Higher-Order Multicuts for Geometric Model Fitting and Motion Segmentation

Evgeny Levinkov*, Amirhossein Kardoost*, Bjoern Andres, and Margret Keuper

Abstract—The minimum cost lifted multicut problem is a generalization of the multicut problem (also known as correlation clustering) and is a means to optimizing a decomposition of a graph w.r.t. both positive and negative edge costs. It has been shown to be useful in a large variety of applications in computer vision thanks to the fact that multicut-based formulations do not require the number of components given a priori; instead, it is deduced from the solution. However, the standard multicut cost function is limited to pairwise relationships between nodes, while several important applications either require or can benefit from a higher-order cost function, i.e. hyper-edges. In this paper, we propose a pseudo-boolean formulation for a multiple model fitting problem. It is based on a formulation of any-order minimum cost lifted multicuts, which allows to partition an undirected graph with pairwise connectivity such as to minimize costs defined over any set of hyper-edges. As the proposed formulation is NP-hard and the branch-and-bound algorithm (as well as obtaining lower bounds) is too slow in practice, we propose an efficient local search algorithm for inference into resulting problems. We demonstrate versatility and effectiveness of our approach in several applications: 1) We define a geometric multiple model fitting, more specifically, a line fitting problem on all triplets of points and group points, that belong to the same line, together. 2) We formulate homography and motion estimation as a geometric model fitting problem where the task is to find groups of points that can be explained by the same geometrical transformation. 3) In motion segmentation our model allows to go from modeling translational motion to Euclidean or affine transformations, which improves the segmentation quality in terms of F-measure.

Index Terms—Combinatorial optimization, local search algorithm, higher-order multicut, motion segmentation, geometric model fitting

1 Introduction

Multicut-based formulations have recently received considerable attention and have been successfully applied to a variety of tasks in computer vision [1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12]. Their application to a new problem is particularly easy: one is only required to provide probability estimates of pairs of nodes to belong together, while no information about the exact number of clusters or their expected sizes is necessary. These parameters are defined by the solution. Multicut-based frameworks are therefore particularly interesting for tasks such as multiple model fitting or multiple object motion segmentation, where one wants to avoid additional model selection steps and prefers the correct number of objects/models to be directly inferred from each problem instance.

However, the multicut cost function [13], [14] can assign a cost or a reward only to direct neighbors in the graph, which can be a serious limitation in certain applications. For example, in case of image segmentation and a 4-connected graph the final solution is likely to deteriorate significantly, since inter-pixel edge probability estimates tend to be noisy. Keuper et al. [3] introduced additional (*lifted*) edges into the multicut objective, which allow to capture information in a non-local neighborhood, but preserve the original feasible set of solutions. Furthermore, Kim et al. [15] proposed a higher-order multicut formulation, that allows to model dependencies between more than two nodes. In this work, we combine these two ideas in one formulation. This generalization allows us to apply minimum cost multicuts to geometric model fitting as well as motion segmentation problems, which both require higher-order non-local costs and can have a variable number of objects in each problem.

On the downside, Bansal et al. [14] showed, that solving the multicut problem is exactly NP-hard. This result extends to the above mentioned multicut-based formulations. Although branch-and-bound [2] algorithms, as well as LP relaxations [15], [16], are feasible when applied to small problems, they do not easily scale [17]. Instead, we propose a local search algorithm based on an efficient move making algorithm [3]. This original heuristic by Keuper et al. proposes feasible solutions for the (second order) lifted multicut problem. Here, we extend it to handle also higher-order terms and their combinations. Such heuristics do not provide any guarantees on the quality of solutions or computation time, but work well in practice [17] and provide feasible solutions at any time. Thanks to the affordable runtime of our proposed local search algorithm, we were able to apply higher order (lifted) multicuts to large problems, which we describe in details below.

1.1 Geometric Model Fitting

The task of robust geometric model fitting is to explain observational data under a given model assumption. In this paper we tackle the most general problem setting, i.e. we assume that the number of models is unknown, there is

- *E. Levinkov and A. Kardoost contributed equally.
- E. Levinkov is with Bosch Center for Artificial Intelligence, Renningen, Germany.
- A. Kardoost is with the Data and Web Science Group, University of Mannheim, Mannheim, Germany.
- B. Andres is with TU Dresden and the Center for Systems Biology Dresden (CSBD).

He acknowledges support by the Dresden/Leipzig Center for Scalable Data Analytics and Artificial Intelligence (ScaDS.AI).

- M. Keuper is with the University of Siegen, Max Planck Institute for Informatics, Saarland Informatics Campus.

M. K. and A. K. acknowledge funding from the German Research Foundation project KE 2264/1-1.
Contact e-mail: margret.keuper@uni-siegen.de

a significant amount of background noise, and the models themselves are perturbed.

The most traditional way to solve this task is the random sampling consensus (RANSAC) [18]. It starts with a random subset of data points and iteratively adds or removes data points to grow the *inliers set*, i.e. the set of points with a model error below a certain threshold w.r.t. a single model. A straightforward extension of RANSAC to multiple model fitting is to iteratively fit a model, remove all associated inliers, and proceed with fitting another model. This can lead to undesired results since the information about the possible relation between the removed and the remaining data is lost. Zuliani et al. [19] proposed multiRANSAC to improve over RANSAC in this respect, which requires a user to specify the number of models. As other recent geometric model fitting approaches [20], [21], it is based on random sampling and thus sensitive to the initial condition.

In contrast, we cast the geometric model fitting problem as a point grouping problem as previously done e.g. in [22], [23], [24]. Our approach is entirely based on local observations over which a global and probabilistically motivated optimization is possible. In this setting, the considered problem sizes are usually small since previous approaches (e.g. [20], [21]) require large computation times, such that we can easily employ and solve models of order up to 5 and thereby show that our formulation and heuristic are principled.

1.2 Motion Segmentation

Motion segmentation, as addressed for example in [25], [26], [27], [28], [29], is a task of segmenting salient moving objects in a video. According to the Gestalt principle of common fate [30], motion patterns of objects are often more homogeneous than their appearance and provide robust cues for moving object segmentation. Thus, from accurately estimated point-wise motions, object motion models can be fit through the formulation of a point grouping problem over local motion similarities.

The Euclidean difference between two local motion descriptors such as optical flow vectors or point trajectories measures how well the behavior of the two entities can be described by a single translational motion model. A simple model with only pair-wise potentials can yield good performance in practice [4], [31]. While being successful in providing segmentations in simple scenarios, more complex motion patterns can not be resolved with only pair-wise potentials. For example scaling (e.g. zooming in/out of the camera or movement of objects toward the camera), out-of-plane rotation and highly non-rigid motion of object parts hinders providing high quality motion segmentations. Therefore, higher-order motion models, that can compare more than two motion vectors at a time, are required.

Transformations describing translation, rotation and scaling can be estimated from two motion vectors. Thus, for any three points, one can estimate how well their motion can be explained by one Euclidean transformation through residual errors. Costs that describe such motion differences are thus at least of order three. Affine motion differences can be estimated from four motion vectors, and to assign costs to differences in homographies the minimum required order is five. Our model offers the flexibility to combine edges of varying order in one problem instance - which we exploit to produce robust model fits up to Euclidean transformations. Yet, complementary to the geometric model fitting application, the relevant motion segmentation benchmarks yield rather large problem sizes such that we are limited to models up to order three in this setting. Thus, the first application we consider, geometric model fitting, indicates the principled applicability of our formulation while the second, motion segmentation, proves the practical relevance of these results.

One additional adversity in motion segmentation is to distinguish between different objects with similar underlying motion patterns. Lifted Multicuts [3] have shown to resolve such ambiguities appropriately in the context of image segmentation. We show that higher-order graphs with third order edges and their combination with lifted edges propose better motion segmentations and disambiguate complex motion patterns like similarly moving objects, scaling motions and out-of-plane rotations of the objects.

Contributions In summary, we make the following contributions: We provide a formulation for the geometric model fitting problem using higher-order lifted minimum cost multicuts. We show its applicability to multiple model fitting problems such as line fitting from total least squares estimates and prove its practical benefit for motion segmentation. In contrast to previous approaches, our model allows the segmentation of noisy data into segments w.r.t. motion models beyond in-plane translation by combining second and third order edges and allows for efficient optimization.

2 Related Work

Geometric Model Fitting Most recent and well-performing approaches to geometric model fitting are based on *preference* analysis. They start with sampling a number of model hypotheses to build a preference matrix with the inliers to these hypotheses. J-Linkage [21] builds a preference matrix by assigning either 1 or 0 indicating if a point belongs to a hypothesis, determined by a threshold. Then they perform a greedy agglomerative clustering of points using the Jaccard distance, i.e. they group points with similar preference sets. To avoid hard 01-assignments, T-Linkage [20] relaxes the membership value and uses a different similarity metric [32]. Magri and Fusiello [33] cast geometric model fitting as a maximum set coverage problem: Given an integer k, exactly k subsets of hypotheses are selected, covering the maximum number of points.

A separate line of research seeks to find low-rank representations of the preference matrix and thus discover the models. Robust Preference Analysis (RPA) [34] builds a symmetric kernel of pairwise similarities (measured with Tanimoto distance). Then it performs Robust PCA to obtain a rank-k representation and clusters points in this reduced space using Symmetric Non-negative Matrix Factorization. Avoiding all intermediate operations, Non-negative Matrix Underapproximation (RS-NMU) [35] directly finds a low-rank representation of the preference matrix and yields high-quality results. Their method gains robustness via a t-test that efficiently filters out statistically insignificant hypotheses. Denitto et al. [36] propose an approach to compute a sparse low-rank representation of the preference matrix

using FABIA [37] and obtain *bi-clusters*, i.e. clustering in rows and columns of the matrix, that allows points to belong to different geometric models simultaneously.

Isack and Boykov [38] also formulated a combinatorial optimization problem with a discrete label space of possible models parameters. They iteratively find inliers to the models by applying the α-expansion algorithm [39] and then re-estimate model parameters and prune the redundant ones. Amayo et al. [40] proposed an efficient primal-dual optimization method for a convex relaxation of the discrete energy of [38]. Recently, Barath and Matas [41] have introduced density modes into models' parameters space and showed that applying mean-shift [42] can efficiently reduce the number of models.

Higher-order clustering has previously been used in [22], [23], [24], [43], [44], [45], [46] for geometric model fitting. Higher-order potentials are defined over k-tuples of points and a probability of the latter to belong together is computed based on the residuals to the sampled hypotheses. However, to perform clustering, these approaches transform higher-order terms into pairwise potentials or use randomized techniques and apply spectral clustering. Thus, in principle, they also require the number of models to be given. In contrast, we work directly with the higher-order potentials and perform correlation clustering so as to automatically recover the optimal number of clusters. However, the results presented in [22] and [24] can be seen as a motivation for the use of local evidence on larger than minimal sets, as employed in the proposed approach.

