Motion Trajectory Segmentation via Minimum Cost Multicuts

Margret Keuper[1], Bjoern Andres[2], and Thomas Brox[1]

[1] *Computer Vision Group, University of Freiburg, Germany*
[2] *Max-Planck-Institute of Informatics, Saarbruecken, Germany*

Abstract

For the segmentation of moving objects in videos, the analysis of long-term point trajectories has been very popular recently. In this paper, we formulate the segmentation of a video sequence based on point trajectories as a minimum cost multicut problem. Unlike the commonly used spectral clustering formulation, the minimum cost multicut formulation gives natural rise to optimize not only for a cluster assignment but also for the number of clusters while allowing for varying cluster sizes. In this setup, we provide a method to create a long-term point trajectory graph with attractive and repulsive binary terms and outperform state-of-the-art methods based on spectral clustering on the FBMS-59 dataset and on the motion subtask of the VSB100 dataset.

1. Introduction

Point trajectories are an informative intermediate representation for the motion of object parts or whole objects. They have been the key component in quite a number of works on video segmentation [7, 15, 25, 16, 29, 30, 41]. Most of these methods rely on the spectral clustering paradigm, where eigenvectors of the normalized graph Laplacian are used to generate segmentations that approximate the optimal normalized cut [35]. The underlying assumption is that objects in the scene have a certain size such that, for example, single trajectories do not end up being segmented as objects. However, a closer look at ground truth segmentations (compare Figure 1) shows that objects vary strongly in size and very small segments can be very reasonable. The minimum cost multicut [10, 12] framework allows to formulate the trajectory segmentation as an unbalanced grouping problem - allowing for very small but reasonable segments to be retrieved. If the problem is cast correctly, even the number of segments can be inferred, making the additional model selection step obsolete, that is usually needed for spectral clustering based methods.

Besides the balancing effect, spectral clustering methods have one further limitation: they are well defined only for non-negative affinities between trajectories. In practice, this means that it is easy to specify which trajectories should certainly end up in the same segment but impossible to impose that they should be separated. In the example given in Figure 1, the motion of the car (dark red label in our segmentation) differs clearly from the rest in all frames such that it can be segmented very robustly. However, the van next to it does not move for the first 30 frames and some trajectories within it already end before the motion starts. These trajectories have strong affinities to the background, hampering the van to be segmented correctly.

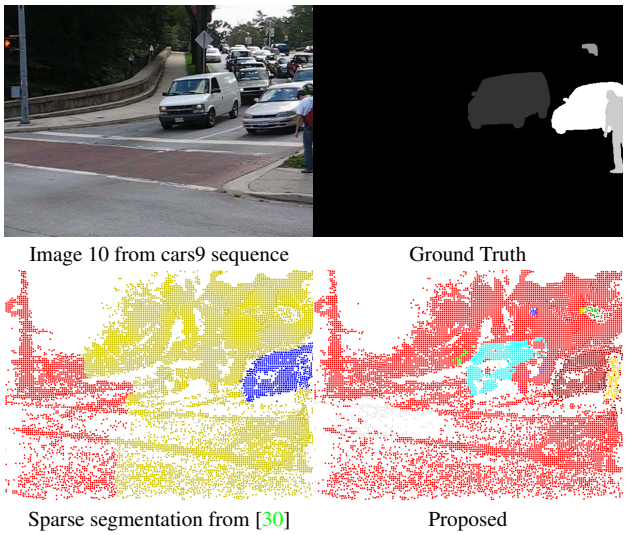


Figure 1. The objects visible in the FBMS-59 sequences are very unbalanced in size. Our method based on minimum cost multicuts without any balancing criterion outperforms the state-of-the-art spectral clustering based method [30].

In this paper, we introduce a formulation of the motion trajectory segmentation as minimum cost multicut problem on a trajectory graph. In this graph, positive and negative edge weights are computed from motion, position and color cues. Positive and negative weights allow to optimize for the correct number of moving objects including small objects that show discriminative motion in a small number

of frames. An example is the pedestrian as well as the small car (green label in our segmentation in Figure 1) in the back of the scene in Figure 1. To substantiate this intuition, we show that our model consistently outperforms spectral clustering based motion segmentation [30] on the two largest public benchmarks FBMS-59 [30] and VSB100 (motion subtask) [18, 36].

1.1. Related Work

Motion segmentation has been cast as the problem of clustering point trajectories from an image sequence with respect to their motion in for example [7, 25, 29, 26, 34, 30, 32, 19]. In [9, 13, 7, 30, 26], the grouping is based on pairwise affinities, while in [29] third order terms are employed to explain not only translational motion but also in-plane rotation and scaling. [41] model even more general 3D motions using group invariants and [14] model higher order motion subspaces. The actual grouping in these methods is done using spectral clustering with the exception of [32] who employ multi-label graph cuts and [19] who optimize an unbalanced energy that models the motion segmentation at the same time as the point matching and solve it via the Alternating Direction Method of Multiplier, *i.e*. they do not rely on any previous method to define point trajectories.

In [15], motion trajectory grouping in a setup similar to [7] is used to perform tracking. Although the grouping in [15] is computed using spectral clustering, repulsive weights computed from segmentation topology are used in the affinity matrix, based on the findings of [40]. In our approach, we compute both, attractive and repulsive weights, from motion cues.

In [16], motion trajectories are combined with framewise detection and applied to object tracking. Again, the segmentations are optimized with respect to the normalized cut objective.

