Motion Segmentation via Synchronization

Federica Arrigoni and Tomas Pajdla
CIIRC – Czech Technical University in Prague
Federica.Arrigoni@cvut.cz, pajdla@cvut.cz

Abstract

In this paper we consider the problem of segmenting points in a collection of images that contain multiple moving objects. Our contribution is three-fold: (i) we propose a matrix representation of segmentation that permits to formulate the problem in terms of "synchronization" of binary matrices; (ii) we derive a spectral solution to solve such a problem, which is inspired by previous works on synchronization of rotations, homographies, rigid motions and permutations; (iii) we explain how our solution can be interpreted in terms of spectral clustering. The proposed approach is validated on both synthetic and real scenarios, in addition to the Hopkins benchmark.

1. Introduction

The synchronization problem is a well studied task in Computer Vision arising in a variety of applications. The term originates from *time synchronization* [17, 11] where the goal is to synchronize clocks in a network by measuring time differences between pairs of clocks. Other names include "averaging" and "graph optimization". In general, the task of synchronization is to recover elements of a group by measuring ratios between pairs of elements. Such group can represent the set of rotations [45, 10, 13, 51, 6] or the set of rigid motions [12, 49, 5, 3, 38], which find application in structure from motion, registration of 3D point sets, and simultaneous localization and mapping. Other examples include homographies [42, 40] and permutations [35, 63, 43], which are related to image mosaicking and multi-image matching, respectively.

There are many techniques available to address synchronization. Among them, the method developed in [45] – where the problem is cast as a spectral decomposition – is particularly interesting since it can be applied to any group admitting a matrix representation. Originally developed for rotations in 2-space, the same technique was extended to rotations in 3-space [46, 1], rigid motions [5, 3], homographies [42] and permutations [35]. It has been recently shown that the spectral solution can also be derived in situations where the group structure is missing, such as the case of *partial* permutations [2, 30], which form a semigroup and model missing correspondences in multi-image matching.

In this paper we extend the spectral solution to the case of binary (i.e. 0-1) matrices, which contain the set of partial permutations, but they have a poorer structure as multiplication of two binary matrices is not necessarily a 0-1 matrix. Synchronization of binary matrices finds application in *motion segmentation*, that is the problem of clustering points in multiple images according to a number of moving objects.

The paper is organized as follows. In Sec. 1.1 we review the previous work on motion segmentation. In Sec. 2 we define a matrix representation of motion segmentation, that permits to formulate the problem in terms of synchronization. In Sec. 3 we derive a spectral solution to solve it, which builds upon previous works [45, 46, 1, 5, 3, 42, 35, 2, 30]. We also show that our solution can be interpreted in terms of *spectral clustering* [58]. Besides providing an interesting insight, as the graph clustering literature is connected with the synchronization theory, this observation permits to develop some variants of our method. Experiments on both synthetic and real data are reported in Sec. 4, where the advantages and limitations of the proposed approach are discussed.

1.1. Related work

Motion segmentation is the problem of clustering point trajectories over a sequence of images according to the different motions they belong to. It is an essential task in several applications in Computer Vision, including surveillance, action recognition, scene understanding and autonomous driving. Motion segmentation can be cast as a *subspace clustering* problem since – under the affine camera model – the point trajectories lie in the union of d subspaces in $\mathbb{R}^{2n}$ of dimension at most 4, where d denotes the number of motions and n denotes the number of images. Existing solutions include Generalized Principal Component Analysis (GPCA) [54], Local Subspace Affinity (LSA) [62], Power Factorization (PF) [56], Agglomerative Lossy Compression (ALC) [37], Low-Rank Representation (LRR) [25], Sparse Subspace Clustering (SSC) [9],

Structured Sparse Subspace Clustering (S^3C) [21], and Robust Shape Interaction Matrix (RSIM) [16].

Motion segmentation can also be expressed in terms of *multiple model fitting* – under the affine camera model – by fitting multiple subspaces to feature trajectories in an image sequence. Existing solutions include consensus-based approaches (e.g. the Hough transform [60], Sequential RANSAC [57], Multi-RANSAC [66] and Random Sample Coverage [29]), preference-based approaches (e.g. Residual Histogram Analysis [65], J-Linkage [47], T-linkage [27], Random Cluster Model [36] and Robust Preference Analysis [28]), and energy minimization (e.g. PEARL [15] and Multi-X [4]). Model fitting techniques can also be used to solve motion segmentation in two images under the perspective camera model, by fitting multiple fundamental matrices to corresponding points in an image pair.

Other solutions to motion segmentation include [23, 20, 61]. The authors of [23] formulate a joint optimization problem which builds upon the SSC algorithm, where it is required that all image pairs share a common sparsity profile. In [20] an accumulated correlation matrix is built by sampling homographies over consecutive image pairs, and spectral clustering [58] is applied to get the sought segmentation. Such approach is generalized in [61] where multiple models (affine, fundamental and homography) are combined to get an improved segmentation. Different approaches are analyzed to reach such task, namely Kernel Addition (KerAdd) [7], Co-Regularization (Coreg) [19] and Subset Constrained Clustering (Subset) [59]. Motion segmentation is also a sub-task of *multibody structure from motion*, that is a generalization of structure from motion [34] to the dynamic case [41], where motion segmentation has to be solved in addition to 3D reconstruction. Available approaches include geometric solutions [55, 53], statistical techniques [48, 33, 39] and factorization methods [8, 22, 64].

Our solution to motion segmentation differs from the literature as it does not assume point trajectories over multiple images, but it only requires matches between pairs of images. We will clarify this later.

2. Problem Formulation

Here we precisely formulate the problem we are solving. Let n denote the number of images, let d denote the number of motions and let p_i denote the number of points in image i. The task is to cluster image points according to d motions. We observe that the classification of points in two images i and j can be represented as a matrix $S_{ij} \in \{0,1\}^{p_i \times p_j}$ constructed as follows:

- $[S_{ij}]_{h,k} = 1$ if point h in image i belongs to the same motion as point k in image j;
- $[S_{ij}]_{h,k} = 0$ otherwise.

The binary matrix S_{ij} is referred to as the *partial segmentation* of the pair (i, j). It is a *local* representation of segmentation since it says which points in two different images belong to the same motion but it does not say which motion it is.