Motion Segmentation Higher-order graph decompositions and spectral clustering approaches have been used extensively in the works of [27], [47], [48], [49], [50], [51], [52] in computer vision. The higher-order graph decomposition is proposed first by Agarwalal et al. [43]. Specifically, [27], [50], [51] model higher-order motions. Zografos et al. [50] model 3D motions using group invariants and [51] model motion subspaces. The segmentation is then generated by projecting the resulting hyper-graph onto its primal graph and solving the spectral clustering problem there. Higher-Order Markov Random Field (MRF) and Conditional Random Field (CRF) models have been addressed in [53], [54], [55] and [56].

Our approach relies on the minimum cost lifted multicut formulation for hyper-graph decomposition. Such formulation is different from both previous approaches. In contrast to spectral clustering, the multicut formulation does not suppose any balancing criterion. Moreover, we directly infer segmentations from the hyper-graph without any projection onto its primal graph. In contrast to MRFs, the proposed approach allows higher-order edges to connect vertices globally, violating the Markov property. Further, MRFs and CRFs aim at inferring a node labeling with labels given a priori while multicut approaches aim at inferring an edge labeling yielding an optimal number of segments.

We cast motion segmentation as a point trajectory grouping problem and treat it like a model fitting problem to generate segments based on different motion pattern. In a similar way, it has previously been addressed in [4], [25], [26], [27], [28], [29], [31], [57]. From sparse motion segmentations, frame-wise dense segmentations can be computed by variational approaches [58] or through learning [12].

In contrast, end-to-end trained CNN based approaches to motion segmentation are based on single frame segmentations from optical flow [59], [60], [61], [62], [63], [64]. Tokmakov et al. [59], [60], [65] use large amounts of synthetic training data [66] and learn to generate binary object masks. [59] combine motion cues with an ImageNet [67] pre-trained appearance model and a GRU for increased temporal consistency. In a similar way, Jain et al. [64] employ a realistic dataset extracted from pairs of frames from the ImageNet video dataset [67] to learn object motion and appearance cues. While these approaches directly yield pixel accurate segmentations, they replace explicit motion model assumptions by huge amounts of training data and can not inherently determine the number of moving objects. In DyStaB [68] unsupervised moving object segmentation is done by partitioning the motion field w.r.t. a mutual information based objective and learning object models from the segments. Yang et al. [69] propose a self-supervised transformer model to segment optical flow fields into primary objects and background in a generative way. The combination of appearance based detectors and geometric motion segmentation is used to segment *rigid* motions in [70].

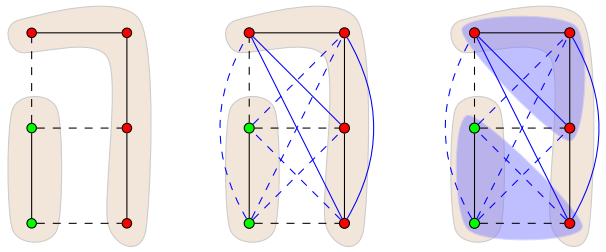
In [71], motion segmentation is approached in a probabilistic way and the camera motion is subtracted from each frame for improved training. In [72], this idea of prior camera motion subtraction is used to allow for better CNN training for frame-wise segmentation. Bideau et al. [73] propose a multi-step procedure in which first the camera motion, fit using RANSAC in the first frames, is subtracted, then a set of rigid motion models is fitted and last object segmentation proposals from CNNs are used to combine the rigid motion parts into objects. In [74], a unified approach is proposed to handle the moving object detection problem separately in the 2D and 3D scenes based on geometric interpretations and the parallax motion analysis. Our approach directly estimates rigidly moving object parts in a single step, accounting for camera motion using third-order models and does not depend on external object proposals.

Minimum Cost Multicuts Most previous works on minimum cost multicuts in computer vision focus on problems with pair-wise potentials [3], [4], [75], [76]. The exception is the model first presented by [15], [77] and extensively studied in [16]. Therein, higher-order costs are used for image segmentation on superpixel graphs with pairwise neighborhood connectivity. In [16] an implementation of a branch-and-bound algorithm is proposed for small problem instances. We propose a higher-order *lifted* multicut model, which allows the definition of higher-order edge costs for lifted edges as well as for connectivity defining edges. Lifted multicuts w.r.t. a pair-wise graphs have been proposed in [3] along with a local search algorithm to find solutions in reasonable time. A different algorithm for the problem was proposed in [78] and [79]. Here, we generalize the solver from [3] to facilitate the inference in higher-order problems.

3 Higher-Order Lifted Multicut Problem

A decomposition of a graph $G = (V, E)$ can be represented by assigning to each vertex an identifier of a component it belongs to, i.e. a *vertex labeling*. The drawback of such an encoding is that a permutation of components' identifiers will result in a different vertex labeling, while encoding

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2022.3148795, IEEE Transactions on Pattern Analysis and Machine Intelligence



(a) **(b)** **(c)**

Fig. 1. **(a)** An example of a graph decomposition and its encodings. Switching green and red labels will produce a different encoding for the same decomposition. Dashed lines, in turn, constitute a multicut of a graph and uniquely define its decomposition. **(b)** An example of a *lifted* graph decomposition and it encoding. Blue lines denote lifted edges, that connect vertices which are not direct neighbors in the graph. **(c)** An example of 3rd-order costs, that consider three nodes at a time (light blue triangles) for a better join / cut decision. If a higher-order cost does not correspond to a clique in the graph, we add lifted edges.

the same decomposition. This ambiguity creates problems during optimization that are hard to deal with, because the search space of feasible solutions can be factorially large. An alternative approach is to assign either 0 or 1 to each edge such that edges labeled 1 connect nodes only inside connected components (Fig. 1 (a)).

Such 01-edge labeling, complying with constraints we define below, is called a *multicut* of a graph. Chopra and Rao [80] define a binary linear program called *minimum cost multicut problem*, that allows to optimize for the 01-edge labeling, or in other words to find an optimal decomposition of a graph. Its main advantage is that no prior information is necessary about the number of clusters in the data, instead, it is deduced from the solution. This is exactly the setting in the applications we consider in this paper, as we do not know beforehand how many geometric models or moving objects the data contains.

However, there are two main limitations of the multicut problem: 1) It allows to specify costs or rewards only for direct neighbors in the graph, which limits the expressiveness of the cost functions only to local neighborhoods. Keuper et al. [3] introduced *lifted* edges (Fig. 1 (b)) into the 2nd-order multicut problem and showed that it greatly improved image and mesh segmentation results. An important difference of the lifted edges is that they only define a cost between vertices, but not connectivity, thus preserving the original feasible set of solutions. See [81] for a detailed proof, that the lifted multicut problem is not simply equivalent to a multicut problem with more edges. 2) It allows to specify only pairwise edge costs, which is not enough for some applications. For example, for line fitting and motion segmentation we need 3rd-order costs, while homography estimation requires 5th-order costs. Kim et al. [15] and Kappes et al. [16] proposed a formulation that allows to specify costs of arbitrary order (Fig. 1 (c)). They used a cutting-plane algorithm to solve emerging models, which turns out to be impractical for large real-world instances due to the LP-solver's bottleneck. Below, we combine the two above-mentioned findings in a joint formulation.

Definition 1 For a simple, connected graph $G = (V, E)$ and lifted edges F, such that $F \subseteq \binom{V}{2} \setminus E$, let $\mathcal{V} = \{U \mid U \in 2^V, |U| \geq 2\}$ denote the set of connected subsets of nodes in G. For a given cost function $c\colon \mathcal{V} \to \mathbb{R}$, written below is an instance of the higher-order minimum cost lifted multicut problem

$$\min_{y \in \{0,1\}^{E \cup F}} \sum_{U \in \mathcal{V}} c_U \prod_{\{v,w\} \in \binom{U}{2} \cap \{E \cup F\}} y_{vw}\,, \tag{1}$$

with y subject to the following linear constraints

$$\forall C \in \text{cycles}(G), \forall e \in C\colon \quad (1 - y_e) \leq \sum_{e' \in C \setminus \{e\}} (1 - y_{e'}) \tag{2}$$

$$\forall f = \{v, w\} \in F, \forall P \in vw\text{-paths}(G)\colon \quad (1 - y_f) \leq \sum_{e \in P} (1 - y_e) \tag{3}$$

$$\forall f = \{v, w\} \in F, \forall T \in vw\text{-cuts}(G)\colon \quad y_f \leq \sum_{e \in T} y_e\,. \tag{4}$$

Cycle inequalities (2) ensure that the cut in graph G does not have holes. Path inequalities (3) guarantee that $\forall f = \{v, w\} \in F$, y_f can be assigned value 1, iff there exists a path P in graph G, that connects vertices v and w. Otherwise, the solver then has two options: either create such path or set $y_f = 0$. Cut inequalities (3) guarantee, that $\forall f = \{v, w\} \in F$, y_f can be assigned value 0, iff there exists a cut T in graph G, that separates vertices v and w. Otherwise, the solver has to either create such a cut or set $y_f = 1$.

Note that in our formulation (1), the set of decompositions is defined over a pairwise connected graph G. The costs, however, are defined over connected subsets $U \in 2^V$ of nodes of arbitrary cardinality larger than 1. Normally, only subsets of fixed cardinality k are used, e.g. $\mathcal{V} = \binom{V}{k}$. However, one can define a cost function over several cardinalities $K \subset \mathbb{N} \setminus \{1\}$. It is easy to see, that in case $K = \{2\}$ we get the lifted multicut formulation from [3]. Therefore, here we propose a strictly more general formulation.

Keuper et al. [3] showed the connection of optimization problem (1) with $\mathcal{V} = \binom{V}{2}$ to finding the most likely multicut in Bayesian sense: Let $p(y_{vw} = 1 \mid x_{vw})$ be a conditional probability estimate for two nodes $\{v, w\} \in \mathcal{V}$ to belong together given some features x_{vw}. If we set costs as $c_{vw} = \log \frac{1 - p(y_{vw}=1|x_{vw})}{p(y_{vw}=1|x_{vw})}$, then minimizing (1) is the same as performing MAP inference in the induced Bayesian network. The extension from 2nd-order sets to higher-order cases is straightforward.

This allows to interpret solutions of (1) in terms of local probabilities. If $p(\prod_{\{v,w\} \in \binom{U}{2} \cap \{E \cup F\}} y_{vw} = 1 \mid x_U)$ is greater than 0.5 for some $U \in \mathcal{V}$, then the corresponding cost c_U is less than 0. This means that all nodes in U are likely to belong together. We call such terms *attractive*. Conversely, if $p(\prod_{\{v,w\} \in \binom{U}{2} \cap \{E \cup F\}} y_{vw} = 1 \mid x_U)$ is less than 0.5, then the corresponding cost c_U is greater than 0 and acts as a penalty. We call such terms *repulsive*.