For the evaluation on the VSB100 benchmark [18, 36], video segmentation methods that segment densely with respect to appearance and motion are necessary. The current state-of-the art is again defined by methods based on spectral clustering [17, 23, 24]. Concerning related work on minimum cost multicut formulations, most prior applications in computer vision were focussing on image segmentation on superpixel graphs [1, 2, 20] and pixel graphs [22]. Recently, minimum cost multicuts have also been used in a pedestrian tracking scenario [37] as an alternative to network flow approaches [31, 38]. While both minimize a cost function, the multicut formulation in [37] does not solve the tracking problem in a disjoint path manner as [31, 38] but clusters and classifies detections within each frame and over time based on binary and strong unary terms. Unlike in the multi-object tracking scenario [31, 38, 37], we assume that we are directly given long point trajectories on objects and background. The task is to cluster these trajectories into spatially and temporally consistent objects and background segments according to their long term motion behavior which provides us binary terms only. We are the first to formulate this point trajectory segmentation as a minimum most multicut problem.

2. Motion Trajectories

The basic idea behind the work presented in [7, 29, 30] is to base segmentations on the long-term motion of tracked points. This long-term motion is described by a spatio-temporal curve called trajectory. In [30], a method that generates such point trajectories for a given point sampling rate is presented. Based on large displacement optical flow [8], all points showing some underlying structure are tracked until they are occluded. The occlusion reasoning is done based on the comparison of forward and backward optical flow. When trajectories get lost due to occlusions such that the sampling is sparser than the desired rate, new trajectories are started. The result is a set of reliable trajectories that start in some frame of the sequence and end in another. Depending on the data, many trajectories do not have any frames in common. However, the longer they are, the more valuable motion information they are expected to carry. Instead of the large displacement optical flow [8], more recent optical flow approaches (*e.g*. [39, 33]) could be used instead. However, we stick to [8] for the sake of comparability to [30].

3. The Minimum Cost Multicut Problem

The minimum cost multicut problem [10, 12] is the problem of decomposing a graph $G = (V, E)$ into an optimal number of segments such that the overall cost in terms of edge weights c_e is minimized. This node labeling problem can equivalently be formulated as a binary *edge* labeling problem

$$\begin{aligned} \min_{y \in \{0,1\}^E} \quad & \sum_{e \in E} c_e y_e \\ \text{subject to} \quad & y \in \text{MC}, \end{aligned} \tag{1}$$

where MC is the set of the characteristic functions of all multicuts of G, *i.e*. all $y \in \{0,1\}^E$ that form closed boundaries and thus represent valid decompositions of the graph. Formally, these characteristic functions are described by the following cycle inequalities [10]

$$\forall C \in \text{cycles}(G), \forall e \in C: \quad y_e \leq \sum_{e' \in C \setminus \{e\}} y_{e'} \ . \tag{2}$$

In [10], it was shown that it is sufficient to consider all *chordless* cycles, *i.e*. all cycles in which each node is only connected to its successor and predecessor.
To avoid trivial solutions, the edge weights are typically

chosen to be negative for edges that should be cut and positive for those connecting nodes that should be joined. Given the cut probabilities p_e of an edge $e \in E$, the negative of the *logit* function $\text{logit}(p_e) = \log \frac{p_e}{1-p_e}$ provides such behavior. If $p_e < 0.5$, $-\text{logit}(p_e) > 0$, i.e. the according edge cost c_e is positive and e is expensive to cut. If $p_e > 0.5$, $-\text{logit}(p_e) < 0$ and it is beneficial to cut e. The inverse of the logit function is the logistic function

$$f(z) = \frac{1}{1 + \exp^{-z}}, \tag{3}$$

which can be used to compute pseudo cut probabilities from distances d with $z = \beta_0 + \beta_1 d$. The prior probability of cutting two trajectories is defined by the intercept value β_0. Assuming we learned β_0 for a certain prior cut probability p and are given a new prior cut probability $\bar{p}$, we can adapt z by

$$z = \beta_0 - \log\left(\frac{p}{1-p}\right) + \log\left(\frac{\bar{p}}{1-\bar{p}}\right) + \beta_1 d. \tag{4}$$

If the cut prior is increased, more edges will be cut and the number of resulting segments increases. Small cut priors result in an undersegmentation.

3.1. Casting Motion Trajectory Segmentation as Minimum Cost Multicut Problem

To formulate the trajectory grouping as a minimum cost multicut (MC) problem, we build a graph G such that every point trajectory is represented by a vertex $v \in V$ (compare Figure 2). If every vertex is connected by an edge $e \in E$ to its nearest neighbors, all solutions to the MC problem yield a segmentation into connected components. However, this setup makes it impossible to disambiguate partial occlusions and dis-occlusions, *e.g.* if an object passes behind the trunk of a tree such that some points on the object get occluded while others reappear on the other side of the trunk. If we insert also edges between trajectories that are further apart, we can disambiguate such cases by exploiting the similar motion behaviors of both visible components of the object. However, distinct objects that are moving coherently (like cars driving one after another on the street with the same speed) might end up in the same segment. Still, we follow [30] and chose the second option *i.e.* we insert long-range edges - hoping for the second scenario to be the less prominent one in the dataset. The weights of the edges $e \in E$ define how similar or dissimilar two trajectories are. Details about our pairwise terms are given in section 4. Now, the minimum cost multicut computed on G yields our desired segmentation into the optimal number of segments. It can be represented as a vertex labeling or as a binary edge labeling (compare Figure 2).