Similarly, the classification of points in image i can be represented as a matrix $S_i \in \{0,1\}^{p_i \times d}$ constructed as follows:

- $[S_i]_{h,k} = 1$ if point h in image i belongs to motion k;
- $[S_i]_{h,k} = 0$ otherwise.

The binary matrix S_i is referred to as the *total segmentation* of image i. It is a *global* representation of segmentation since it reveals the membership of points with respect to an absolute numbering of motions. Note that we can interpret S_i as the partial segmentation between image i and a reference image called the "universe" which contains one point for each motion. The concepts of total and partial segmentations are illustrated in Fig. 1.

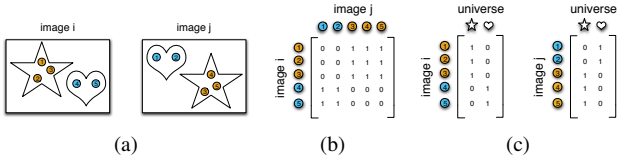

Figure 1: Sub-figure (a) represents the segmentation of points in two images i and j, where a label (yellow or blue) is assigned to each point based on the moving object (star or heart) it belongs to. Sub-figure (b) reports the binary matrix representing the partial segmentation of the pair (i, j). Sub-figure (c) reports the binary matrices representing the total segmentations of images i and j.

It can be checked that

$$S_{ij} = S_i S_j^{\mathsf{T}}. \tag{1}$$

Equation (1) is called the *consistency constraint*. In simple words, it states that the partial segmentation of the pair (i, j) can be obtained by composing the segmentation between image i and the universe and the segmentation between the universe and image j. Thus motion segmentation can be solved in two steps:

1. compute the partial segmentations S_{ij} for $(i, j) \in \{1, \ldots, n\} \times \{1, \ldots, n\}$;
2. compute the total segmentations $S_1, \ldots, S_n$ such that Eq. (1) is best satisfied.

Concerning Step 1, any algorithm able to perform two-frame segmentation can be used, such as Robust Preference Analysis (RPA) [28]. Note that tracks over multiple images are not required and the actual input is a set of correspondences between image pairs. Indeed, only pairwise

matches are needed in order to perform two-frame segmentation. Step 2 can be recognized as a *synchronization* problem [45, 35], where the output matrices $S_1, \dots, S_n$ represent labels of image points. The solution is not unique as $S_{ij} = S_i S_j^\mathsf{T} = (S_i Q)(S_j Q)^\mathsf{T}$ for any $d \times d$ permutation matrix Q, which corresponds to a different numbering of motions. Note that the coordinates of points are used in Step 1 only and they are not used in Step 2.

In practice, the input partial segmentations may contain some errors, which can be either caused by mismatches or by failure of the algorithm used for two-frame segmentation. Thus the task is to compute the total segmentations such that these errors get compensated by exploiting redundant measures.

3. Proposed Method

In this section we show how to address Step 2 via spectral decomposition. Our approach can be viewed as the extension of previous works on synchronization of rotations [45, 46, 1], rigid motions [5, 3], homographies [42] and permutations [35, 2, 30] to binary matrices.

3.1. The exact case

Let $p = \sum_{i=1}^n p_i$ denote the total amount of points over all the images, and let us collect all the total and partial segmentations in two block-matrices $X \in \{0,1\}^{p \times d}$ and $Z \in \{0,1\}^{p \times p}$ constructed as follows

$$X = \begin{bmatrix} S_1 \\ S_2 \\ \dots \\ S_n \end{bmatrix}, \quad Z = \begin{bmatrix} S_{11} & S_{12} & \dots & S_{1n} \\ S_{21} & S_{22} & \dots & S_{2n} \\ \dots & & & \dots \\ S_{n1} & S_{n2} & \dots & S_{nn} \end{bmatrix}. \tag{2}$$

Using this notation, Eq. (1) becomes

$$Z = XX^\mathsf{T} \tag{3}$$

which implies that Z is symmetric positive semidefinite and it has rank d.

Proposition 1. *The columns of X are d (orthogonal) eigenvectors of Z.*

Proof. Note that $S_i^\mathsf{T} S_i$ is a $d \times d$ diagonal matrix such that the (k,k)-entry counts the number of points in image i that belong to motion k. Thus $X^\mathsf{T} X = \sum_{i=1}^n (S_i^\mathsf{T} S_i)$ is a diagonal matrix such that the (k,k)-entry counts the number of points over all the images that belong to motion k. Combining this observation with Eq. (3) we get

$$ZX = XX^\mathsf{T} X = X \underbrace{\sum_{i=1}^n (S_i^\mathsf{T} S_i)}_{\Lambda} = X \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_d \end{bmatrix} \tag{4}$$

which is a spectral decomposition, i.e., the columns of X are d eigenvectors of Z and the corresponding eigenvalues are contained in the diagonal of Λ. Note that $\lambda_1, \dots, \lambda_d$ are the largest eigenvalues of Z, since Z has rank d, and all the other eigenvalues are zero. □

We assume here that each motion contains a *different* number of points, i.e., all the nonzero eigenvalues of Z are distinct. This implies that – in the absence of noise – the block-matrix X (and hence the unknown total segmentations) can be uniquely recovered from the leading eigenvectors of Z. The size of Z may be large in practical scenarios. However, note that such a matrix is sparse, being composed of binary matrices, so sparse solvers can be exploited (e.g. the Matlab command `eigs`).

3.2. The noisy case

In the presence of noise, Eq. (3) will not be satisfied in general, so Z will not have exactly d nonzero eigenvalues. However, Prop. 1 suggests that the eigenvectors of Z corresponding to the d *largest* eigenvalues can be viewed as an estimate of X. First, we explain the meaning of this procedure in terms of an optimization task. In particular, we show that the leading eigenvectors solve a *relaxed* version of a reasonable maximization problem. Then, we derive an heuristic to obtain the sought total segmentations from the leading eigenvectors. Such rounding step is required since the eigenvectors are an approximate solution that is not guaranteed to have binary entries.