4 Local Search Algorithm

The multicut problem is known to be NP-hard [14]. Various cutting-plane and branch-and-bound algorithms [2], [15], [75], [77], [82], [83], [84] do not scale to instances of

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2022.3148795, IEEE Transactions on Pattern Analysis and Machine Intelligence

Algorithm 1: Kernighan-Lin Algorithm. The function UPDATE_BOUNDARY is given in Alg. 2.

Data: weighted, undirected graph $G = (V, E)$, lifted edges F, cost function c, starting 01-edge labeling $y^0 \in \{0,1\}^{E \cup F}$
Result: 01-edge labeling y^{t-1}

1 $t \leftarrow 1$
2 **while** $t < max\_iter$ **and** $y^t \neq y^{t-1}$ **do**
3 **foreach** $(A, B) \in$ *adjacent_components*(y^{t-1}) **do**
4 $y^t \leftarrow$ update_boundary(G, F, c, A, B)
5 **foreach** $A \in$ *components*(y^t) **do**
6 $y^t \leftarrow$ update_boundary$(G, F, c, A, \emptyset)$

the size we consider even for 2nd-order problems, c.f. the comparative study [17]. Indeed, we implemented a branch-and-bound method in Gurobi [85] for the 3rd-order case by linearizing the objective (1), but we could not obtain a solution even for the smallest problem we consider in 12 hours. Toward scalable algorithms, Keuper et al. [3] define a generalization of Kernighan and Lin's primal local search algorithm for graph partitioning problems to the case of lifted multicut problem. It shows the best performance for this problem in [17]. We generalize their algorithm further to lifted multicut problems to include costs of arbitrary order.

Overview The algorithm takes as input an instance of the higher-order lifted multicut problem and an initial decomposition of G and outputs a decomposition of G whose higher-order lifted multicut has an objective value lower than or equal to that of the initial decomposition. As the original KLj-Algorithm [3], it always maintains, throughout its execution, a feasible decomposition of G. The pseudo-code is given in Alg. 1. New components are introduced by updating a boundary of a component against an empty set $\emptyset$, as given by lines 5–6, exactly as in the 2nd-order version [3].

Function UPDATE_BOUNDARY (Alg. 2) receives two components A and B and updates the cut between only them. It constructs a sequence M of elementary transformations of the components A and B greedily such that every consecutive move-operation increases the cumulative gain S maximally (or decreases it minimally). Therefore, the operation COMPUTE_GAINS computes, at the beginning of each execution of UPDATE_BOUNDARY, for every element $v \in A \cup B$ the difference in the objective function (gain) when v is moved from A to B or from B to A. These differences are updated as described in Alg. 2, ll. 9-21. To escape local optima, we determine $i^* = \text{argmax}_i S_i$ such as to maximize the total gain of the *sequence of operations*. If the objective value can be decreased by executing either the first i^* elementary transformations or by joining the components A and B the optimal of these two operations is carried out. While components are defined with respect to the graph $G = (V, F)$, differences in objective value are computed with respect to the graph $G' = (V, E \cup F)$.

In [3] as well as in our algorithm, all transformations of feasible solutions are local, resulting in changes of the objective value that are computed in linear time (in the size of the graph). The combination of the locality of individual transformations and the non-locality of sequences of transformations has proven effective for diverse applications [17]. As in KLj [3], the number of outer iterations of Alg. 1 is not bounded by a polynomial and we cannot give any guarantee for convergence. However, in practice, the algorithm converged in less than 50 iterations for the experiments described in Sec. 6.

Algorithm 2: Function UPDATE_BOUNDARY greedily moves vertices from one component to the other.

Data: weighted, undirected graph $G = (V, E)$, lifted edges F, cost function c, a pair of partitions A and B
Result: 01-edge labeling y

1 $D^{A \cup B} \leftarrow$ compute_gains(G, F, c, A, B)
2 $\Omega \leftarrow$ find_boundary_nodes(V, E, A, B)
3 $\Delta_{join} \leftarrow$ compute_gain_from_joining(G, F, c, A, B)
4 $S_0 = 0$
5 $M = [\,]$ `// array to store moves`
6 **for** $i \leftarrow 1$ **to** $|\Omega|$ **do**
7 $v^* \leftarrow \arg\max_{v \in \Omega} D^{A \cup B}$
 `// w.l.o.g. let` $v^* \in A$
8 **foreach** $U \in \{U' \mid U' \in \mathcal{V}, v^* \in U', U' \subseteq A \cup B\}$ **do**
9 $U' \leftarrow U \setminus \{v^*\}$
10 **if** $U' \subseteq A$ **then**
11 **foreach** $w \in U'$ **do**
12 $D_w \leftarrow D_w - c_U$
13 **else if** $U' \subseteq B$ **then**
14 **foreach** $w \in U'$ **do**
15 $D_w \leftarrow D_w + c_U$
16 **if** $|U' \cap A| = 1$ **then**
17 $w \leftarrow U' \cap A$
18 $D_w \leftarrow D_w - c_U$
19 **if** $|U' \cap B| = 1$ **then**
20 $w \leftarrow U' \cap B$
21 $D_w \leftarrow D_w + c_U$
22 M.push(v^*) `// move` v^* `from` A `to` B
23 $S_i \leftarrow S_{i-1} + D_{v^*}$ `// cumulative gain`
24 $\Omega \leftarrow$ update_boundary(v^*, E, A, B)
25 $i^* \leftarrow \text{argmax}_i S_i$ `// best number of moves`
26 **if** $\Delta_{join} > S_{i^*}$ **and** $\Delta_{join} > 0$ **then**
27 join_components(y, A, B)
28 undo_moves(y, A, B, i^*) `// undo moves after` i^*

Implementation Details For efficiency, we pre-compute all the gains for vertices in $A \cup B$ (line 1) and keep track of the vertices, that currently lie on the boundary Ω between A and B (lines 2 and 21); this dramatically improves the runtime for sparse graphs. We iteratively pick a vertex v^* with the largest gain (line 7), that can also be negative. Then, we update gains of all other vertices in A and B, that are in subsets U that contain v^* (lines 8–18). Note, that in the case of 2nd-order costs updates as given in lines 10–18 specialize to the corresponding updates in [3]. In the end, we find, which first i^* moves produce the greatest decrease (note, i^* can be also 0), and either merge A and B together (lines 23–24) or undo the moves after i^* (line 25).

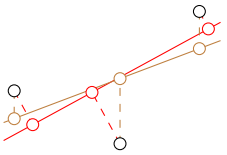
5 Graph Construction

5.1 Geometric Model Fitting

We cast geometric model fitting as a point grouping problem or in other words as a graph decomposition problem

which is defined over local observations. Specifically, we consider minimal residual errors of n-tuples of points as local features indicating these points' likelihood of being sampled from the same model. Given this local evidence, we can phrase the geometric model fitting problem as a higher-order minimum cost multicut problem.

We compute residual errors relative to a model fit using total least squares (TLS). Unlike least squares (brown color), that minimize only axis-aligned residuals, total least squares (red color) minimize the distance between a point and its projection on the model. We solve TLS using Singular Value Decomposition.



For the minimal set plus 1 of data points, where the minimal set is the set of points required to estimate model parameters, we compute a probability for these points to belong together, i.e. being sampled from the same model. This probability is inversely proportional to the points' residuals to the estimated model. For line models, the minimum set is of cardinality two. Therefore, we need edges of at least order three to assign such probabilistically motivated costs.

Line Fitting For line fitting, we create a fully-connected graph over all $\binom{V}{2}$ vertices, that defines the set of feasible decompositions. The cost function c is defined over all $\mathcal{V} = \binom{V}{3}$ as follows: We fit a line into each triplet $\{u, v, w\} \in \mathcal{V}$ using TLS and compute their residuals r, an example is given in Fig. 2 (a). We assume, that points have been sampled from a Gaussian centered on the ground-truth line with a standard deviation of σ. That makes $p_v(r_v) = \text{erfc}(r_v; 0, \sigma^2)$, where $\text{erfc}(\cdot)$ is the complementary error function. Further assuming that all the points are i.i.d., we get $p_{uvw}(y_{uv}y_{uw}y_{vw} = 1 \mid r_u, r_v, r_w) = p_u(r_u)p_v(r_v)p_w(r_w)$. Finally, the cost is $c_{uvw} = \log \frac{1-p_u(r_u)p_v(r_v)p_w(r_w)}{p_u(r_u)p_v(r_v)p_w(r_w)}$. This corresponds to minimizing the sum of residuals w.r.t. a non-noisy line.

Homography and Motion Estimation For homography and motion estimation we proceed in the same manner as described above. For this task, we have to subsample cost terms, as $\mathcal{V} = \binom{V}{5}$ is a prohibitive for $|V| > 150$. For every vertex we thus sample all $\binom{20}{5}$ subsets of its 20 nearest neighbors (in the image pixel space) to capture the local scope and $2 \cdot 10^5$ random subsets to capture the global scope. We fit an elementary homography into each of these 5 pairs of points and assume the points' probability to be inversely proportional to the distance between the ground-truth correspondence and its projection via the fit homography, c.f. Fig 2 (b). This corresponds again to minimizing the sum of re-projection errors.

5.2 Motion Segmentation

5.2.1 Point Trajectories

Point trajectories are spatio-temporal curves that describe the trajectory of a single object point in the image plane. They build the basis for many motion segmentation methods such as [4], [25], [31], [86], [87]. Here, we use the method from [25] to generate dense long-term point trajectories from precomputed optical flow [88] to allow for a direct comparison to prior work. For a video of length N, [25] yields n point trajectories p_i with the maximum length

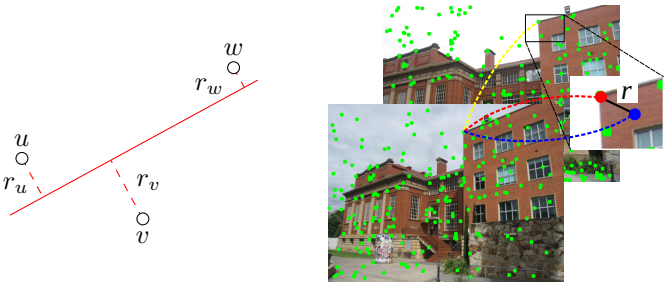


Fig. 2. **(a) Line fitting**: We fit a line using TLS into a set of points and assume that the latter are independently drawn from a 1D Gaussian centered on the line and orthogonal to it. **(b) Homography Estimation**: For a pair of images with annotated correspondences (yellow line) we directly model the distance r between the corresponding and projected point. In the example above, a point from the second image corresponds to a red dot, but its projection via the estimated homography is a bit off (blue dot). Uncertainty model corresponds to a 2D isotropic Gaussian centered at the red point.

N, where n depends on the desired sampling rate. Due to occlusions and mistakes in the optical flow estimation, most trajectories are significantly shorter than N, and some trajectories start after frame 1 to ensure even point sampling throughout the sequence.