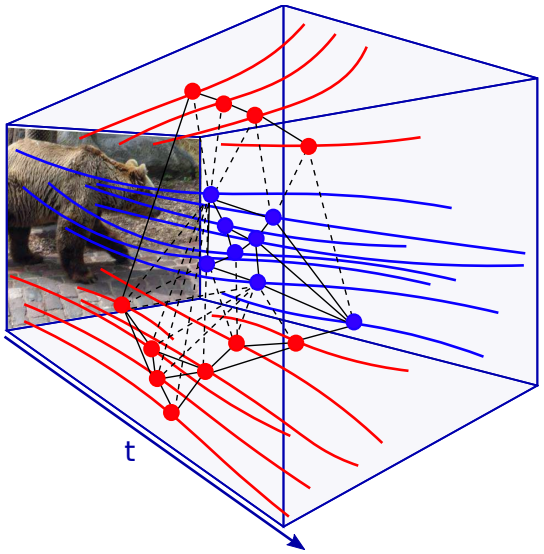


Figure 2. Long-term point trajectories are represented by nodes in the graph $G = (V, E)$. They are connected with edges $e \in E$ to trajectories with some temporal overlap but can also be connected over time. Provided that not all trajectories end in the same frame before the end of the video sequence, all nodes in V are connected. A segmentation can either be represented as a node labeling (displayed in colors) or a consistent edge labeling (solid line $\cong 0$, dashed line $\cong 1$).

3.2. Optimization

Although problem defined in equation (1) is NP-hard [5] and even APX-hard [11], instances have been solved within tight bounds or even to optimality in for example [2] using the constrained integer linear programming (ILP) problem formulation from equation 1 and branch-and-cut. While this is reasonably fast for some problem instances, it can take arbitrarily long for others. However, there has been some work on finding good heuristic solutions. In [6] for example, the move-making paradigm is employed. Kernighan and Lin [4] provide a simple primal $O\left(n^2 \log(n)\right)$ heuristic that also lead to very reasonable results in similar scenarios [21]. In our experiments, we used our own implementation of the Kernighan Lin heuristic.

The basic concept of Kernighan and Lin's method is to iteratively update a given segmentation such as to locally optimize the gain in energy. For every pair of segments a sorted list of greedily optimal label switches is generated. Those first k switches that yield the best gain in energy are made. The same is done for every given segment and the empty set, to optimize the number of segments, and repeated until convergence.

While this method has the reputation of being slow [6], we found our implementation to converge to very good results in a few seconds on graphs of up to ~ 6000 vertices.

4. Pairwise Potentials

Overall, we compute pairwise affinities for two different kinds of edges: Edges that link trajectories which have at least two frames in common and edges between trajectories without any temporal overlap. For the first kind, similarity in motion and underlying color as well as the spatial distance play a role. Similarly to [30], we define the motion difference of two trajectories at time t as

$$d_t^{\text{motion}}(A,B) = \frac{\|\partial_t A - \partial_t B\|}{\sigma_t}, \tag{5}$$

where $\partial_t A$ and $\partial_t B$ are the partial derivatives of A and B with respect to the time dimension and σ_t is the variation of the optical flow (compare [30] for details). The overall motion distance of two trajectories is computed as

$$d^{\text{motion}}(A,B) = \max_t d_t^{\text{motion}}(A,B). \tag{6}$$

Additionally to the motion distance, we compute for every pair of trajectories that share common frames the average spatial distance d^{spatial} within these frames and the average color distance d^{color} within these frames using CieLab color space. While in the affinities in [30] are computed from the product of spatial and motion distances, we find that this can easily lead to an oversegmentation in the minimum cost multicut framework. Large distances between trajectories in the background lead to repulsive weights and can induce a background tessellation. To avoid this, we allow spatial and color distances only to increase join probabilities, *i.e.* we compute $z(A,B)$ by a non-linear function

$$\begin{aligned} z(A,B) \quad = \quad \max\,(\bar{\beta}_0 \quad &+ \quad \beta_1 d^{\text{motion}}(A,B) \\ &+ \quad \beta_2 d^{\text{spatial}}(A,B) \\ &+ \quad \beta_3 d^{\text{color}}(A,B), \\ \beta_0 \quad &+ \quad \beta_1 d^{\text{motion}}(A,B)\ )\,. \end{aligned} \tag{7}$$

If the sum of spatial and color distances is large, only the motion distances are considered. This way, trajectories that are far apart but move similarly can still end up in the same segment. The feature weights and intercept values can be optimized on training data. In our experiments, we set $\bar{\beta}_0 = 6$, $\beta_0 = 2$, $\beta_1 = \beta_3 = -0.02$ and $\beta_2 = -4$.

In rapidly moving objects, it might happen that the trajectories are generally very short. In these cases, we want to support links between trajectories that appear in locations where other, lost trajectories were heading. For every trajectory A, we compute the average motion vector over the last 10 frames of its lifetime (or less if the trajectory is shorter) and compute the expected distances to every trajectory B starting within the following 5 frames as

$$r^{AB} = \| \begin{pmatrix} x^A_{\text{end}} \\ y^A_{\text{end}} \end{pmatrix} + d^{\text{gap}}(A,B)\partial_t A - \begin{pmatrix} x^B_{\text{start}} \\ y^B_{\text{start}} \end{pmatrix} \|,$$

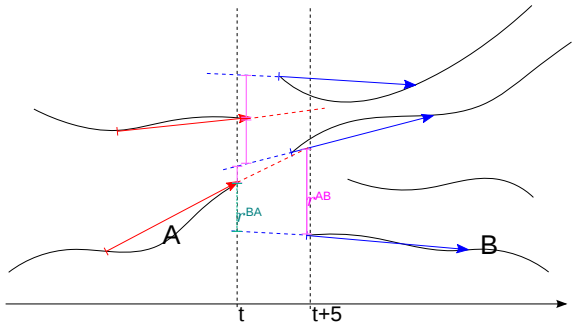


Figure 3. From every ending (red) and every beginning (blue) trajectory, we compute the hypothetic location for the following (preceding) frames. The distance of asynchronous trajectories is computed as the maximum of these hypothetic distances (magenta).