Let us consider the following problem[1]

$$\begin{gathered} \max_{S_1,\dots,S_n} \sum_{i,j=1}^n \text{trace}(S_{ij}^\mathsf{T} S_i S_j^\mathsf{T}) \\ \text{s.t. } S_i \in \{0,1\}^{p_i \times d},\ S_i \mathbf{1} = \mathbf{1} \quad \forall\, i = 1, \dots, n \end{gathered} \tag{5}$$

where $\mathbf{1}$ denotes a vector of ones (of appropriate dimensions) and the constraint $S_i \mathbf{1} = \mathbf{1}$ means that each point must belong to (exactly) one motion. We also require each motion to be non-empty. The cost function in (5) counts, for each image pair, the number of points equally labelled by the partial segmentations S_{ij} and $S_i S_j^\mathsf{T}$. It can also be expressed as $\sum_{i,j=1}^n \text{trace}(S_i^\mathsf{T} S_{ij} S_j)$ by using basic properties of the trace operator. Thus Eq. (5) can be rewritten in matrix form as

$$\max_X \text{trace}(X^\mathsf{T} Z X) \quad \text{s.t. } X \in \{0,1\}^{p \times d},\ X\mathbf{1} = \mathbf{1}. \tag{6}$$

[1] In real scenarios we might not be able to compute the partial segmentation of all the image pairs, due to missing correspondences. In such a situation S_{ij} is set to zero, resulting in a zero block in Z, so that the cost function in (5) counts the contributions coming from the available partial segmentations only. In particular, note that we can not compute S_{ii} for $i = 1, \dots, n$ since it is equivalent to the knowledge of the unknown S_i, thus the diagonal of Z will be filled with zero blocks in practice.

Note that the columns of X are *orthogonal*, since each point belongs to one motion only.

Solving (6) is a difficult task since the optimization variable is constrained to be a binary matrix. In order to make the computation tractable, we relax the constraints and consider the following problem

$$\max_{U} \text{ trace}(U^\mathsf{T} Z U) \quad \text{s.t. } U \in \mathbb{R}^{p \times d},\ U^\mathsf{T} U = I_d \qquad (7)$$

where the optimization variable is treated as a real matrix instead of a binary matrix, and its columns are enforced to be orthonormal. The notation I_d represents the $d \times d$ identity matrix, and the notation U (instead of X) is used to underline that, due to the relaxation, the optimal U will not be a binary matrix, in general. The constraint $U^\mathsf{T} U = I_d$, besides enforcing the columns of U to be orthogonal, also constraints each column to have the unit norm, thus avoiding the trivial solution where all the points belong to the same motion. Equation (7) is a generalized Rayleigh problem, whose solution is given by the d leading eigenvectors of Z. Such eigenvectors are then scaled by the square root of the corresponding eigenvalues, in order to ensure that the sum of nonzero entries in each column in U is (approximately) equal to the number of points in the corresponding motion.

We now explain how to turn U into a binary matrix representing the sought total segmentations. Recall that – in the absence of noise – each row of U, which corresponds to an image point, contains exactly one entry equal to 1, which corresponds to the motion such point belongs to, and all other entries are zero. The presence of noise cripples the structure of Z, so that U will not have entries in $\{0, 1\}$ in general. However, we expect that, for each row in U, the entry that reveals the membership to a specific motion is close to one and all other entries are close to zero. Thus a reasonable approach is to construct the output matrix $X \in \{0, 1\}^{p \times d}$ as follows:

- $[X]_{h,k} = 1$ if $[U]_{h,k}$ is the maximum value over row h of U;
- $[X]_{h,k} = 0$ otherwise.

This procedure can be regarded as a "projection" onto the feasible set.

Dealing with mismatches. In the presence of gross errors, the structure of U may not be so evident. To handle this situation, we propose to label only those points for which we are sure about the class. Specifically, for each row in U, which corresponds to an image point, we compute the ratio between the largest entry and the second-largest entry: if this ratio – which should be infinite in the absence of noise – is larger than a threshold θ, then the point is classified as explained above; otherwise, the corresponding row in X is set to zero, meaning that the point is labelled as "unclassified" or "unknown". Thus the projection procedure is modified as follows:

- $[X]_{h,k} = 1$ if the following conditions are satisfied:
  - $[U]_{h,k}$ is the maximum value over row h;
  - $[U]_{h,k} \neq 0$;
  - $[U]_{h,k} / [U]_{h,l} > \theta$ where $[U]_{h,l}$ is the second-maximum value over row h;
- $[X]_{h,k} = 0$ otherwise.

The condition $[U]_{h,k} \neq 0$ is introduced to handle zero rows in U. Indeed, due to the presence of mismatches, the algorithm used for two-frame segmentation may classify some matches as outliers, i.e., the corresponding points are not assigned to a motion, resulting in zero rows/columns in some partial segmentations. In particular, it may happen that a point is labelled as outlier in all the image pairs, resulting in a zero row in Z (and hence in U). In such a situation the point is not assigned to a motion, since there is no valid information to classify it. In order to handle rows that are nearly (but not exactly) zero, we set all the entries in U that are smaller than a threshold τ to zero before applying the projection procedure. The resulting method is named SYNCH.

3.3. Spectral Clustering interpretation

We now show that SYNCH can be interpreted in terms of *spectral clustering* [58]. This interesting observation, besides linking synchronization with graph clustering, allows us to develop some variants of our approach. In general, the task of graph clustering is to divide data points – represented as a graph – into several groups. The idea behind spectral clustering techniques is to consider a different representation of points – based on the eigenvectors of specific matrices associated with the graph – so that clusters can be trivially extracted in the new representation.

The key observation is that the matrix Z defined in (2) can be viewed as the *adjacency matrix* of a graph constructed as follows:

- each node corresponds to a point in an image;
- an edge is present between two nodes if and only if the corresponding points belong to the same motion.

In a nutshell, SYNCH first performs the spectral decomposition of Z and then it applies a projection procedure in order to get a binary matrix representing the total segmentations. So it can be viewed as a sort of spectral clustering, where image points are clustered based on the membership to a specific motion. In particular, the usage of the adjacency matrix for graph clustering is referred to as the *average association* in the literature (see [44]).

Typically, spectral clustering algorithms adopt *k-means* [14] as a post-processing step in order to get clusters from the eigenvector representation. This suggests an alternative projection procedure for our approach: after computing the top d eigenvectors of Z, which are collected in a matrix U, the total segmentations can be recovered by applying k-means to the rows of U. Specifically, in order to handle mismatches, $d+1$ clusters are computed and the cluster that is closest to the zero row is identified as the group of "unclassified" points. This choice is motivated by the fact that – in the presence of high corruption – points that are mismatched in all the image pairs should correspond to zero rows in Z (and hence in U), as already observed in Sec. 3.2. This variant of our method is named SYNCH-KMEANS.