5.2.2 Higher-Order Motion Models

Although it is not sufficient to accurately describe object motion in a 3D environment recorded with a possibly moving camera, we restrict ourselves to edge potentials of order two and three for practical reasons. This allows to measure the difference of point motions according to Euclidean motion models, i.e. from the group of transformations describing translation, rotation and scaling in the 2D plane. This is a subset of the group of similarity transformations in the 2D plane, with reflections excluded.

We further argue that in any case, the easiest model that can explain the motion of a set of points with a single transformation should be used. If two points are moving according to the same translational motion model, we can assume that they belong to the same object without looking at further points around them. Only if their motion is different according to a purely translational model, looking at more complex motion models adds information. This results in a *motion-adaptive graph construction* strategy.

Motion-Adaptive Graph Construction We propose to construct the higher-order graph G from the pairwise costs computed from motion differences. The algorithm is described in Alg. 3. For any pair of trajectories, we compute their cost of belonging to the same translational motion model. Only if this cost is positive, i.e. repulsive, we look at all further points to compute for every three-tuple the cost of belonging to the same motion model for translation, rotation and scaling. The respective third-order edges are inserted along with their costs.

This strategy allows to integrate second and third order potential without losing model capacity. Further, compared to generating the full graph with higher-order potentials, it yields a significant space reduction in practice.

Lifted Graph Construction To construct higher-order lifted graph $G' = (V, E \cup F)$, we compute for every trajec-

Algorithm 3: Motion Adaptive Graph Construction.

Data: set of point trajectories V with $p_k \in V$ with $k \in \{1 \dots n\}$
Result: weighted undirected higher-order graph $G = (V, E)$, cost vector c

```
G ← (V, E = ∅)
c = [ ]
foreach (u, v) ∈ (V 2) do
    c_uv ← compute_translational_motion_cost(p_u, p_v)
    E ← E ∪ (u, v)                          // add edge
    if c ≤ 0 then
        c.push(c_uv)
    else
        foreach w ∈ V \ {u, v} do
            c_uvw ←
              compute_HO_motion_cost(p_u, p_v, p_w)
            E ← E ∪ (u, w) ∪ (v, w)          // add edge
            c.push(c_uvw)
```

tory the set of its 12 spatially nearest neighbors $\mathcal{N}$. The edge set E of edges between direct neighbors in G' is computed according to Alg. 3. It contains exactly all pairwise edges $e_{ij} \in E$ for which at least one of the following three conditions holds: (1) $p_i \in \mathcal{N}(p_j)$, (2) $p_j \in \mathcal{N}(p_i)$ (3) the maximum spatial distance between p_i and p_j is below 40 pixels.

Second Order Costs Second order costs are computed from pairwise differences on point trajectories. We compute such differences only for trajectories which have at least two frames in common. Since it has proven successful in previous work [4], we compute such differences based on motion, color and spatial distance cues. As suggested by [31], we define the pairwise motion difference of two trajectories at time t as

$$d_t^{\text{motion}}(p_i, p_j) = \frac{\|\partial_t p_i - \partial_t p_j\|}{\sigma_t}. \tag{5}$$

Here, $\partial_t p_i$ and $\partial_t p_j$ are the partial derivatives of p_i and p_j with respect to the time dimension and σ_t is the variation of the optical flow as defined in [31]. The motion distance of two trajectories is defined by the maximum over time

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

As proposed in [4] color and spatial distances d^{color} and d^{spatial} are computed as average distances over the common lifetime of two trajectories. These three cues are combined non-linearly to compute the costs

$$\begin{aligned} c_{ij} \quad = \quad \max\big(\bar{\theta}_0 \quad &+ \quad \theta_1 \quad d^{\text{motion}}(p_i, p_j) \\ &+ \quad \theta_2 \quad d^{\text{spatial}}(p_i, p_j) \\ &+ \quad \theta_3 \quad d^{\text{color}}(p_i, p_j)\,, \\ \theta_0 \quad &+ \quad \theta_1 \quad d^{\text{motion}}(p_i, p_j) \qquad \big) \end{aligned} \tag{7}$$

with weights and intercept values θ as proposed in [4][1].

Third Order Costs We compute third order motion differences as proposed in [27]. For any two trajectories p_i and p_j we estimate the Euclidean motion model $\mathcal{T}_{ij}(t)$, consisting of rotation R_α, translation $v := (v_1, v_2)^\top$ and scaling s as

$$\begin{aligned} \alpha &= \arccos\left(\frac{(p_i(t') - p_j(t'))^\top (p_i(t) - p_j(t))}{\|p_i(t') - p_j(t')\| \cdot \|p_i(t) - p_j(t)\|}\right) \\ s &= \frac{\|p_i(t') - p_j(t')\|}{\|p_i(t) - p_j(t)\|} \\ v &= \frac{1}{2}\left(p_i(t') + p_j(t') - sR_\alpha(p_i(t) + p_j(t))\right), \end{aligned} \tag{8}$$

where t denotes the first point in time where both trajectories co-exist and t' is the last. The distance to any third trajectory p_k existing from t to t' can then be measured by $d_{ij}^t(p_k) = \|\mathcal{T}_{ij}(t)p_k(t) - p_k(t')\|$. For numerical reasons, $d_{ij}^t(p_k)$ is normalized by

$$\gamma_{ij}^t = \frac{1}{\sigma_t}\left(\frac{1}{2}\left(\frac{\|p_i(t) - p_j(t)\|}{\|p_i(t) - p_k(t)\|} + \frac{\|p_i(t) - p_j(t)\|}{\|p_j(t) - p_k(t)\|}\right)\right)^{\frac{1}{4}}, \tag{9}$$

with σ_t being the optical flow variation as in (5).

To render distances symmetric, [27] propose to consider the maximum $d_{\max}^t(i,j,k) = \max(\gamma_{ij}^t d_{ij}^t(p_k), \gamma_{ik}^t d_{ik}^t(p_j), \gamma_{jk}^t d_{jk}^t(p_i))$, which yields an over-estimation of the true distance. While this is unproblematic in a spectral clustering scenario, where distances are used to define positive point affinities, it can lead to problems in the multicut approach. Over-estimated distances lead to under-estimated join probabilities and thus eventually to switching the sign of the cost function towards repulsive terms. To avoid this effect, we compute both $d_{\max}^t(i,j,k)$ and, analogously, $d_{\min}^t(i,j,k)$. For both, we compute the maximum motion distance over the common lifetime of p_i, p_j and p_k as $d_{\max}(i,j,k) = \max_t d_{\max}^t(i,j,k)$ and $d_{\min}(i,j,k) = \max_t d_{\min}^t(i,j,k)$. We evaluate the costs $c(d_{\max}(i,j,k))$ and $c(d_{\min}(i,j,k))$ for both distances as $c(d) = \theta_0 + \theta_1 d$ and compute the final edge costs

$$c_{ijk} = \begin{cases} c(d_{\min}(i,j,k)) & \text{if} \quad c(d_{\max}(i,j,k)) > 0 \\ c(d_{\max}(i,j,k)) & \text{if} \quad c(d_{\min}(i,j,k)) < 0 \\ 0 & \text{otherwise.} \end{cases} \tag{10}$$

Thus, we make sure not to set any costs for edges whose underlying motion is controversial. Here, we set $\theta_0 = 1$ and $\theta_1 = -0.08$ manually.

Implementation Details In practice, we insert pairwise edges e_{ij} in G and G' only if the spatial distance between p_i and p_j is below 100 pixels even for lifted edges in F. This is in analogy to [4] and due to the fact that for nearby points, the approximation of the true motion by a simplified model is usually better than for points at a large distance. Also, since the number of pairwise edges increases quadratically with the maximal spatial distance, this heuristic decreases the computational load significantly. For the same reason, we introduce an edge sampling strategy for third order edges. For every triplet of points, we compute the maximum pairwise distance d. From all triplets with $20 < d < 300$, we randomly sample $\frac{100}{d^2}\%$, while we insert all edges e_{ijk} with $d \leq 20$. This also prevents from a too strong imbalance of long range edges over short range edges.

1. Specifically, $\bar{\theta}_0 = 6$, $\theta_0 = 2$, $\theta_1 = \theta_3 = -0.02$ and $\theta_2 = -4$.

TABLE 1
Quantitative results on synthetic line fitting data from Toldo and Fusiello [21]. The total runtime is split into sampling and solving time.

	J-Linkage	T-Linkage	RansaCov†	RS-NMU	Our
Misclassification error [%, ↓]					
Stairs4	10.6	8.6	4.0	3.0	4.2
Star5	5.4	7.4	2.0	1.0	2.2
Star11	12.0	9.55	2.73	1.91	2.64
mean	9.33	8.52	2.91	1.97	3.01
Runtime (sampling/solving) [s, ↓]					
Stairs4	58/30	58/120	58/42	58/36	10/5
Star5	54/15	54/54	54/15	54/11	10/6
Star11	294/112	294/1652	294/113	294/210	107/42

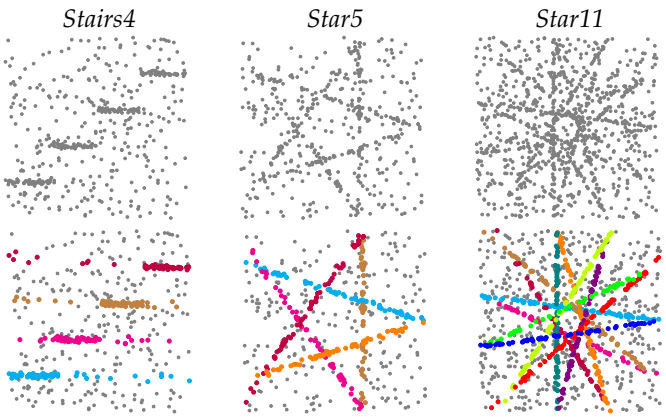


Fig. 3. Qualitative results on synthetic line fitting data from Toldo and Fusiello [21]. Colored dots denote points assigned to the same line model. Gray dots denote points assigned to noise/background.

6 Experiments

6.1 Geometric Model Fitting

We start with line fitting experiments on synthetic data from Toldo and Fusiello [21]. This dataset consists of 3 instances with lines arranged in different shape (namely, stairs with 4 lines and star with 5 and 11 lines) and perturbed by Gaussian noise. Each instance contains around 50% of uniformly sampled gross outliers, c.f. Fig. 3 top row.

Here, we compare against J-Linkage [21], T-Linkage [20], RansaCov [33], and RS-NMU [35]. We use the publicly available code of these methods and hand tune their parameters for optimal performance on each instance. It has been shown in [21] that J-Linkage outperforms multiRANSAC [19], residual histogram analysis [89], and mean-shift [42].