where d^{gap} is the number of frames A and B are apart, and symmetrically for the start points

$$r^{BA} = \| \begin{pmatrix} x^A_{\text{end}} \\ y^A_{\text{end}} \end{pmatrix} - d^{\text{gap}}(A,B)\partial_t B - \begin{pmatrix} x^B_{\text{start}} \\ y^B_{\text{start}} \end{pmatrix} \|.$$

Compare also Figure 3. We insert an edge between A and B if they were expected to be close to one another, had they lived in the same frame, *i.e.* if $d^{\text{async}} := \max(r^{AB}, r^{AB}) < 20$, and compute the weight from $z(A,B) = 2 - 0.1 * d^{\text{async}}$. In the logistic function (equation (3)), this evaluates to attractive weights only. In [30], such edges between asynchronous trajectories do not exist.

We assume, all resulting pseudo-probabilities are given for a cut prior $p = 0.5$ (compare equation 4). In our experiments, we modify this prior to $\bar{p}$ to generate coarser and finer segmentations.

5. Postprocessing

In [30, 7], after the spectral clustering with model selection step, the segmentations are postprocessed to reduce oversegmentation. This is done in a greedy procedure: for all neighboring segments, affine motion subspaces are computed. If they are sufficiently similar, segments are merged. We want to investigate this step, since our motion model only describes translational motion. Thus, out-of-plane motion is expected to cause oversegmentations.

6. Experiments and Results

We evaluated our method on the FBMS-59 [30] dataset as well as on the motion subtask of the VSB100 dataset [18, 36] and compare against the spatially consistent normalized cut with model selection approach from [30]. When comparing to [30], we want to disambiguate the effect of the proposed trajectory graph with richer features from the effect of modeling the decomposition by minimum cost multicuts instead of normalized cuts. We therefore ran two kinds of experiments.

Comparison of Multicuts (MC) versus Normalized Cuts To compare the performance of the minimum cost multicut problem to the well established spatially consistent normalized cut with model selection framework [30, 7], we ran a first set of experiments, denoted by MC (MultiCut), where we build the graph similarly to [30]: As in [30, 7], we only connect trajectories that share common frames. The cut probabilities are computed from motion cues only, *i.e.* as $\beta_0 + \beta_1 d^{\text{motion}}(A, B)$ with the β chosen as mentioned above. To allow for some spatial consistency, we connect in the graph G all trajectories that have an average spatial distance of less than 100 pixels. In [30], the square root of the spatial distance is used as a multiplicative factor on the motion distance in the affinity computation for this same purpose. In this experiment, we also investigate the influence of the postprocessing step on MC (MC+pp).

Multicuts on our proposed Graph (MCe) The results of the minimum cost multicut optimization computed on the enriched graph with color and spatial distance features and asynchronous connections are denoted by MCe. While we investigate the postprocessing step on MC (MC+pp), all results for MCe are given without any postprocessing.

Results on FBMS-59 The FBMS-59 [30] dataset consists of 59 video sequences split into a training set of 29 and a test set of 30 sequences. It is an extended version of the BMS-26 benchmark from Brox-Malik [7]. For this data, we show results in terms of the metrics proposed in [30] for MC with and without the postprocessing step proposed in [7] and for MCe for different trajectory densities (4 px and 8 px distance) and for the densified results using [28] computed with the public binary from [1]. The metrics evaluate precision and recall of the segmented regions after a one-to-one mapping. Thus, oversegmentation does not increase the recall (compare [30]). An additional measure of segmentation quality is the number of segmented object with an F-measure of more the 75%. Results are shown in table 1. MCe outperforms all other methods in terms of f-measure. Compared to [30], MCe yields an improvement of the f-measure of at least 7% in all sparse setups. On the test set, the improvement is slightly better than on the training set. However, on the test set with sparse trajectory sampling with 8 px distance, the number of found objects with f-measure $> 75\%$ is slightly higher if only motion cues are used. In general, the number of objects extracted with a certain per object f-measure is difficult to interpret. If an object gets oversegmented because of its articulated motion, its f-measure might drop below the threshold although its parts have been perfectly segmented. With respect to these metrics, the postprocessing from [30, 7] (MC + pp) did not further improve over MC. Since the affinities were chosen such as to not perform too much of an oversegmentation at cut prior $\bar{p} = 0.5$, this was also not to be expected. When comparing MC against MCe, the importance of the multicut formulation for the segmentation results can be assessed. The gap between [30] and MC is about 5% on the f-measure in the sparse scenarios, the further improvement by color and spatial features amounts to about 2%.

The measures from [30] are very meaningful if the segmentation is meant to find whole objects that could be used in some larger computer vision workflow. However, they do not reflect the number of correctly found object boundaries nor the trade-off between under- and oversegmentation. That is why we additionally looked at the variation of information [27]

$$VI(\text{GT}, \text{C}) = H(\text{GT}) + H(\text{C}) - 2MI(\text{GT}, \text{C}), \quad (8)$$

where $H(\text{GT})$ is the entropy of the ground truth segmentation GT, $H(\text{C})$ the entropy of the proposed segmentation C, and $MI(.,.)$ the mutual information. It can easily be seen that the VI can be split into a precision component

$$VI_{\text{precision}}(\text{GT}, \text{C}) = H(\text{C}) - MI(\text{GT}, \text{C}), \quad (9)$$

which reflects the amount of found segments that are correct and which is high when C is an oversegmentation, and a recall component