The above analysis implies that any algorithm able to address spectral clustering (e.g. those reviewed in the tutorial [58]) can be used to compute the total segmentations. One of the most popular is the *normalized cuts* solution developed in [31], where the least eigenvectors of the *symmetric normalized Laplacian matrix* are computed. Such a matrix is defined as follows

$$L = I - D^{-1/2} Z D^{1/2} \tag{8}$$

where D denotes the *degree matrix* of the graph, that is a diagonal matrix whose (k,k)-entry is the sum of the k-th row of the adjacency matrix, namely $D = Z\mathbf{1}$. At the end $d+1$ clusters are identified with k-means, with one cluster representing unclassified points. This solution is named SYNCH-NCUTS.

Note that, although spectral clustering is widely used in segmentation literature, it has never been applied to the graph encoded by the matrix Z in Eq. (2) – which represents multiple two-frame segmentations – so SYNCH-KMEANS and SYNCH-NCUTS are indeed new approaches. In particular, note that our graph has p nodes, where p denotes the total amount of points over all the images. Existing techniques exploiting spectral clustering for motion segmentation (e.g. [23, 20, 61]), instead, consider a graph whose number of nodes is equal to the number of tracks, meaning that the task is to cluster tracks over multiple images.

4. Experiments

In this section we report our experimental results. First, we compared all the variants of our method[2] (i.e. SYNCH, SYNCH-KMEANS and SYNCH-NCUTS) on synthetic datasets based on the Hopkins155 benchmark [50] (Sec. 4.1). Then, we selected the best version and we compared it with the state of the art on both simulated and real scenarios (Sec. 4.2). In order to compute the partial segmentations – which constitute the input to our approach – we fitted multiple fundamental matrices to correspondences in each image pair using RPA [28], whose code is available online[3]. This technique extracts multiple models from outlier-contaminated data by combining principles of robust principal component analysis [24] and non-negative matrix factorization [18]. Concerning the parameters of SYNCH, we used $\tau = 0.01$ and $\theta = 1.5$ in all the experiments. As done by most papers in segmentation literature, we assumed that the number of motions was known a priori and we gave such value as input to all the competing methods.

4.1. Comparisons between our methods

The Hopkins155 benchmark [50] contains 155 sequences of indoor and outdoor scenes with two or three motions, that are categorized into checkerboard, traffic and articulated/nonrigid sequences. In order to study the robustness to mismatches of our approach, we considered three sequences from Hopkins155, namely *1R2RCR_g12*, *2RT3RTCRT* and *cars1*, whose properties are summarized in Tab. 1. First, noise-free pairwise matches were obtained from the available tracks. Then, synthetic errors were introduced by randomly switching a fraction of the correspondences (ranging from 0 to 0.8) in each image pair. All the results were averaged over 10 trials.

Table 1: The category of the scene, the number of motions d, the number of images n, and the total number of image points p are reported for three sequences from Hopkins155 [50].

Dataset	Category	d	n	p
1R2RCR_g12	*checkerboard*	2	24	3672
2RT3RTCRT	*checkerboard*	3	23	8211
cars1	*traffic*	2	20	6140

Since ground-truth segmentation is available, a quantitative evaluation can be provided. More precisely, performance was measured in terms of *misclassification error*, defined as the percentage of misclassified points over the total amount of classified image points – meaning that points labelled as unknown (if any) did not contribute to the error. We also reported the percentage of classified points.

Results obtained with all the variants of our approach are reported in Fig. 2. Recall that SYNCH-KMEANS and SYNCH-NCUTS are general-purpose algorithms for solving spectral clustering problems that we applied to the matrix Z in Eq. (2), whereas SYNCH exploits the specific structure of the matrices involved in our formulation of motion segmentation. Concerning *1R2RCR_g12* and *cars1*, there are no significative differences between all the analysed techniques, which can tolerate a high percentage of wrong correspondences. Among them, SYNCH classifies the highest amount of points. Concerning *2RT3RTCRT*, the improvement of SYNCH over SYNCH-KMEANS and SYNCH-

[2] https://github.com/federica-arrigoni/ICCV_19

[3] http://www.diegm.uniud.it/fusiello/demo/rpa/

NCUTS in terms of misclassification error is evident. Note that such dataset is more difficult than the others since it involves three motions. Therefore, we elect SYNCH as our choice and drop SYNCH-KMEANS and SYNCH-NCUTS in subsequent comparisons.

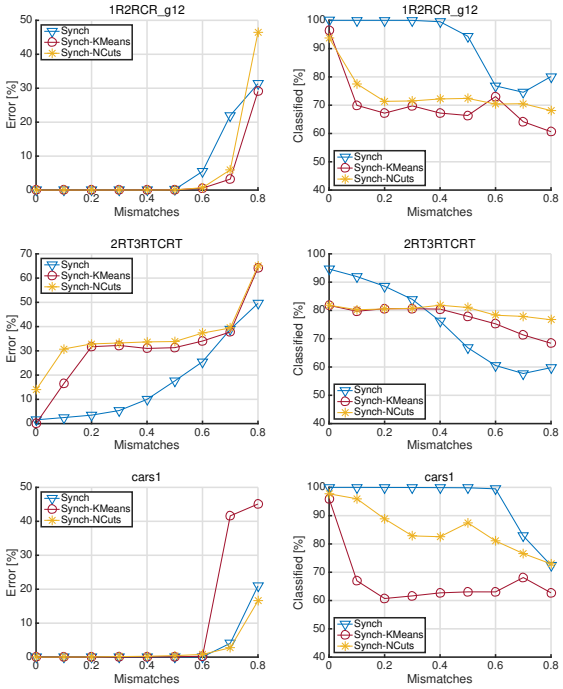

Figure 2: Misclassification error [%] and classified points [%] versus fraction of mismatches for some variants of our method on three sequences from [50].

4.2. Comparisons with the state of the art

In order to compare SYNCH with state-of-the-art techniques, we considered both synthetic data and real images, in addition to the Hopkins benchmark.