As a performance measure we use the widely accepted [20], [21], [33], [35] misclassification error (ME), that is the ratio of the misclassified points over the total number of points. The classification is performed by matching the predicted and the ground truth clusters such that the number of misclassified points is minimized using the Hungarian algorithm [90] for minimum weight bipartite matching. The point is then considered to be correctly classified if its cluster label corresponds to the matched ground truth one.

Xiao et al. [23] notice that it is impossible to estimate the number of models present in the data without introducing additional stronger assumptions. For example, if we assume that points lying on a line constitute a model, then many more lines can be discovered in the presence of many outliers, as is the case of the data from Toldo and Fusiello [21]. Indeed, the term "outliers" here is rather subjective, as any two points perfectly define a line. A common solution [20], [21], [33] is to simply take the top-K largest clusters and assign the rest to the "outliers" class, where K is the number of clusters in the ground-truth. A more principled approach was proposed by Tepper and Sapiro [35], it is based on the statistical t-test that allows to keep only statistically significant models. In this paper we adopt the usual strategy and match the predicted models to the ground truth ones so as to minimize the misclassification error.

The results are reported in Tab. 1. All competing methods sample a small number of hypotheses before optimization, while we sample all possible triplets. For a fair comparison we sample all possible hypotheses for the competing methods as well. However, this dramatically increases runtimes of these methods. In our experiments we noticed, that the statistical filtering of Tepper and Sapiro [35] very efficiently gets rid of irrelevant ones, thus, we apply it to the sampled hypotheses before optimization. It can be seen from Tab. 1, that our method is the fastest at the same time showing competitive accuracy, especially on the hardest problem *Star11*. Visualizations of our results are presented in Fig. 3.

We analyze the behavior of our method regarding possible uniformly random sub-sampling of the cost terms in Fig. 4 (a). It can be seen, that our method can get away with as little as 5% of all the possible triplets. As noted earlier, our method discovers more lines, than there is in the ground-truth, because any 2 points define a line in the Euclidean space. This corresponds to over-segmentation of the graph decomposition we obtain. Yet, if we plot the costs of the resulting clusters as given in Fig. 4 (b), computed according to the objective function (1), it is rather easy to spot the models (lines) of interest. We think, this may allow to develop a method for automatic selection of models.

Next we turn to the real-world challenge posed by Adelaide Robust Model Fitting (AdelaideRMF) data set [91]. It consists of 38 image pairs with a set of interest points and human-annotated correspondences for each pair. The dataset is split into two equal parts and offers two challenges: 1) estimate multiple homographies (a transformation that relates points belonging to the same planar surface in 3D space) and 2) estimate multiple motion models.

Numerical results are shown in Tab. 2, some visual results are provided in Fig. 5. Here we also compare against Random Cluster Models Simulated Annealing (RCMSA) [92], Robust Preference Analysis (RPA) [34], FABIA [36], Multi-X [41], Mode-Seeking on Hypergraphs Fitting (MSHF) [46], and Convex Relaxation Algorithm (CORAL) [40]. Our model yields competitive performance with a particularly low error for motion model estimation.

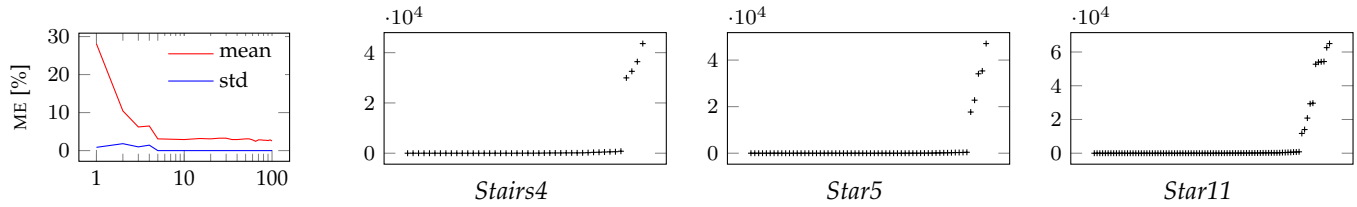
6.2 Motion Segmentation

First, we apply the proposed higher-order lifted multicut model on the motion segmentation benchmark FBMS-59 [31], an extended version of the BMS-26 benchmark from Brox-Malik [25].

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2022.3148795, IEEE Transactions on Pattern Analysis and Machine Intelligence

TABLE 2
Misclassification errors on the Adelaide Robust Model Fitting data set [91]. Our model yields competitive results at low standard deviation. † these methods require the exact number of models before optimization.

	RCMSA	J-Lnkg	T-Lnkg	RPA†	FABIA†	RCov†	Multi-X	MSHF	RS-NMU	CORAL	Our
Homography Estimation [%, ↓]											
mean	28.30	25.50	24.66	17.20	15.94	12.91	9.72	7.38	5.82	4.21	10.01±0.14
median	29.40	24.48	24.53	17.78	17.96	12.34	2.49	2.37	2.07	3.48	5.61±0.00
Motion Estimation [%, ↓]											
mean	12.37	16.43	9.37	5.49	4.61	6.04	2.97	7.41	5.72	–	3.45±0.06
median	9.87	14.29	7.80	4.57	1.66	4.27	0.00	2.44	3.64	–	2.09±0.02



(a) Costs sub-sampling **(b) Sorted clusters costs**

Fig. 4. **(a)** Mean ME and standard deviation of our method on *Star11* averaged over 20 runs as functions of the sub-sampling rate of the cost edges; σ is fixed. Already at 5% of all the $\binom{1100}{3}$ possible cost edges the mean error becomes almost optimal and the standard deviation drops to 0. **(b)** Plotted are sorted absolute costs per cluster in the optimal solutions for *Stairs4*, *Star5*, and *Star11*. For each problem the corresponding 4, 5, and 11 most expensive clusters are visually clearly separated form the lower cost clusters, the latter being lines fit into the outliers.

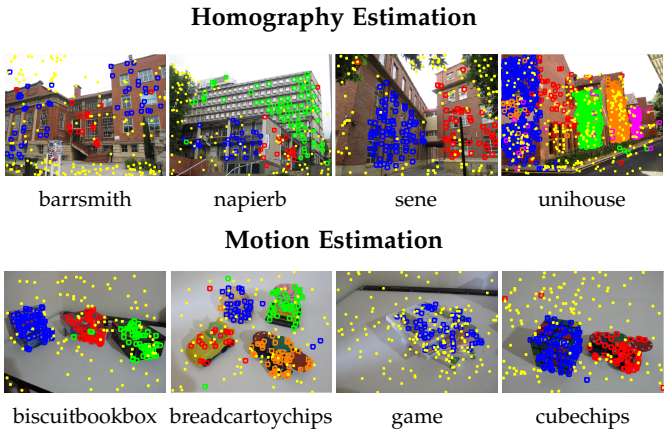


Fig. 5. Some visualizations of our results on the Adelaide Robust Model Fitting data set [91]. Only one image of the pair is shown. Colored boxes denote points assigned to the same geometric model. Yellow dots denote points assigned to noise/background.

It contains 59 sequences of varying length (from 19 to 800 frames) and diverse content and motion: severe camera shaking, non-rigid multi-object motion, scaling, out-of-plane rotation of objects, zooming of the camera, etc..

To allow for training, the data set has been split into two subsets of 29 and 30 sequences for training and testing, respectively. While we agree that training all model parameters is highly desirable, we did not do so. This is due to the fact that (1) neither of the state-of-the-art methods [4], [27], [31] is training-based and (2), the training set, with 29 sparsely annotated sequences, is rather small. Thus, to avoid confusion, we hence denote the training split by *Set A* and the test split by *Set B*.

FBMS-59 [31] provides manual annotations for all moving objects in the videos for every 20-th frame as well as *ground truth definition* files that downweight annotated segments in some scenes. Thus, objects in some sequences that contain severe camera motion, for example, a mistakenly segmented wall in Fig. 9, attain a lower weight in the evaluation. All objects that move in at least one frame are segmented in all annotated frames. A second set of ground truth annotations at a similar level of sparsity has later been provided in [93] to evaluate a slightly different motion segmentation paradigm. In [73], [93], it was argued that all freely moving objects in 3D space but nothing more should be segmented per frame. Specifically, this means that objects moving only in few frames are only to be segmented in these frames. Scene geometry and camera motion yielding apparent motion in the image plane, such as the wall in the example of Fig. 9, are not considered. Our work addresses the task originally annotated by [31]. It aims to segment all objects that move in at least one frame of a sequence and does not provide full 3D motion segmentations, as we limit ourselves to third order motion terms. Yet, we find it interesting to consider both sets of annotations during evaluation to allow for a comparison to the respective competing methods. Note, that also the evaluation metrics differ slightly. Both [31] and [93] measure precision, recall and f-measure, where [31] evaluate precision and recall over all frames and compute the f-measure in the end, while [93] evaluate the f-measure per frame and report the mean value. In addition, [31] report the number of objects O that are segmented with an f-measure above 0.75. Instead, [73] propose the ΔObj metric, which measures the average absolute difference between the number of ground truth objects in each frame and the number of segmented objects in this frame. While O should be large, ΔObj should be small.

For our evaluation, we employ the annotations provided

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2022.3148795, IEEE Transactions on Pattern Analysis and Machine Intelligence

Fig. 6. Samples of our Lifted AOMC segmentations densified by [58]. Even for articulated motion, our segmentations show little over-segmentation.

TABLE 3
Segmentation results on the FBMS-59 dataset on Set A (top) and Set B (bottom). We report **P**: average precision in %, **R**: average recall in %, **F**: F-measure in % and **O**: extracted objects with F $\geq 75\%$. All results are computed for sparse trajectory sampling at 8 pixel distance. Our result **HO MC** is computed on the non-lifted purely higher-order model to allow for a direct comparison to the listed competing methods.

Set A (29 sequences)	P [%, ↑]	R [%, ↑]	F [%, ↑]	O [↑]
SC [31]	85.10	62.40	72.0	17/65
Higher-Order SC [27]	81.55	59.33	68.68	16/65
MC [4]	84.94	71.22	77.48	23/65
HO MC (ours)	83.20	74.34	**78.52**	**29/65**
Set B (30 sequences)	P [%, ↑]	R [%, ↑]	F [%, ↑]	O [↑]
SC [31]	79.61	60.91	69.02	24/69
Higher-Order SC [27]	82.11	64.67	72.35	**27/69**
MC [4]	82.87	69.89	**75.83**	**27/69**
HO MC (ours)	82.92	68.82	75.22	**27/69**

TABLE 4
Segmentation Results on FBMS-59 on Set A (top) and Set B (bottom). We report **P**: average precision, **R**: average recall, **F**: F-measure and **O**: extracted objects with F $\geq 75\%$. All results are computed for sparse trajectory sampling at 8 pixel distance. The proposed approach **Lifted AOMC** performs best.