$$VI_{\text{recall}}(\text{GT}, \text{C}) = H(\text{GT}) - MI(\text{GT}, \text{C}), \quad (10)$$

which reflects how much of the GT has been recovered in C and is high for an undersegmentation. For the sake of readability, we report in this VI metric results for sparse trajectory sampling of 8px for SC [30], multicut on motion cues without and with postprocessing (MC and MC+pp) and multicut on our more descriptive features (MCe) sampled for different cut prior values (Figure 4). We applied the postprocessing only to segmentations with cut prior≥ 0.5, *i.e.* such segmentations that are expected to be oversegmentations. Thus, the green curve in Figure 4 starts with prior 0.5. The VI curves show that MCe with the richer features is consistently better than MC on the motion differences only - even if the postprocessing step with affine merging is applied there. The postprocessing indeed reduces the oversegmentation and for strong oversegmentations, the improvement is measurable. However, it improves the segmentations only little in the interesting VI range. The formulation in the minimum cost multicut framework in every of those setups outperforms SC with model selection [30]. The example segmentations of the lion01, dogs01 and people05 sequence in Figure 5 show some properties of our results. The motion boundaries are retrieved nicely and even though the lion01 sequence shows strong articulated motion, the lion is not heavily oversegmented with cut prior 0.5. The

[1] http://lmb.informatik.uni-freiburg.de/resources/binaries/.

	Training set (29 sequences)					**Test set** (30 sequences)				
	D	P	R	F	O	D	P	R	F	O
SC[30] sparse(8)	0.87%	85.10%	62.40%	72.0%	17/65	0.92%	79.61%	60.91%	69.02%	24/69
MC sparse(8), prior 0.5	0.87%	84.94%	71.22%	77.48%	23/65	0.92%	82.87%	69.89%	75.83%	**27/69**
MC sparse(8) + pp , prior 0.5	0.87%	83.62%	72.12%	77.44%	24/65	0.92%	80.60%	**70.38%**	75.15%	25/69
MCe sparse(8), prior 0.5	0.81%	86.73%	**73.08%**	**79.32%**	**31/65**	0.87%	**87.88%**	67.7%	76.48%	25/69
MCe sparse(8), prior 0.6	0.81%	**86.91%**	71.33%	78.35%	25/65	0.86%	87.57%	70.19%	**77.92%**	25/69
SC[30] sparse(4)	3.71%	82.33%	64.26%	72.27%	17/65	3.95%	76.15%	61.11%	67.81%	22/69
MC sparse(4), prior 0.5	3.75%	82.98%	70.15%	76.03%	24/65	3.98%	81.04%	**68.67%**	74.34%	**25/69**
MCe sparse(4), prior 0.5	3.47%	**86.79%**	**73.36%**	**79.51%**	**28/65**	3.72%	**86.81%**	67.96%	**76.24%**	**25/69**
SC[30] dense (4)	100%	81.50%	63.23%	71.21%	16/65	100%	74.91%	60.14%	66.72%	20/69
MC dense(4), prior 0.5	100%	82.92%	68.19%	74.84%	**24/65**	100%	81.14%	**66.58%**	73.14%	22/69
MCe dense(4), prior 0.5	100%	**85.31%**	**68.70%**	**76.11%**	**24/65**	100%	**85.95%**	65.07%	**74.07%**	**23/69**

Table 1. Results on the FBMS-59 dataset on training (left) and test set (right). We report results for **D**: average region density, **P**: average precision, **R**: average recall, **F**: F-measure and **O**: extracted objects with F $\geq$ 75%. Trajectory sampling rates for sparse and dense segmentations are given in the parentheses. We compare minimum cost multicut computed on motion cues only (**MC**) to our multicut on the enriched graph (**MCe**) to the spectral clustering based method from [30] (**SC**).

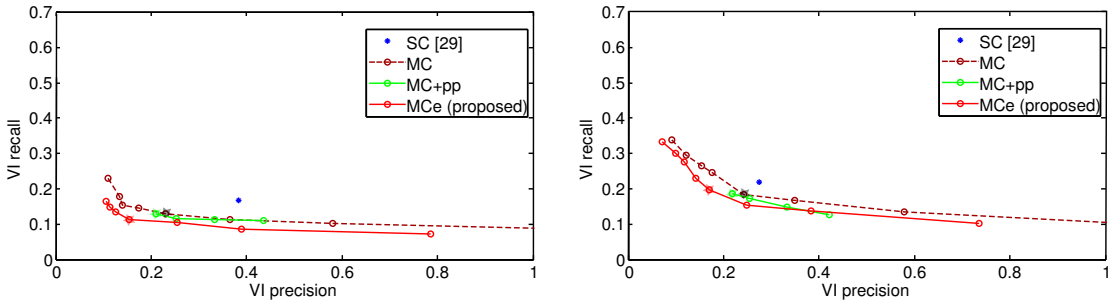


Figure 4. Results on the FBMS-59 dataset on training (left) and test set (right) in terms of variation of information (VI) split into its precision and recall component. The VI is a region metric and yields small numbers if two decompositions are similar (*i.e.* smaller is better). We report results for different prior cut probabilities to sample curves ($\bar{p} = 0.1, 0.2, \ldots, 0.8$). The values for prior 0.5 are marked with a star. The dashed, dark red curve shows results for MC from motion cues only. The values only improve little when the postprocessing step from [30] is applied (MC+pp, green curve). Our results for minimum cost multicut with enriched features (MCe, red curve) are best.

dog in the dogs01 sequence is correctly tracked despite the occlusion and disocclusion. The people05 sequence shows strong camera motion. Therefore, objects at different levels of depth tend to be oversegmented but also the motion of the lawn is not interpreted correctly. However, the actually moving objects can still be segmented.

