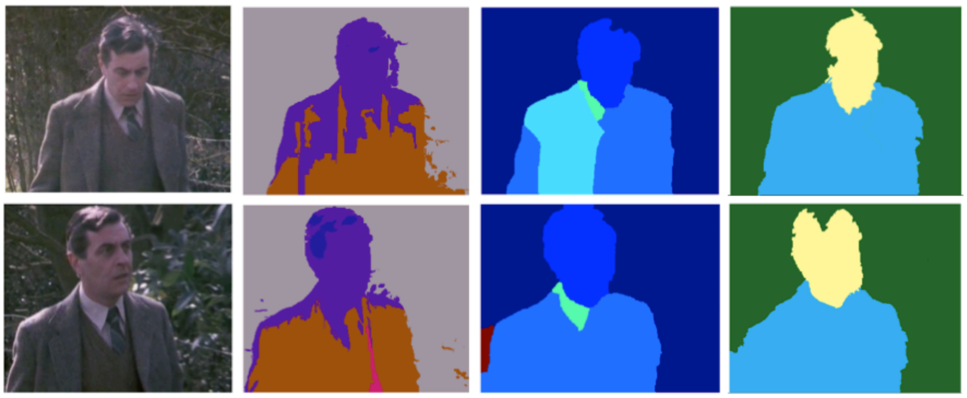

Figure 2: Comparison of our technique vs motion segmentation by [16] and [7]. Left-to-right shows video frames 1 and 50. Second row: results from Hierarchical graph segmentation [7]; third segmentation with the full graph using a cluster of GPUs [16]; fourth row, our result runs on a single core, does not employ contour smoothing or texture similarity.

compared with their method that uses all the pixels for clustering. Compared to the motion segmentation methods [1, 13], our over-segmentation error is higher but that is due to the fact that these methods only use sparse number of points to label about 4% of the data while we are labeling all the pixels and achieve better accuracy.

5 Discussion

There are several avenues for future research based on the methods presented in this work. Sparse spectral clustering algorithms are designed to be agnostic to the problem domain. However, we have a substantial amount of domain knowledge we can exploit to expedite the clustering.

The matrix factorization employed uses uniform sampling to select columns. However, selective sampling strategies generally decrease the approximation error for a given number of columns. As the error rate of our method is comparable to using 100% of the columns, this indicates that we may achieve an even higher savings in memory using selective sampling. An additional likely source of improvement is through the use of similarity measures based on feature functions that encode texture or other information sources in addition to color.

6 Conclusions

In this work, we have demonstrated a novel method for spatio-temporal segmentation of dense pixel trajectories based on spectral clustering. We found that fully connecting pixels to their spatial neighbors within a given radius is an effective strategy for improving segmentation accuracy. Additionally, we use optical flow to more accurately compute temporal connectivity than a simple method based on an interpretation of the video sequence as a 3D volume. In contrast to previous work, we do not resort to super-pixel segmentation

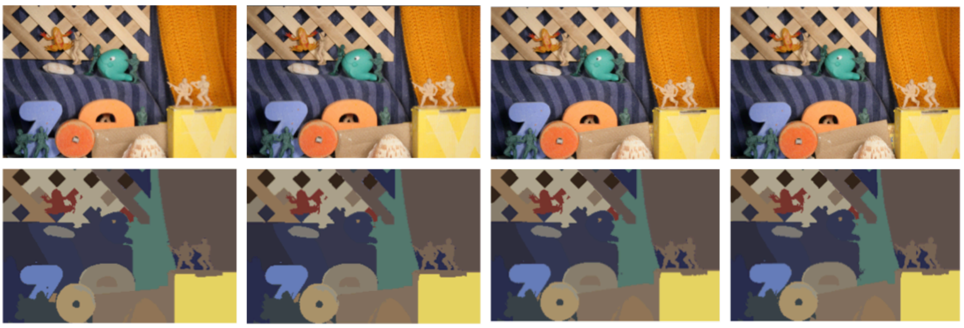
Figure 3: Results form the Army Men sequence. Notice the fine detail around the top-right gun figure.