4.2.1 Hopkins benchmark

The Hopkins155 benchmark [50] (that is widely used in segmentation literature) and the Hopkins12 dataset [56] (that contains 12 additional sequences with missing data) provide noise-free tracks over multiple images with ground-truth labels. Existing works were typically evaluated in terms of misclassification error, defined as the percentage of misclassified tracks over the total amount of tracks. However, observe that our approach addresses a different task than the literature: it requires as input pairwise matches only and it provides a segmentation of image points, whereas existing techniques segment tracks. For this reason, we evaluated our approach in two different ways.

First of all, we computed the percentage of misclassified points over the total amount of classified image points – as done in Sec. 4.1 – and we also reported the percentage of classified points. This is the most natural way to evaluate the performances of SYNCH. Secondly, in order to make a meaningful comparison with the state of the art, we applied the following procedure to the output of our approach: we labelled each track with the most frequent value among the labels of all the points belonging to the track. We named the resulting method SYNCH-tracks. Thus the traditional misclassification error could be computed as the percentage of misclassified tracks, where tracks labelled as unknown (if any) were counted as errors.

Results are reported in Tables 2 and 3 where SYNCH and SYNCH-tracks are compared to several segmentation algorithms. The fact that our approach is not the best is not surprising since we are making more difficult assumptions (i.e., pairwise matches instead of tracks). However, both variants of our method present good performances: SYNCH achieves a mean error of 1.19% over all the sequences in Hopkins155 and a median error of 0.28% over all the sequences in Hopkins12, and it classifies a significant amount of points on both datasets; SYNCH-tracks is comparable to most existing techniques, with a mean error of 3.67% over all the sequences in Hopkins155 and a median error of 0.57% over all the sequences in Hopkins12. The fact that SYNCH-tracks is generally worse than SYNCH is due to the evaluation protocol that counts as errors all the unclassified tracks.

4.2.2 Synthetic data

We considered the synthetic dataset based on the *1R2RCR_g12* sequence [50] used in Sec. 4.1. We compared SYNCH with RSIM[4] [16] – which provides a robust solution to subspace clustering, and Subset[5] [61] – which can be considered the current state of the art in motion segmentation with mean error of 0.31% on the Hopkins155 benchmark (see Tab. 2). We exploited two different algorithms for computing tracks from pairwise matches, namely StableSfM[6] [32] and QuichMatch[7] [52]. As in Sec. 4.1, performance was measured in terms of misclassification error, defined as the percentage of misclassified points over the total amount of classified image points, and we also computed the percentage of points classified by each method.

Results are reported in Fig. 3. Note that the error of SYNCH remains equal to 0% with up to 50% of mismatches and the percentage of classified points remains equal to 100% with up to 40% of mismatches. Subset and RSIM, instead, are not robust to mismatches (to different extends). Indeed, errors in the correspondences propagate into the tracks making traditional motion segmentation difficult to solve. By manual inspection it was found that Subset and

[4] https://github.com/panji1990/Robust-shape-interaction-matrix
[5] https://alex-xun-xu.github.io/ProjectPage/CVPR_18/
[6] http://www.maths.lth.se/matematiklth/personal/calle/sys_paper/sys_paper.html
[7] https://bitbucket.org/tronroberto/quickshiftmatching

Table 2: Average misclassification error [%] for several methods on the Hopkins155 benchmark [50]. Results are copied from [61]. The percentage of points classified by SYNCH is also reported.

	LSA [62] Error	GPCA [54] Error	ALC [37] Error	SSC [9] Error	TPV [23] Error	LRR [25] Error	T-Linkage [27] Error	S^3C [21] Error	RSIM [16] Error	MSSC [20] Error	KerAdd [61] Error	Coreg [61] Error	Subset [61] Error	SYNCH- tracks Error	SYNCH Error	SYNCH Classified
2 Motions	4.23	4.59	2.40	1.52	1.57	1.33	0.86	1.94	0.78	0.54	0.27	0.37	**0.23**	2.70	0.81	96.13
3 Motions	7.02	28.66	6.69	4.40	4.98	4.98	5.78	4.92	1.77	1.84	0.66	0.75	**0.58**	6.99	2.48	85.01
All	4.86	10.02	3.56	2.18	2.34	1.59	1.97	2.61	1.01	0.83	0.36	0.46	**0.31**	3.67	1.19	93.61

Table 3: Average and median misclassification error [%] for several methods on the Hopkins12 benchmark [56]. Results for different variants of ALC and SSC are taken from [16] whereas results for the remaining methods are copied from the respective papers. The percentage of points classified by SYNCH is also reported.

	PF [56] Error	PF+ALC [37] Error	RPCA+ALC [37] Error	ℓ_1+ALC [37] Error	SSC-R [9] Error	SSC-O [9] Error	RSIM [16] Error	KerAdd [61] Error	Coreg [61] Error	Subset [61] Error	SYNCH- tracks Error	SYNCH Error	SYNCH Classified
Mean	14.94	10.81	13.78	1.28	3.82	8.78	0.61	0.11	**0.06**	**0.06**	5.46	3.19	88.06
Median	9.31	7.85	8.27	1.07	0.31	4.80	0.61	**0.00**	**0.00**	**0.00**	0.57	0.28	99.35

RSIM cluster all the tracks, and unclassified data correspond to image points that were not included in any track by the algorithm used for multi-image matching. The good behaviour of our technique is due to two reasons: first, a robust method (RPA) is used for computing the partial segmentations; secondly, SYNCH is able to reduce (potential) errors in the partial segmentations by exploiting redundant measures in a global way. This aspect can be appreciated in Fig. 4, which reports the histograms of misclassification error achieved by RPA over all the image pairs. Observe that RPA produces errors even in the absence of wrong correspondences. Let us consider (e.g.) the case of 40% of mismatches: it is remarkable that, despite individual partial segmentations are noisy, our method achieves zero error, as shown in Fig. 3.

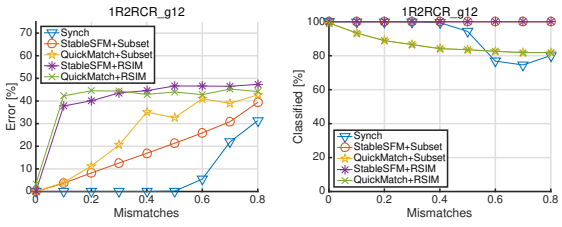

Figure 3: Misclassification error [%] and classified points [%] versus fraction of mismatches for several methods on the *1R2RCR_g12* sequence [50].