We also measured the computation times for MCe and the SC approach of [30] (see Table 2). For both methods, the computation times vary strongly between the sequences, depending on the size of the resulting graphs. However, the average runtimes are comparable. For MCe with sparse trajectories of 8px distance, the average number of vertices is 8,569 on the training and 9,183 on the test set, for 4px trajectory sampling rate it is 37,629 vertices on the training and 40,789 vertices on the test set on average.

	Training set		**Test set**	
	mean	std dev	mean	std dev
SC[30] sparse(4)	667	968	820	1.31e+03
SC[30] sparse(8)	176	149	195	166.5
MCe sparse(4)	631	1.01e+03	780	1.42e+03
MCe sparse(8)	143	148	169	221

Table 2. Computation times of the full motion trajectory segmentation in seconds measured on an Intel Xeon CPU with 2.70GHz. The runtimes vary strongly with the number of trajectories.

Results on VSB100 Additionally, we evaluated our approach on the motion subtask of the recently proposed VSB100 benchmark [18, 36]. The dataset consists of 40 train and 60 test sequences with a maximum length of 121 frames. For every 20th frame, several human annotations are given. For the motion subtasks of VSB100, non-moving objects are ignored in the evaluation. In Figure 6, we compare our results to those achieved by [30] and to the segmentation propagation which was introduced as a baseline in [18] as well as to the recently proposed state-of-the-art for video segmentation [24]. For the segmentation propagation, segments are generated by applying a good image segmentation method [3] to the central frame and propagating it using optical flow. While this approach is straightforward, it still shows competitive results on VSB100. Our

Figure 5. Frames $1, 20, 40, \ldots, 100$ of the lion01 sequence, frames $80, 100, \ldots, 180$ of the dogs01 sequence and frames $120, 140, \ldots, 220$ of the people05 sequence of FBMS-59 and the results of our MCe motion trajectory segmentation. Articulated motion can lead to a slight overegmentation as in the lion01 example (top). In the dogs01 sequence (center), the dog is correctly tracked despite the occlusion. The people05 sequence (bottom row) shows the effect of a moving camera. While the moving objects in the scene (adult biker, child on the balance bike) are segmented, some static objects are segmented as well, as for example the blackboard.

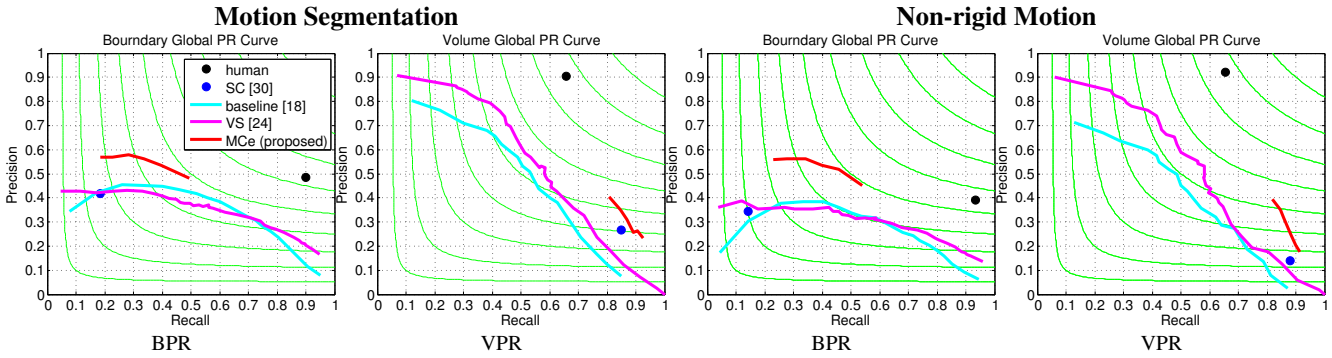
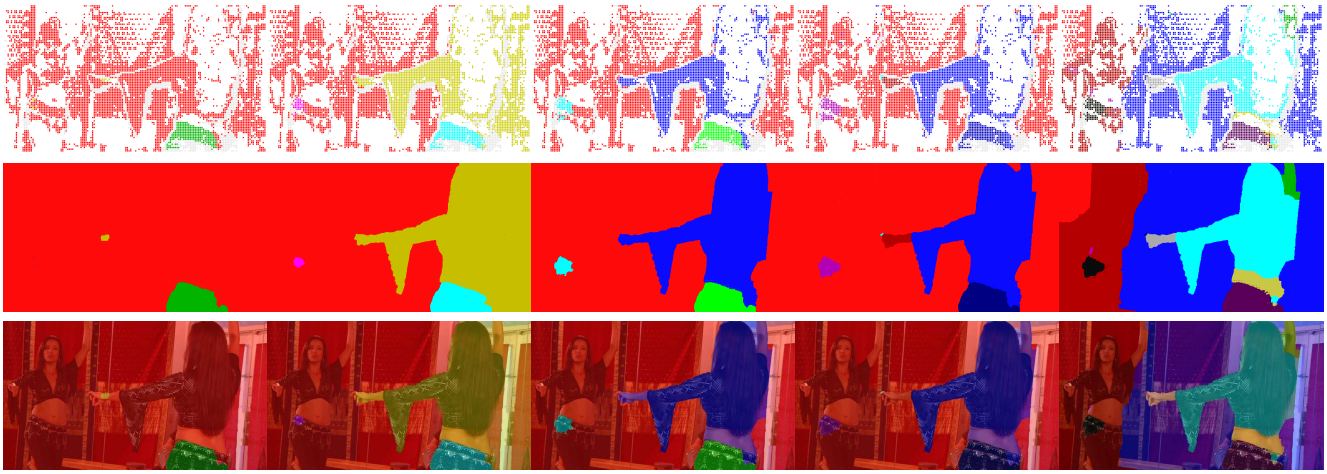


Figure 6. Results on the motion subtask of VSB100 [18]. We consistently outperform [30] and the baseline for our sampled cut priors.

results were computed for cut priors $0.2, 0.3, \ldots, 0.7$ with trajectories with sparsity 4 and densified using [28].