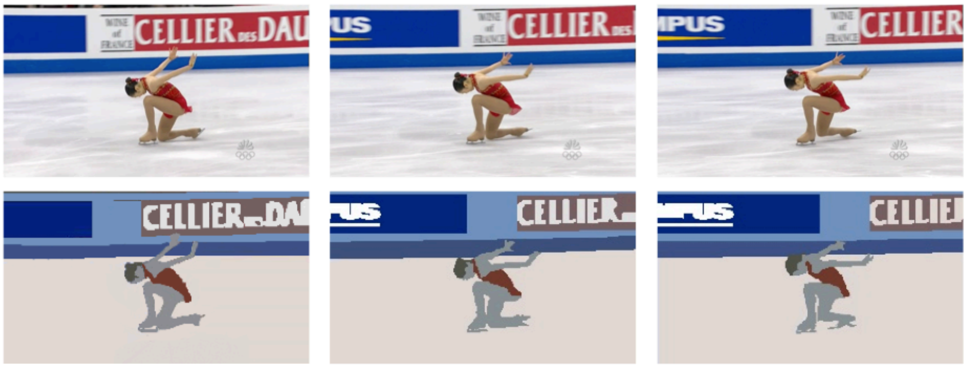
Figure 4: Results from Ice-skater Yu-Na Kim, 2009 World Championships, ©2009 NBC Olympics.

to achieve computational tractability and memory efficiency. Instead, we exploit the natural sparsity structure of the graph, and employ a low rank approximation of the Laplacian closely related to the Nyström method. We have found that sampling 30-50% of the pixels to index columns of the low rank approximation leads to comparable performance with a method that uses 100% of the columns. This strategy results in a spectral clustering method that can run on a single processor with the graph representation fitting in core memory. For instance, a 100 frame 640x480 sequence naively requires 6.8GB of memory, while with our method we are able to reduce this to 2.1GB and still retain accuracy. We have demonstrated the effectiveness of the approach on the Hopkins 155 data set, where we have achieved the best reported results for dense segmentation using an order of magnitude less computation than [16].

References

[1] Thomas Brox and Jitendra Malik. Object segmentation by long term analysis of point trajectories. In *Computer Vision–ECCV 2010*, pages 282–295. Springer, 2010.

[2] Jason Chang, Donglai Wei, and John W Fisher III. A video representation using temporal superpixels. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 2051–2058. IEEE, 2013.

[3] Timothée Cour, Florence Bénézit, and Jianbo Shi. Spectral segmentation with multiscale graph decomposition. In *In CVPR*, pages 1124–1131, 2005.

[4] Daniel DeMenthon and Remi Megret. *Spatio-temporal segmentation of video by hierarchical mean shift analysis*. Computer Vision Laboratory, Center for Automation Research, University of Maryland, 2002.

[5] Charless Fowlkes, Serge Belongie, Fan Chung, and Jitendra Malik. Spectral grouping using the nystrom method. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(2):214–225, 2004.

[6] Fabio Galasso, Roberto Cipolla, and Bernt Schiele. Video segmentation with superpixels. In *Computer Vision–ACCV 2012*, pages 760–774. Springer, 2013.

[7] Matthias Grundmann, Vivek Kwatra, Mei Han, and Irfan Essa. Efficient hierarchical graph-based video segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2141–2148. IEEE, 2010.

[8] Yuchi Huang, Qingshan Liu, and Dimitris Metaxas.] video object segmentation by hypergraph cut. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1738–1745. IEEE, 2009.

[9] Alex Levinshtein, Cristian Sminchisescu, and Sven Dickinson. Spatiotemporal closure. In *Computer Vision–ACCV 2010*, pages 369–382. Springer, 2011.

[10] Mu Li, Xiao-Chen Lian, James T Kwok, and Bao-Liang Lu. Time and space efficient spectral clustering via column sampling. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 2297–2304. IEEE, 2011.

[11] Ce Liu. *Beyond pixels: exploring new representations and applications for motion analysis*. PhD thesis, Citeseer, 2009.

[12] Abhijit S Ogale, Cornelia Fermuller, and Yiannis Aloimonos. Motion segmentation using occlusions. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(6):988–992, 2005.

[13] Shankar R Rao, Roberto Tron, René Vidal, and Yi Ma. Motion segmentation via robust subspace separation in the presence of outlying, incomplete, or corrupted trajectories. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[14] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):888–905, 2000.

[15] Paul Smith, Tom Drummond, and Roberto Cipolla. Layered motion segmentation and depth ordering by tracking edges. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(4):479–494, 2004.

[16] Narayanan Sundaram and Kurt Keutzer. Long term video segmentation through pixel level spectral clustering on gpus. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 475–482. IEEE, 2011.

[17] Jiangjian Xiao and Mubarak Shah. Accurate motion layer segmentation and matting. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 698–703. IEEE, 2005.

[18] Lihi Zelnik-Manor and Pietro Perona. Self-tuning spectral clustering. In *NIPS*, volume 17, page 16, 2004.