Set A (29 sequences)	P [%, ↑]	R [%, ↑]	F [%, ↑]	O [↑]
MCe [4]	86.73	73.08	79.32	31/65
HOPMC	87.66	74.15	80.34	31/65
AOMC	82.29	76.17	79.11	32/65
Lifted HOPMC	87.07	70.84	78.12	28/65
Lifted AOMC	86.20	78.35	**82.08**	**34/65**
Set B (30 sequences)	P [%, ↑]	R [%, ↑]	F [%, ↑]	O [↑]
MCe [4]	87.88	67.7	76.48	25/69
HOPMC	85.00	67.75	75.40	25/69
AOMC	84.48	73.08	78.37	**27/69**
Lifted HOPMC	87.07	70.84	78.12	28/69
Lifted AOMC	87.82	71.45	**78.79**	24/69

in [31] when considering the metric proposed therein. We use the annotations from [93] when evaluating using the metric proposed in [73] to allow for direct comparison to their work. We start by an evaluation using annotations and metrics from [31].

Evaluation To assess the capacity of our model components, we first evaluate a purely higher-order non-lifted version of our model. In this model, all pairwise costs are removed and all edges are connectivity defining. We compare this simple model to [27], [31] and the purely motion-based version of [4]. While [31] and [4] only consider translational motion, the affinities in [27] are defined most similarly to our higher-order costs. As the proposed approach, [4] formulate a multicut problem while [27], [31] follow a spectral clustering approach. The results are given in Tab. 3 in terms of precision, recall, F-measure and the number of extracted objects. Precision and recall are not directly comparable but they can serve as cues for under- or over-segmentation. The F-measure is a weighted harmonic mean of the two. From Tab. 3, we can observe that our higher-order lifted multicut model outperforms the higher-order spectral clustering method from [27] by about 10% on Set A and 3.5% on Set B. While there is a clear improvement on both sets, the imbalance is remarkable. A similarly remarkable imbalance can be observed when comparing the performance between the two spectral clustering methods [31] and [27]. The higher-order model [27] results in a lower F-measure on Set A compared to [31]. Yet it outperforms [31] on Set B by about 3%. This indicates that the motion statistics in both splits are significantly different. When we compare our higher-order model to the pairwise minimum cost multicut model from [4], we can observe an improvement on Set A. On Set B, both models perform almost equally.

In Tab. 4, we show the evaluation of our higher-order multicut model with the motion-adaptive order, denoted AOMC (compare Alg. 3). This model has access to similar pairwise cues as the motion and color-based version from [4], denoted MCe. It has as well access to the higher-order motion cues from equation (10).

As a sanity check for the motion-adaptive graph construction, we also generate graphs that simply contain all pairwise costs c_{ij} as well as all third order edges with costs c_{ijk} without any adaptation with respect to the costs. We denote this additive model by HOPMC (higher-order + pairwise multicut). On Set A, all three approaches produce similar results, whereas the proposed AOMC shows particularly good performance on the test set with about 2% improvement over MCe [4] in f-measure.

Lifted versions of both types of problems (HOPMC and AOMC) yield a small further improvement on Set B. However, on Set A, the segmentation quality of Lifted HOPMC is even below the one of MCe by about 1%. In contrast, the proposed Lifted AOMC consistently outperforms all competing methods and baselines.

In Tab. 5, we evaluate the impact of the quality of

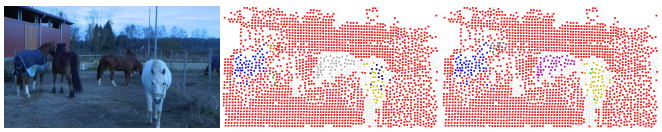
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2022.3148795, IEEE Transactions on Pattern Analysis and Machine Intelligence

TABLE 5
Segmentation results of the proposed model Lifted AOMC on the FBMS-59 dataset on Set A (top) and Set B (bottom) for different optical flow methods. We report **P**: average precision, **R**: average recall, **F**: F-measure and **O**: extracted objects with F $\geq 75\%$. All results are computed for sparse trajectory sampling at 8 pixel distance.

Set A	Flow	P [%, ↑]	R [%, ↑]	F [%, ↑]	O [↑]
MCe [4]	LDOF [88]	86.73	73.08	79.32	31/65
Lifted AOMC		86.20	78.35	82.08	34/65
MCe [4]	FlowNet [94]	89.63	73.38	80.69	29/65
Lifted AOMC		88.22	77.34	82.42	33/65
MCe [4]	FlowNet [95]	89.77	75.78	82.19	34/65
Lifted AOMC		89.19	79.35	83.98	38/65
Set B	Flow	P [%, ↑]	R [%, ↑]	F [%, ↑]	O [↑]
MCe [4]	LDOF [88]	87.88	67.7	76.48	25/69
Lifted AOMC		87.82	71.45	78.79	24/69
MCe [4]	FlowNet [94]	86.73	68.77	76.71	26/69
Lifted AOMC		86.89	69.81	77.42	24/69
MCe [4]	FlowNet [95]	84.59	7019	76.72	27/69
Lifted AOMC		88.39	72.12	79.43	26/69

TABLE 6
Results for densified segmentations on FBMS-59 using annotations and metrics as in [73] for freely moving 3D objects. For ΔObj, lower is better. Results for [4], [96], [59], [60] and [73] are taken from [73].

		P [%, ↑]	R [%, ↑]	F [%, ↑]	ΔObj [↓]
	MCe [4]	74.64	62.03	63.59	7.7
	Taylor at al. [96]	72.69	54.36	56.32	11.7
	Bideau et al. [73]	74.23	63.07	64.97	4
	Lifted AOMC	73.29	60.26	63.58	4.41
binary	Tokmakov at al. [59]	87.29	72.19	74.79	-
binary	Tokmakov et al. [60]	92.40	85.07	86.96	-
binary	Lifted AOMC	79.98	76.97	76.16	-



MCe [4] ours

Fig. 7. The scaling motion of the white horse moving towards the camera causes over-segmentation with a simple motion model [4]. With the proposed Lifted AOMC, this can be avoided.

the point trajectories when they are computed from different optical flow models [88], [94], [95]. The proposed approach Lifted AOMC consistently outperforms the pairwise MCe [4] on comparable optical flows. While the most recent FlowNet [95] performs best, the overall differences are small.

Several examples of pixel-segmentations computed from our sparse segmentation using [58] are given in Fig. 6. The densified segmentations look reasonable. On the bear example, the articulated leg motion still causes some over-segmentation. One of the horses in the *horses05* sequence is missed. However, even small objects such as the tray in the *marple12* sequence or the phone in the *marple13* sequence can be correctly segmented. Such densified segmentations, computed from sparse results using FlowNet [95], can be evaluated by the metrics from [73] with their matching annotations [93]. Tab. 6 shows our results in the setting by [73], i.e. considering a frame-wise evaluation on all freely moving 3D objects. While the f-measure of our model is similar to the one reached by [4], the ΔObj metric is significantly improved, i.e. lower, and almost on par with the results from [73], which are dedicated to this setting. Evaluating binarized segmentations (Tab. 6) allows for a comparison to learning based encoder-decoder models such as [59]. Without learning any prior on object saliency, our f-measure for binary segmentation on FBMS-59 is 76.16% and thus slightly better than the learnt model from plain motion cues in [59] with 74.79%, but inferior to [60], who learn an additional appearance stream and reach 86.96% [73]. In the following, we discuss several example segmentations in detail.

Fig. 7 shows the trajectory segmentation quality under scaling. In the *horses05* sequence, scaling is caused by the motion of the white horse towards the camera. This causes over-segmentation in the competing method MCe [4], which can not handle higher-order motion models. With the proposed Lifted AOMC, the segmentation can be improved.

Fig. 8 and 9 both show examples where the same label is assigned to distinct objects that move similarly. In the *cars2* sequence in Fig. 8, this is due to similar real-world object motion, whereas, in the *marple10* sequence, the effect is due to camera motion and the scene geometry. In both cases, the formulation of the Lifted AOMC problem allows to tell the distinct objects apart. However, in the *marple10* sequence (Fig. 9), we can observe a spurious segment in the background, which is probably caused by imprecise flow.

In Fig. 10, we show an example of the *goats01* sequence. Here, the head and body of the goat in front are segmented into a distinct components by the pairwise method [4], because of the expressed articulated motion. Although our third order model can not explicitly handle articulation, the over-segmentation can be fixed in this case.

Fig. 11 shows a failure case of the proposed method. Due to the dominant camera motion in a scene with complex geometry, the Euclidean motion model fits particularly badly. Thus our model leads to the segmentation of the scene into its depth layers, and thus to strong over-segmentation.

Next, we evaluate our approach on two additional datasets widely considered for motion segmentation, the DAVIS_{16} dataset [100] and the VSB100 dataset [97], [98]. Both have originally been designed for different purposes. DAVIS_{16} is a dataset for binary video object segmentation which has been used to learn appearance and motion patterns of salient objects in videos, e.g. in [60]. While the sequences may contain several moving objects, the task is to track the segmentation of the dominant object throughout the sequence. Complementary to this, the VSB100 dataset was originally proposed as a video segmentation dataset where the task is to mimic human boundary level annotations, i.e. the segments do not necessarily have a notion of objectness. This general purpose multi-label video segmentation dataset has a motion subtask which can be used to evaluate motion segmentation approaches before, e.g. as in [4]. Tab. 7 shows our results on the DAVIS_{16} dataset in terms of Jaccard index (J) and the f-measure (F) which measures the boundary fidelity of the segmentation. We compare to

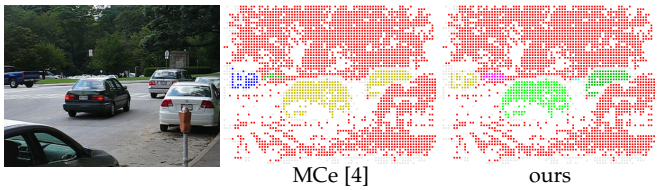


Fig. 8. The two cars in front move to the same direction, leading an assignment to the same cluster with the non-lifted multicut approach [4]. The Lifted AOMC can assign the different cars to distinct segments.

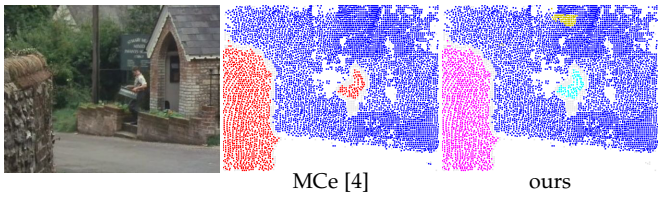


Fig. 9. Due to camera motion the person and the wall are assigned to the same cluster with the non-lifted multicut approach [4]. The Lifted AOMC allows for correct segmentation.