4.2.3 Real data

In order to test our method on real data, we created a small benchmark consisting of 7 image collections. Note that there are no standard datasets for the specific task of segmentation from pairwise matches. We considered indoor scenes with two motions where one object is fixed (i.e. it belongs to the background), and we acquired from 6 to 10

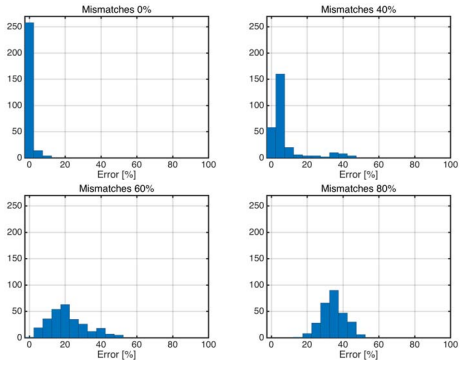

Figure 4: Histograms of misclassification error achieved by RPA [28] on a single trial on the *1R2RCR_g12* sequence [50]. The horizontal axis corresponds to the misclassification error in an individual image pair; the vertical axis corresponds to the number of pairs where a given error is obtained.

images of size 2922×2000 with a moving camera. More details about the dataset[8] are available in the supplementary material. SIFT keypoints [26] were extracted in all the images and correspondences between image pairs were established using the nearest neighbor and ratio test as in [26], using the VLFeat library[9]. Only symmetric matches were kept and points that are not matched in any image were removed. Such correspondences are noisy, as shown in Fig. 6.

As in Sec. 4.2.2, we compared our approach with RSIM [16] and Subset [61], where StableSfM [32] and QuickMatch [52] were used to compute tracks over multiple images. In order to compute the misclassification error, we manually labelled points in each sequence, thus producing a ground-truth segmentation. Results are shown in Tab. 4, which also reports the percentage of points classified by each method. See also Fig. 5 and the supplementary ma-

[8] https://github.com/federica-arrigoni/ICCV_19
[9] http://www.vlfeat.org/

Table 4: Misclassification error [%] and classified points [%] for several methods on our dataset. The number of motions d, the number of images n, and the total number of image points p are also reported.

				SYNCH		StableSfM + Subset [61]		QuichMatch + Subset [61]		StableSfM + RSIM [16]		QuichMatch + RSIM [16]	
Dataset	d	n	p	Error	Classified	Error	Classified	Error	Classified	Error	Classified	Error	Classified
Pen	2	6	4550	**0.82**	83.23	17.12	99.36	14.57	82.07	13.94	99.36	12.13	82.07
Pouch	2	6	4971	**4.15**	69.89	26.14	99.60	24.09	66.12	32.60	99.60	37.30	66.12
Needlecraft	2	6	6617	**1.04**	76.76	19.13	99.61	17.97	72.51	23.58	99.61	26.84	72.51
Biscuits	2	6	13158	**0.51**	87.28	9.22	99.47	8.91	82.49	4.58	99.47	34.87	82.49
Cups	2	10	14664	**1.01**	69.82	12.53	99.29	12.97	74.75	22.78	99.29	33.09	74.75
Tea	2	10	32612	28.12	52.21	7.11	99.37	**5.46**	82.67	46.98	99.37	41.99	82.67
Food	2	10	36723	**0.56**	80.66	12.86	99.32	13.83	72.85	19.18	99.32	20.38	72.85

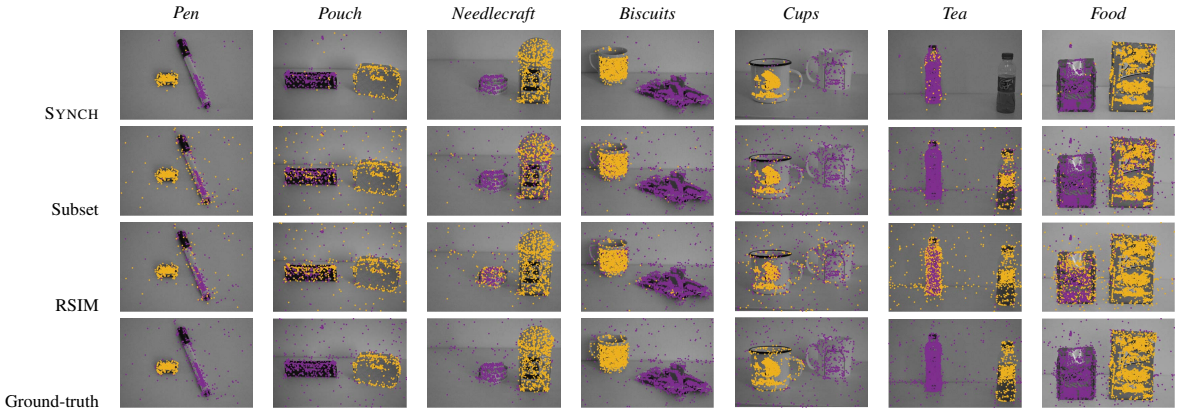

Figure 5: Segmentation results are reported on sample images from our dataset. Different colours encode the membership to different motions. For better visualization, unclassified points are not drawn. Among all the competitors, results for Subset and RSIM combined with StableSfM are reported only.

terial for qualitative evaluations.

Our method achieves the lowest misclassification error in 6 out of 7 sequences, outperforming both Subset and RSIM, and it classifies a significant amount of points in most cases. Traditional methods present poor performances on our dataset since they are not robust to mismatches, confirming the outcome of the synthetic experiments in Sec. 4.2.2. The *Tea* sequence constitutes a failure case of our approach. After inspecting the solution, it was found that the gap between the third-largest eigenvalue (which should be zero) and the second-largest eigenvalue is not significant, meaning that it is difficult to solve motion segmentation via spectral decomposition. Recall that SYNCH solves a *relaxed* version of the original optimization problem. Hence, although based on reasonable principles, there are no guarantees that it provides a correct segmentation.

5. Conclusion

We formulated motion segmentation in a novel way as a "synchronization" problem, where the task is to find a consistent set of total segmentations – which represent the clustering of points in all the images, starting from a redundant set of partial segmentations – which represent the clustering of corresponding points in different image pairs. We showed that such a problem can be (approximately) solved via spectral decomposition, generalizing to binary matrices previous works on synchronization of rotations, rigid motions, homographies and permutations. Our approach can deal with mismatches, as demonstrated by synthetic and real experiments, but it lacks theoretical guarantees. It also admits an interesting interpretation in terms of spectral clustering. Future work will explore under which assumptions (if any) SYNCH exactly recovers the unknown total segmentations. We also aim at investigating different relaxations (e.g. semidefinite programming) to solve Problem (6).