The curves show that our segmentations are rather in the low recall range of the boundary precision recall (BPR) with a significantly higher precision than all competing methods. However, we also outperform the current state-of-the-art at equal error rate. Similarly, in the volume precision recall curve (VPR), our segmentations are in the high recall range. This means that we are missing objects in the scene, but at the given level of granularity, our segmentations are more consistent than those from competing methods. An example for our segmentation results and the densification for different cut priors is given in Figure 7. The prominent motion in this sequence is the dancer's hip motion, but there is



$\bar{p} = 0.3$ $\bar{p} = 0.4$ $\bar{p} = 0.5$ $\bar{p} = 0.6$ $\bar{p} = 0.7$

Figure 7. Our results on frame 1 of the belly dancing sequence from VSB100 for different cut priors $\bar{p}$. From our trajectory segmentation (top row), we compute dense segmentations using [28] (center row). The overlay with the original image is shown in the bottom row. The most prominently moving part, dancer's hip, is segmented with the lowest cut prior.

	P	R	F	O	VI
Training set (7 sequences)					
MCe+ILP	94.161%	80.673%	86.897%	10/15	0.0798
MCe	94.173%	80.534%	86.821%	10/15	0.0794
Test set (6 sequences)					
MCe+ILP	93.292%	82.222%	87.41%	8/13	0.0680
MCe	93.294%	82.227%	87.44%	8/13	0.0680

Table 3. Results on a subset the FBMS-59 dataset on training (top) and test set (bottom) for MCe solve with the Kernighan Lin Algorithm [4] (proposed) versus MCe with the ILP solver from [2]. Again, we report results for **P**: average precision, **R**: average recall, **F**: F-measure, **O**: extracted objects with F $\geq 75\%$, and **VI**: Variation of information (smaller is better). All results are given for cut prior 0.5 and 8px trajectory sampling.

also a slight backward motion. As the cut prior increases, more and more details of the dancer are segmented. The reflection in the mirror is only segmented with the largest cut prior $\bar{p} = 0.7$.

6.1. Optimality

We also tried to optimize our MCe problems with sparsity 8 and cut prior 0.5 with the optimal ILP solver from [2][2] using the Cplex optimizer. However, the search for the optimal solution did finish for none of the FBMS-59 sequences within 10 hours. By setting an allowed absolute optimality gap of 0.05, some of the cars sequences ($\sim$ 2000 - 6000 trajectories per sequence) finished within four hours, but none of the other sequences did so within 2 days. With optimality gap of 0.5, the MCe problems for the sequences cars2, cars3, cars6, cars7, marple11, cats04, and cats07 from the training set and the sequences cars1, cars5, people1, people2, goats01, and rabbits04 from the test set were computed within one day.

On this subset, we evaluate the segmentations with the precision and recall metrics from [30] and additionally report the variation of information (VI) (see Table 3). The absolute numbers are much better than for the whole FBMS-59 benchmark (compare Table 1). This tells that the solved sequences were rather easy examples. The difference between the purely heuristic solution and the the ILP solver, that provides a solution within bounds of the optimal solution, is very small on these sequences.

7. Conclusion

We proposed a formulation of the motion trajectory segmentation problem in terms of minimum cost multicuts. In this formulation, costs are defined by positive and negative edge weights, that are computed between synchronous as well as between asynchronous trajectories. We showed that the solutions to this minimum cost multicut problem, found as fixed points of the Kernighan and Lin heuristic [4], consistently outperform the spectral clustering based state-of-the-art in motion segmentation on FBMS-59 as well as on the motion subtask of VSB100.

Acknowledgements

M.K. and T.B. acknowledge funding by the ERC Starting Grant VideoLearn.

References

[1] B. Andres, J. H. Kappes, T. Beier, U. Köthe, and F. A. Hamprecht. Probabilistic image segmentation with closedness

[2] https://github.com/bjoern-andres/graph.

constraints. In *ICCV*, 2011. 2

[2] B. Andres, T. Kröger, K. L. Briggman, W. Denk, N. Korogod, G. Knott, U. Köthe, and F. A. Hamprecht. Globally optimal closed-surface segmentation for connectomics. In *ECCV*, 2012. 2, 3, 8

[3] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE TPAMI*, 33(5), 2011. 6

[4] S. L. B. W. Kernighan. An efficient heuristic procedure for partitioning graphs. *Bell Systems Technical Journal*, 49:291–307, 1970. 3, 8

[5] N. Bansal, A. Blum, and S. Chawla. Correlation clustering. *Machine Learning*, 56(1–3):89–113, 2004. 3

[6] T. Beier, T. Kroeger, J. Kappes, U. Kothe, and F. Hamprecht. Cut, glue, & cut: A fast, approximate solver for multicut partitioning. In *CVPR*, 2014. 3

[7] T. Brox and J. Malik. Object segmentation by long term analysis of point trajectories. In *ECCV*, 2010. 1, 2, 4, 5

[8] T. Brox and J. Malik. Large displacement optical flow: descriptor matching in variational motion estimation. *IEEE TPAMI*, 33(3):500–513, 2011. 2