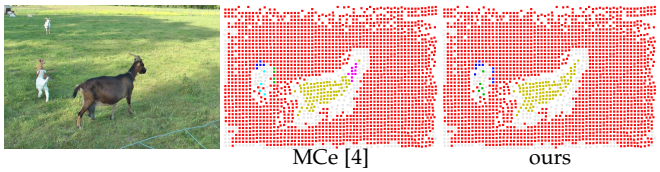


Fig. 10. The articulated motion causes over-segmentation in [4]. The Lifted AOMC performs better.

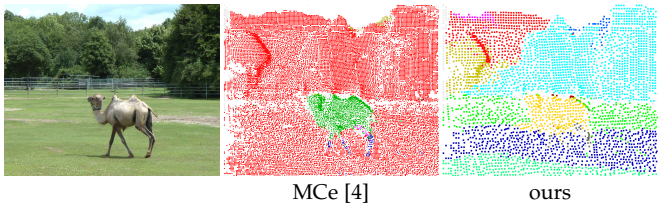


Fig. 11. Failure case. The dominant camera motion causes strong over-segmentation with the proposed method. Here, our third order model can not model the motion appropriately.

the results by [4] which is, as ours, a method for multi-label motion segmentation. The proposed approach improves significantly over those results. Yet, note that dedicated video object segmentation approaches such as recently proposed in [68], [69] yield higher numbers on the DAVIS benchmark with mean Jaccard index of up to 80% on the validation set.

The evaluation of our model on the motion subtask of VSB100 is given in Fig. 12 in terms of boundary precision and recall (BPR) and the region metric volume precision and recall (VPR). It can be seen that the proposed higher order model with adaptive edge order outperforms the previous models on this task. As expected, the differences in the BPR are rather small while they are more significant in VPR. The dashed lines indicate results of our model using FlowNet [95] to compute optical flow while the solid lines are based on [88] to ensure fair comparison to [4] and [31]. The improved optical flow has a slightly larger impact on the BPR values, indicating that the issues addressed by more robust optical flow estimation and the issues addressed by our more complex motion model are complementary.

Scalability Analysis on FBMS-59 Last, we want to evaluate our proposed heuristic for the higher-order minimum cost lifted multicut problems (compare Alg. 1) in terms of computation times. Figure 13 plots the computation times of our full pipeline on FBMS-59 w.r.t. the number of point trajectories. The runtime distribution indicates linear runtime behavior and shows that heuristic solutions can be generated in a few minutes for most instances. Yet, the number of large problem instances is too small to make any claim.

7 Conclusion

We presented a multicut-based approach that can be applied to computer vision tasks such as motion segmentation and geometric model fitting. To do so, we proposed a pseudo-boolean formulation that allows to define costs on subsets of vertices of arbitrary cardinality and includes lifted edges. In motion segmentation higher than simple pair-wise costs allow to model object motion more precisely (Euclidean instead of in-plane translational motion). Line fitting can be formulated only with 3-rd order costs, while homography estimation requires 5-th order costs. Since the emerging higher-order multicut problem is NP-hard to solve exactly, we proposed an efficient local search algorithm for inference. When applied to real and toy problems, our approach yields either competitive or state-of-the-art results, is highly flexible and easy to apply.

References

[1] B. Andres, J. H. Kappes, T. Beier, U. Köthe, and F. A. Hamprecht, "Probabilistic image segmentation with closedness constraints," in *ICCV*, 2011.
[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.
[3] M. Keuper, E. Levinkov, N. Bonneel, G. Lavoué, T. Brox, and B. Andres, "Efficient decomposition of image and mesh graphs by lifted multicuts," in *ICCV*, 2015.
[4] M. Keuper, B. Andres, and T. Brox, "Motion trajectory segmentation via minimum cost multicuts," in *ICCV*, 2015.
[5] S. Tang, B. Andres, M. Andriluka, and B. Schiele, "Subgraph decomposition for multi-object tracking," in *CVPR*, 2015.
[6] S. Tang, M. Andriluka, B. Andres, and B. Schiele, "Multiple people tracking by lifted multicut and person reidentification," in *CVPR*, 2017, pp. 3539–3548.
[7] L. Pishchulin, E. Insafutdinov, S. Tang, B. Andres, M. Andriluka, P. Gehler, and B. Schiele, "DeepCut: Joint subset partition and labeling for multi person pose estimation," in *CVPR*, 2016.
[8] E. Insafutdinov, L. Pishchulin, B. Andres, M. Andriluka, and B. Schiele, "DeeperCut: A deeper, stronger, and faster multi-person pose estimation model," in *ECCV*, 2016.
[9] E. Insafutdinov, M. Andriluka, L. Pishchulin, S. Tang, E. Levinkov, B. Andres, and B. Schiele, "ArtTrack: Articulated multi-person tracking in the wild," in *CVPR*, 2017.
[10] A. Kirillov, E. Levinkov, B. Andres, B. Savchynskyy, and C. Rother, "Instancecut: from edges to instances with multicut," in *CVPR*, 2017.
[11] E. Levinkov, J. Uhrig, S. Tang, M. Omran, E. Insafutdinov, A. Kirillov, C. Rother, T. Brox, B. Schiele, and B. Andres, "Joint graph decomposition and node labeling: Problem, algorithms, applications," in *CVPR*, 2017.
[12] A. Kardoost, K. Ho, P. Ochs, and M. Keuper, "Self-supervised sparse to dense motion segmentation," in *ACCV*, 2020.
[13] E. D. Demaine, D. Emanuel, A. Fiat, and N. Immorlica, "Correlation clustering in general weighted graphs," *Theoretical Computer Science*, 2006.
[14] N. Bansal, A. Blum, and S. Chawla, "Correlation clustering," *Machine Learning*, 2004.
[15] S. Kim, S. Nowozin, P. Kohli, and C. D. Y., "Higher-order correlation clustering for image segmentation," in *NeurIPS*, 2011.

TABLE 7
Evaluation on DAVIS$_{16}$. The more complex motion model in Lifted AOMC is beneficial on this dataset of binary object segmentation. Results marked with * are taken from [59].

	MCe [4]			Lifted AOCM		
	train	val	trainval	train	val	trainval
J mean [↑]	53.4	55.2* /54.59	53.9	**62.7**	**57.79**	**60.74**
J recall [↑]	59.5	57.5* /58.41	59.04	**74.18**	**64.72**	**70.40**
J decay [↓]	**-1.57**	**2.2*** / 4	**0.66**	3.8	3.75	3.78
F mean [↑]	51.86	55.2* /52.35	52.05	**61.55**	**57.55**	**59.94**
F recall [↑]	56.39	61.0* /54.60	55.67	**72.36**	**67.63**	**70.47**
F decay [↓]	**2.5**	**3.4*** / 4.25	**3.23**	7.05	5.39	6.38

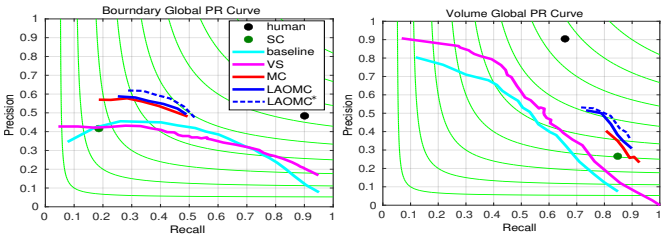


Fig. 12. Evaluation on the motion subtask of the VSB100 dataset [97], [98]. We compare our results to SC [31], the video segmentation approach VS [99], the superpixel tracking baseline from [97], and the multicut models with pairwise terms MCe [4]. The proposed lifted adaptive order model (LAOMC) outperforms the pairwise terms consistenly. LAOMC* shows results based on FlowNet [95], while LAOMC is computed on flows from [88] for fair comparison to [31].

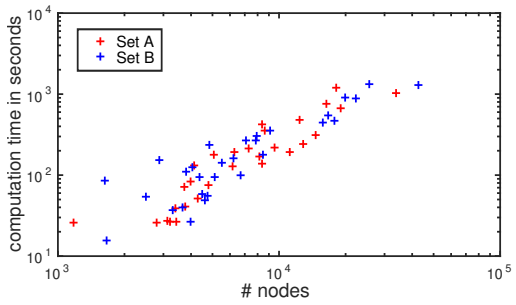


Fig. 13. Computation times in log-scale of the problem instances from Set A and B of FBMS-59 with respect to the number of point trajectories.