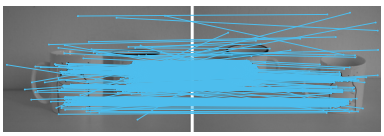

Figure 6: SIFT matches on a sample pair from the *Cups* sequence.

Acknowledgements. The authors would like to thank Luca Magri and Stanislav Steidl for their help with the experiments. This work was supported by the European Regional Development Fund under the project IMPACT (reg. no CZ.02.1.01/0.0/0.0/15_003/0000468).

References

[1] Mica Arie-Nachimson, Shahar Z. Kovalsky, Ira Kemelmacher-Shlizerman, Amit Singer, and Ronen Basri. Global motion estimation from point matches. *Proceedings of the Joint 3DIM/3DPVT Conference: 3D Imaging, Modeling, Processing, Visualization and Transmission*, 2012.

[2] Federica Arrigoni, Eleonora Maset, and Andrea Fusiello. Synchronization in the symmetric inverse semigroup. In *International Conference on Image Analysis and Processing*, pages 70–81. Springer, 2017.

[3] Federica Arrigoni, Beatrice Rossi, and Andrea Fusiello. Spectral synchronization of multiple views in SE(3). *SIAM Journal on Imaging Sciences*, 9(4):1963 – 1990, 2016.

[4] Daniel Barath and Jiri Matas. Multi-class model fitting by energy minimization and mode-seeking. In *Proceedings of the European Conference on Computer Vision*, pages 229–245. Springer International Publishing, 2018.

[5] Florian Bernard, Johan Thunberg, Peter Gemmar, Frank Hertel, Andreas Husch, and Jorge Goncalves. A solution for multi-alignment by transformation synchronisation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[6] Avishek Chatterjee and Venu Madhav Govindu. Robust relative rotation averaging. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[7] Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh. Learning non-linear combinations of kernels. In *Neural Information Processing Systems*, pages 396–404. 2009.

[8] João Paulo Costeira and Takeo Kanade. A multibody factorization method for independently moving objects. *International Journal of Computer Vision*, 29(3):159–179, 1998.

[9] Ehsan Elhamifar and René Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(11):2765–2781, 2013.

[10] Johan Fredriksson and Carl Olsson. Simultaneous multiple rotation averaging using lagrangian duality. In *Proceedings of the Asian Conference on Computer Vision*, 2012.

[11] Arvind Giridhar and P.R. Kumar. Distributed clock synchronization over wireless networks: Algorithms and analysis. *Proceedings of the IEEE Conference on Decision and Control*, pages 4915–4920, 2006.

[12] Venu Madhav Govindu. Lie-algebraic averaging for globally consistent motion estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 684–691, 2004.

[13] Richard Hartley, Jochen Trumpf, Yuchao Dai, and Hongdong Li. Rotation averaging. *International Journal of Computer Vision*, 2013.

[14] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., 2001.

[15] Hossam Isack and Yuri Boykov. Energy-based geometric multi-model fitting. *International Journal of Computer Vision*, 97(2):123–147, 2012.

[16] Pan Ji, Mathieu Salzmann, and Hongdong Li. Shape interaction matrix revisited and robustified: Efficient subspace clustering with corrupted and incomplete data. In *Proceedings of the International Conference on Computer Vision*, pages 4687–4695, 2015.

[17] Richard Karp, Jeremy Elson, Deborah Estrin, and Scott Shenker. Optimal and global time synchronization in sensor nets. Technical report, Center for Embedded Networked Sensing: University of California, Los Angeles, 2003.

[18] Da Kuang, Sangwoon Yun, and Haesun Park. SymNMF: nonnegative low-rank approximation of a similarity matrix for graph clustering. *Journal of Global Optimization*, pages 1–30, 2014.

[19] Abhishek Kumar, Piyush Rai, and Hal Daume. Co-regularized multi-view spectral clustering. In *Neural Information Processing Systems*, pages 1413–1421. 2011.

[20] Taotao Lai, Hanzi Wang, Yan Yan, Tat-Jun Chin, and Wan-Lei Zhao. Motion segmentation via a sparsity constraint. *IEEE Transactions on Intelligent Transportation Systems*, 18(4):973–983, 2017.

[21] Chun-Guang Li and R. Vidal. Structured sparse subspace clustering: A unified optimization framework. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 277–286, 2015.

[22] Ting Li, Vinutha Kallem, Dheeraj Singaraju, and René Vidal. Projective factorization of multiple rigid-body motions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–6, 2007.

[23] Zhuwen Li, Jiaming Guo, Loong-Fah Cheong, and Steven Zhiying Zhou. Perspective motion segmentation via collaborative clustering. In *Proceedings of the International Conference on Computer Vision*, pages 1369–1376, 2013.

[24] Zhouchen Lin, Minming Chen, and Yi Ma. The augmented lagrange multiplier method for exact recovery of corrupted Low-Rank matrices. eprint arXiv:1009.5055, 2010.

[25] Guangcan Liu, Zhouchen Lin, Shuicheng Yan, Ju Sun, Yong Yu, and Yi Ma. Robust recovery of subspace structures by low-rank representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 171–184, 2013.

[26] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[27] Luca Magri and Andrea Fusiello. T-Linkage: A continuous relaxation of J-Linkage for multi-model fitting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3954–3961, June 2014.

[28] Luca Magri and Andrea Fusiello. Robust multiple model fitting with preference analysis and low-rank approximation. In *Proceedings of the British Machine Vision Conference*, pages 20.1–20.12. BMVA Press, September 2015.

[29] Luca Magri and Andrea Fusiello. Multiple models fitting as a set coverage problem. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3318–3326, June 2016.

[30] Eleonora Maset, Federica Arrigoni, and Andrea Fusiello. Practical and efficient multi-view matching. In *Proceedings of IEEE International Conference on Computer Vision*, pages 4568–4576, 2017.

[31] Andrew Y Ng, Michael I Jordan, Yair Weiss, et al. On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, 2:849–856, 2002.