[9] A. Cheriyadat and R. Radke. Non-negative matrix factorization of partial track data for motion segmentation. In *ICCV*, 2009. 2

[10] S. Chopra and M. Rao. The partition problem. *Mathematical Programming*, 59(1–3):87–115, 1993. 1, 2

[11] E. D. Demaine, D. Emanuel, A. Fiat, and N. Immorlica. Correlation clustering in general weighted graphs. *Theoretical Computer Science*, 361(2–3):172–187, 2006. 3

[12] M. M. Deza and M. Laurent. *Geometry of Cuts and Metrics*. Springer, 1997. 1, 2

[13] R. Dragon and J. Rosenhahn, B.and Ostermann. Multi-scale clustering of frame-to-frame correspondences for motion segmentation. In *ECCV*, 2012. 2

[14] E. Elhamifar and R. Vidal. Sparse subspace clustering. In *ICCV*, 2013. 2

[15] K. Fragkiadaki and J. Shi. Detection free tracking: Exploiting motion and topology for segmenting and tracking under entanglement. In *CVPR*, 2011. 1, 2

[16] K. Fragkiadaki, W. Zhang, G. Zhang, and J. Shi. Two-granularity tracking: Mediating trajectory and detection graphs for tracking under occlusions. In *ECCV*, 2012. 1, 2

[17] F. Galasso, M. Keuper, T. Brox, and B. Schiele. Spectral graph reduction for efficient image and streaming video segmentation. In *CVPR*, 2014. 2

[18] F. Galasso, N. Nagaraja, T. Cardenas, T. Brox, and B.Schiele. A unified video segmentation benchmark: Annotation, metrics and analysis. In *ICCV*, 2013. 2, 4, 6, 7

[19] P. Ji, H. Li, M. Salzmann, and Y. Dai. Robust motion segmentation with unknown correspondences. In *ECCV*. 2014. 2

[20] J. H. Kappes, M. Speth, B. Andres, G. Reinelt, and C. Schnörr. Globally optimal image partitioning by multicuts. In *EMMCVPR*, 2011. 2

[21] J. H. Kappes, M. Speth, G. Reinelt, and C. Schnörr. Higher-order segmentation via multicuts. *CoRR*, abs/1305.6387, 2013. 3

[22] M. Keuper, E. Levinkov, N. Bonneel, G. Lavoue, T. Brox, and B. Andres. Efficient decomposition of image and mesh graphs by lifted multicuts. In *ICCV*, 2015. 2

[23] A. Khoreva, F. Galasso, M. Hein, and B. Schiele. Learning must-link constraints for video segmentation based on spectral clustering. In *GCPR*, 2014. 2

[24] A. Khoreva, F. Galasso, M. Hein, and B. Schiele. Classifier based graph construction for video segmentation. In *CVPR*, 2015. 2, 6

[25] J. Lezama, K. Alahari, J. Sivic, and I. Laptev. Track to the future: Spatio-temporal video segmentation with long-range motion cues. In *CVPR*, 2011. 1, 2

[26] Z. Li, J. Guo, L. Cheong, and S. Zhou. Perspective motion segmentation via collaborative clustering. In *ICCV*, 2013. 2

[27] M. Meilă. Comparing clusterings by the variation of information. In *Learning Theory and Kernel Machines*, pages 173–187, 2003. 5

[28] P. Ochs and T. Brox. Object segmentation in video: a hierarchical variational approach for turning point trajectories into dense regions. In *ICCV*, 2011. 5, 7, 8

[29] P. Ochs and T. Brox. Higher order motion models and spectral clustering. In *CVPR*, 2012. 1, 2

[30] P. Ochs, J. Malik, and T. Brox. Segmentation of moving objects by long term video analysis. *IEEE TPAMI*, 36(6):1187 – 1200, Jun 2014. 1, 2, 3, 4, 5, 6, 7, 8

[31] H. Pirsiavash, D. Ramanan, and C. C. Fowlkes. Globally-optimal greedy algorithms for tracking a variable number of objects. In *CVPR*, 2011. 2

[32] H. Rahmati, R. Dragon, O. M. Aamo, L. V. Gool, and L. Adde. Motion segmentation with weak labeling priors. In *GCPR*, 2014. 2

[33] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid. EpicFlow: Edge-Preserving Interpolation of Correspondences for Optical Flow. In *CVPR*, 2015. 2

[34] F. Shi, Z. Zhou, J. Xiao, and W. Wu. Robust trajectory clustering for motion segmentation. In *ICCV*, 2013. 2

[35] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE TPAMI*, 22(8):888–905, Aug. 2000. 1

[36] P. Sundberg, T. Brox, M. Maire, P. Arbelaez, and J. Malik. Occlusion boundary detection and figure/ground assignment from optical flow. In *CVPR*, 2011. 2, 4, 6

[37] S. Tang, B. Andres, M. Andriluka, and B. Schiele. Subgraph decomposition for multi-object tracking. In *CVPR*, June 2015. 2

[38] X. Wang, E. Turetken, F. Fleuret, and P. Fua. Tracking interacting objects optimally using integer programming. In *ECCV*, 2014. 2

[39] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid. DeepFlow: Large displacement optical flow with deep matching. In *ICCV*, 2013. 2

[40] S. X. Yu and J. Shi. Understanding popout through repulsion. In *CVPR*, 2001. 2

[41] V. Zografos, R. Lenz, E. Ringaby, M. Felsberg, and K. Nordberg. Fast segmentation of sparse 3d point trajectories using group theoretical invariants. In *ACCV*, 2014. 1, 2