[16] J. H. Kappes, M. Speth, G. Reinelt, and C. Schnörr, "Higher-order segmentation via multicuts," *CVIU*, 2016.
[17] E. Levinkov, A. Kirillov, and B. Andres, "A comparative study of local search algorithms for correlation clustering," in *GCPR*, 2017.
[18] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, 1981.
[19] M. Zuliani, C. S. Kenney, and B. S. Manjunath, "The multiransac algorithm and its application to detect planar homographies," in *ICIP*, 2005.
[20] L. Magri and A. Fusiello, "T-linkage: A continuous relaxation of j-linkage for multi-model fitting," in *CVPR*, 2014.
[21] R. Toldo and A. Fusiello, "Robust multiple structures estimation with J-Linkage," in *ECCV*, 2008.
[22] R. B. Tennakoon, A. Bab-Hadiashar, Z. Cao, R. Hoseinnezhad, and D. Suter, "Robust model fitting using higher than minimal subset sampling," *TPAMI*, 2016.
[23] G. Xiao, H. Wang, T. Lai, and D. Suter, "Hypergraph modelling for geometric model fitting," *Pattern Recognit.*, 2016.
[24] P. Purkait, T. J. Chin, A. Sadri, and D. Suter, "Clustering with hypergraphs: The case for large hyperedges," *TPAMI*, 2017.
[25] T. Brox and J. Malik, "Object segmentation by long term analysis of point trajectories," in *ECCV*, 2010.
[26] 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.
[27] P. Ochs and T. Brox, "Higher order motion models and spectral clustering," in *CVPR*, 2012.
[28] Z. Li, J. Guo, L. Cheong, and S. Zhou, "Perspective motion segmentation via collaborative clustering," in *ICCV*, 2013.
[29] F. Shi, Z. Zhou, J. Xiao, and W. Wu, "Robust trajectory clustering for motion segmentation," in *ICCV*, 2013.
[30] K. Koffka, *Principles of Gestalt Psychology*, 1935.
[31] P. Ochs, J. Malik, and T. Brox, "Segmentation of moving objects by long term video analysis," *TPAMI*, 2014.
[32] T. Tanimoto, "Technical report," *IBM Internal Report*, 1957.
[33] L. Magri and A. Fusiello, "Multiple models fitting as a set coverage problem," in *CVPR*, 2016.
[34] ——, "Robust multiple model fitting with preference analysis and low-rank approximation," in *BMVC*, 2015.
[35] M. Tepper and G. Sapiro, "Nonnegative matrix underapproximation for robust multiple model fitting," in *CVPR*, 2017.
[36] M. Denitto, L. Magri, A. Farinelli, A. Fusiello, and M. Bicego, "Multiple structure recovery via probabilistic biclustering," in *S+SSPR (workshop)*, 2016.
[37] S. Hochreiter, U. Bodenhofer, M. Heusel, A. Mayr, A. Mitterecker, A. Kasim, T. Khamiakova, S. Van Sanden, D. Lin, W. Talloen, L. Bijnens, H. W. H. Göhlmann, Z. Shkedy, and D.-A. Clevert, "Fabia: factor analysis for bicluster acquisition," *Bioinform.*, 2010.
[38] H. Isack and Y. Boykov, "Energy-based geometric multi-model fitting," *IJCV*, 2012.
[39] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *TPAMI*, 2001.
[40] P. Amayo, P. Piniés, L. M. Paz, and P. Newman, "Geometric multi-model fitting with a convex relaxation algorithm," in *CVPR*, 2018.
[41] D. Barath and J. Matas, "Multi-class model fitting by energy minimization and mode-seeking," in *ECCV*, 2018.
[42] D. Comaniciu and P. Meer, "Mean shift: a robust approach toward feature space analysis," *TPAMI*, 2002.
[43] S. Agarwalal, J. Lim, L. Zelnik-Manor, P. Perona, D. Kriegman, and S. Belongie., "Beyond pairwise clustering," in *CVPR*, 2005.
[44] S. Jain and V. Madhav Govindu, "Efficient higher-order clustering on the grassmann manifold," in *ICCV*, 2013.
[45] V. Madhav Govindu, "A tensor decomposition for geometric grouping and segmentation." in *CVPR*, 2005.
[46] H. Wang, X. Guobao, Y. Yan, and D. Suter, "Searching for representative modes on hypergraphs for robust geometric model fitting," *TPAMI*, 2018.
[47] S. Agarwal, K. Branson, and S. Belongie, "Higher order learning with graphs," in *ICML*, 2006.
[48] D. Zhou, J. Huang, and B. Schölkopf, "Learning with hypergraphs: clustering, classification, and embedding," in *NeurIPS*, 2007.
[49] G. Chen and G. Lerman, "Spectral curvature clustering scc," in *ICCV*, 2009.
[50] 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.
[51] E. Elhamifar and R. Vidal, "Sparse subspace clustering," in *ICCV*, 2013.
[52] T. Wu, A. R. Benson, and D. F. Gleich, "General tensor spectral co-clustering for higher-order data," in *NeurIPS*, 2016.
[53] A. Fix, A. Gruber, and B. Boros, "A graph cut algorithm for higher-order markov random fields," in *ICCV*, 2011.
[54] X. Lan, S. Roth, D. Huttenlocher, and M. J. Black, "Efficient belief propagation with learned higher-order markov random fields," in *ECCV*, 2006.
[55] K. Schelten and S. Roth, "Mean field for continuous high-order mrfs," in *DAGM*, 2012.
[56] J. Liu, J. Jinglu Wang, T. Fang, C.-L. Tai, and L. Quan, "Higher-order crf structural segmentation of 3d reconstructed surfaces," in *ICCV*, 2015.
[57] P. Ji, H. Li, M. Salzmann, and Y. Dai, "Robust motion segmentation with unknown correspondences," in *ECCV*, 2014.
[58] P. Ochs and T. Brox, "Object segmentation in video: a hierarchical variational approach for turning point trajectories into dense regions," in *ICCV*, 2011.
[59] P. Tokmakov, K. Alahari, and C. Schmid, "Learning motion patterns in videos," in *CVPR*, 2017.

[60] ——, "Learning video object segmentation with visual memory," in *ICCV*, 2017.
[61] L. Maczyta, P. Bouthemy, and O. Meur, "Cnn-based temporal detection of motion saliency in videos," *Patt. Recognit. Lett.*, 2019.
[62] K. Fragkiadaki, P. Arbelaez, P. Felsen, and J. Malik, "Learning to segment moving objects in videos," in *CVPR*, 2015.
[63] M. Siam, H. Mahgoub, M. Zahran, S. Yogamani, M. Jagersand, and A. El-Sallab, "Modnet: Motion and appearance based moving object detection network for autonomous driving," in *ITSC*, 2018.
[64] S. Jain, B. Xiong, and K. Grauman, "Fusionseg: Learning to combine motion and appearance for fully automatic segmentation of generic objects in videos," *arXiv preprint arXiv:1701.05384*, 2017.
[65] P. Tokmakov, C. Schmid, and A. Karteek, "Learning to segment moving objects," *IJCV*, 2018.
[66] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," in *ICCV*, 2015.
[67] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *IJCV*, 2015.
[68] Y. Yang, B. Lai, and S. Soatto, "Dystab: Unsupervised object segmentation via dynamic-static bootstrapping," in *CVPR*, 2021.
[69] C. Yang, H. Lamdouar, E. Lu, A. Zisserman, and W. Xie, "Self-supervised video object segmentation by motion grouping," in *ICCV*, 2021.
[70] G. Yang and D. Ramanan, "Learning to segment rigid motions from two frames," in *CVPR*, 2021.
[71] P. Bideau and E. Learned-Miller, "Its moving! a probabilistic model for causal motion segmentation in moving camera videos," 10 2016, pp. 433–449.
[72] P. Bideau, R. R. Menon, and E. Learned-Miller, "Moa-net: Self-supervised motion segmentation," in *ECCV workshop*, 2018.
[73] P. Bideau, A. RoyChowdhury, R. R. Menon, and E. Learned-Miller, "The best of both worlds: Combining cnns and geometric constraints for hierarchical motion segmentation," in *CVPR*, 2018.
[74] M. Irani and P. Anandan, "A unified approach to moving object detection in 2d and 3d scenes," *TPAMI*, 1998.
[75] J. H. Kappes, M. Speth, B. Andres, G. Reinelt, and C. Schnörr, "Globally optimal image partitioning by multicuts," in *EMMCVPR*, 2011.
[76] S. Tang, B. Andres, M. Andriluka, and B. Schiele, "Multi-person tracking by multicut and deep matching," in *ECCVW*, 2016.
[77] S. Kim, C. Yoo, S. Nowozin, and P. Kohli, "Image segmentation using higher-order correlation clustering," *TPAMI*, 2014.
[78] T. Beier, B. Andres, U. Köthe, and F. A. Hamprecht, "An efficient fusion move algorithm for the minimum cost lifted multicut problem," in *ECCV*, 2016.
[79] A. Kardoost and M. Keuper, "Solving minimum cost lifted multicut problems by node agglomeration," in *ACCV*, 2019.
[80] S. Chopra and M. Rao, "The partition problem," *Mathematical Programming*, 1993.
[81] A. Horňáková, J.-H. Lange, and B. Andres, "Analysis and optimization of graph decompositions by lifted multicuts," in *ICML*, 2017.
[82] J. Yarkony, A. Ihler, and C. Fowlkes, "Fast planar correlation clustering for image segmentation," in *ECCV*, 2012.
[83] J. Yarkony, "Analyzing PlanarCC: Demonstrating the equivalence of PlanarCC and the multi-cut LP relaxation," in *NeurIPS Workshop on Discrete Optimization*, 2014.
[84] J. H. Kappes, B. Andres, F. A. Hamprecht, C. Schnörr, S. Nowozin, D. Batra, S. Kim, B. X. Kausler, T. Kröger, J. Lellmann, N. Komodakis, B. Savchynskyy, and C. Rother, "A comparative study of modern inference techniques for structured discrete energy minimization problems," *IJCV*, 2015.
[85] Gurobi Optimization, LLC, "Gurobi Optimizer Reference Manual," 2021.
[86] K. Fragkiadaki, P. Arbelaez, P. Felsen, and J. Malik, "Learning to segment moving objects in videos," in *CVPR*, 2015.
[87] K. Fragkiadaki, G. Zhang, and J. Shi, "Video segmentation by tracing discontinuities in a trajectory embedding," in *CVPR*, 2012.
[88] T. Brox and J. Malik, "Large displacement optical flow: descriptor matching in variational motion estimation," *TPAMI*, 2011.
[89] W. Zhang and J. Kosecka, "Nonparametric estimation of multiple structures with outliers," in *ECCV workshop*, 2006.
[90] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, 1955.
[91] H. S. Wong, T.-J. Chin, J. Yu, and D. Suter, "Dynamic and hierarchical multi-structure geometric model fitting," in *ICCV*, 2011.
[92] T. T. Pham, T.-J. Chin, J. Yu, and D. Suter, "The random cluster model for robust geometric fitting," *TPAMI*, 2014.
[93] P. Bideau and E. Learned-Miller, "A detailed rubric for motion segmentation," 2016.
[94] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *CVPR*, 2017.
[95] E. Ilg, T. Saikia, M. Keuper, and T. Brox, "Occlusions, motion and depth boundaries with a generic network for disparity, optical flow or scene flow estimation," in *ECCV*, 2018.
[96] B. Taylor, V. Karasev, and S. Soattoc, "Causal video object segmentation from persistence of occlusions," in *CVPR*, 2015.
[97] F. Galasso, N. Nagaraja, T. Cardenas, T. Brox, and B.Schiele, "A unified video segmentation benchmark: Annotation, metrics and analysis," in *ICCV*, 2013.
[98] P. Sundberg, T. Brox, M. Maire, P. Arbelaez, and J. Malik, "Occlusion boundary detection and figure/ground assignment from optical flow," in *CVPR*, 2011.
[99] F. Galasso, M. Keuper, T. Brox, and B. Schiele, "Spectral graph reduction for efficient image and streaming video segmentation," in *CVPR*, 2014.
[100] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung, "A benchmark dataset and evaluation methodology for video object segmentation," in *CVPR*, 2016.

Evgeny Levinkov is a research scientist in the Bosch Center for Artificial Intelligence, Germany. He received MSc and PhD degrees from the Saarland University, Germany. During his PhD studies he was with the Max Planck Institute for Informatics, Germany.

Amirhossein Kardoost received the MSc degree from the Saarland University, Germany, in 2017. Since 2017, he is a PhD candidate for Computer Vision in University of Mannheim, Germany. His research is in video object and motion segmentation mostly under the minimum cost lifted multicut formulation. He is supervised by Margret Keuper.

Bjoern Andres is the Professor of Machine Learning for Computer Vision at TU Dresden. Previously, he worked as a group leader and department head at the Bosch Center for Artificial Intelligence, as a Senior Researcher at the Max Planck Institute for Informatics and as a Postdoctoral Fellow at Harvard University. His doctorate in Physics at the University of Heidelberg was supervised by Fred A. Hamprecht.

Margret Keuper is a Professor for Visual Computing at the University of Siegen, Germany. Before, she was a Juniorprofessor at the University of Mannheim and she worked as a postdoctoral researcher for the University of Freiburg and at the Max Planck Institute for Informatics in Saarbruecken. She did her Ph.D. under the supervision of Thomas Brox at the University of Freiburg.