[32] Carl Olsson and Olof Enqvist. Stable structure from motion for unordered image collections. In *Proceedings of the 17th Scandinavian conference on Image analysis (SCIA'11)*, pages 524–535. Springer-Verlag, 2011.

[33] Kemal Egemen Ozden, Konrad Schindler, and Luc Van Gool. Multibody structure-from-motion in practice. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(6):1134–1141, 2010.

[34] Onur Ozyesil, Vladislav Voroninski, Ronen Basri, and Amit Singer. A survey of structure from motion. *Acta Numerica*, 26:305 – 364, 2017.

[35] Deepti Pachauri, Risi Kondor, and Vikas Singh. Solving the multi-way matching problem by permutation synchronization. In *Advances in Neural Information Processing Systems 26*, pages 1860–1868. Curran Associates, Inc., 2013.

[36] Trung-Thanh Pham, Tat-Jun Chin, Jin Yu, and David Suter. The random cluster model for robust geometric fitting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(8):1658–1671, 2014.

[37] Shankar Rao, Roberto Tron, Rene Vidal, and Yi Ma. Motion segmentation in the presence of outlying, incomplete, or corrupted trajectories. *Pattern Analysis and Machine Intelligence*, 32(10):1832–1845, 2010.

[38] David M. Rosen, Luca Carlone, Afonso S. Bandeira, and John J. Leonard. A certifiably correct algorithm for synchronization over the special Euclidean group. In *International Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2016.

[39] Reza Sabzevari and Davide Scaramuzza. Monocular simultaneous multi-body motion segmentation and reconstruction from perspective views. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 23–30, 2014.

[40] Emanuele Santellani, Eleonora Maset, and Andrea Fusiello. Seamless image mosaicking via synchronization. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-2:247–254, 2018.

[41] Muhamad Risqi U. Saputra, Andrew Markham, and Niki Trigoni. Visual SLAM and structure from motion in dynamic environments: A survey. *ACM Computing Surveys*, 51(2):37:1–37:36, 2018.

[42] Pierre Schroeder, Adrien Bartoli, Pierre Georgel, and Nassir Navab. Closed-form solutions to multiple-view homography estimation. In *Applications of Computer Vision (WACV), 2011 IEEE Workshop on*, pages 650–657, 2011.

[43] Yanyao Shen, Qixing Huang, Nati Srebro, and Sujay Sanghavi. Normalized spectral map synchronization. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 4925–4933. Curran Associates, Inc., 2016.

[44] Jianbo Shi and Jitendra Malik. Normalized Cuts and Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[45] Amit Singer. Angular synchronization by eigenvectors and semidefinite programming. *Applied and Computational Harmonic Analysis*, 30(1):20 – 36, 2011.

[46] Amit Singer and Yoel Shkolnisky. Three-dimensional structure determination from common lines in cryo-EM by eigenvectors and semidefinite programming. *SIAM Journal on Imaging Sciences*, 4(2):543 – 572, 2011.

[47] Roberto Toldo and Andrea Fusiello. Robust multiple structures estimation with J-Linkage. In *Proceedings of the European Conference on Computer Vision*, pages 537–547, 2008.

[48] Philip H. S. Torr. Geometric motion segmentation and model selection. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 356(1740):1321–1340, 1998.

[49] Roberto Tron and Kostas Danilidis. Statistical pose averaging with varying and non-isotropic covariances. In *Proceedings of the European Conference on Computer Vision*, 2014.

[50] Roberto Tron and René Vidal. A benchmark for the comparison of 3-d motion segmentation algorithms. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.

[51] Roberto Tron, Xiaowei Zhou, and Kostas Daniilidis. A survey on rotation optimization in structure from motion. In *Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2016.

[52] Roberto Tron, Xiaowei Zhou, Carlos Esteves, and Kostas Daniilidis. Fast multi-image matching via density-based clustering. In *Proceedings of the International Conference on Computer Vision*, pages 4077–4086, 2017.

[53] René Vidal and Richard Hartley. Three-view multibody structure from motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):214–227, 2008.

[54] René Vidal, Yi Ma, and S. Shankar Sastry. Generalized principal component analysis (GPCA). *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(12):1945–1959, 2005.

[55] René Vidal, Yi Ma, Stefano Soatto, and Shankar Sastry. Two-view multibody structure from motion. *International Journal of Computer Vision*, 68(1):7–25, 2006.

[56] René Vidal, Roberto Tron, and Richard Hartley. Multiframe motion segmentation with missing data using powerfactorization and GPCA. *International Journal of Computer Vision*, 79(1):85–105, 2008.

[57] Esther Vincent and Robert Laganiere. Detecting planar homographies in an image pair. In *International Symposium on Image and Signal Processing and Analysis*, pages 182–187, 2001.

[58] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

[59] Xiang Wang, Buyue Qian, and Ian Davidson. On constrained spectral clustering and its applications. *Data Mining and Knowledge Discovery*, 28(1):1–30, 2014.

[60] Lei Xu, Erkki Oja, and Pekka Kultanen. A new curve detection method: randomized Hough transform (RHT). *Pattern Recognition Letters*, 11(5):331–338, 1990.

[61] Xun Xu, Loong-Fah Cheong, and Zhuwen Li. Motion segmentation by exploiting complementary geometric models.

In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[62] Jingyu Yan and Marc Pollefeys. A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and nondegenerate. In *Proceedings of the European Conference on Computer Vision*, pages 94–106, 2006.

[63] Jin-Gang Yu, Gui-Song Xia, Ashok Samal, and Jinwen Tian. Globally consistent correspondence of multiple feature sets using proximal Gauss–Seidel relaxation. *Pattern Recognition*, 51:255 – 267, 2016.

[64] Luca Zappella, Alessio Del Bue, Xavier Lladó, and Joaquim Salvi. Joint estimation of segmentation and structure from motion. *Computer Vision and Image Understanding*, 117(2):113 – 129, 2013.

[65] Wei Zhang and Jana Kosecká. Nonparametric estimation of multiple structures with outliers. In *Workshop on Dynamic Vision, European Conference on Computer Vision 2006*, volume 4358 of *Lecture Notes in Computer Science*, pages 60–74. Springer, 2006.

[66] Marco Zuliani, Charles S. Kenney, and B. S. Manjunath. The multiRANSAC algorithm and its application to detect planar homographies. In *Proceedings of the IEEE International Conference on Image Processing*, pages III–153–6, 2005.