On the Usage of the Trifocal Tensor in Motion Segmentation

Federica Arrigoni[1], Luca Magri[2], and Tomas Pajdla[3]

[1] DISI, University of Trento, Italy – federica.arrigoni@unitn.it
[2] DEIB, Politecnico di Milano, Italy – luca.magri@polimi.it
[3] Czech Institute of Informatics, Robotics and Cybernetics (CIIRC), Czech Technical University in Prague, Czech Republic – pajdla@cvut.cz

Abstract. Motion segmentation, i.e., the problem of clustering data in multiple images based on different 3D motions, is an important task for reconstructing and understanding dynamic scenes. In this paper we address motion segmentation in multiple images by combining partial results coming from triplets of images, which are obtained by fitting a number of trifocal tensors to correspondences. We exploit the fact that the trifocal tensor is a stronger model than the fundamental matrix, as it provides fewer but more reliable matches over three images than fundamental matrices provide over the two. We also consider an alternative solution which merges partial results coming from both triplets and pairs of images, showing the strength of three-frame segmentation in a combination with two-frame segmentation. Our real experiments on standard as well as new datasets demonstrate the superior accuracy of the proposed approaches when compared to previous techniques.

Keywords: motion segmentation, structure from motion, multi-model fitting, trifocal tensor

1 Introduction

Motion segmentation, i.e., the problem of clustering data in multiple images based on different 3D motions, has attracted a lot of attention in Computer Vision. Existing techniques can be divided into three categories, according to the type of data that is being clustered and the assumptions that are made about the input.

The first category, which accounts for the majority of works in the literature, assumes that a set of points is tracked through multiple images, and the task is to cluster those trajectories (i.e., *multi-frame* correspondences) into different groups based on the moving object they belong to. Methods performing subspace separation (e.g., [44, 48, 33, 7, 25, 17]) and multi-model fitting (e.g., [39, 5, 14, 6, 28, 4]) belong to this category. Other solutions include [37, 30, 35, 23, 22, 47]. The typical scenario consists in videos where there are small motions between consecutive frames (e.g., the Hopkins benchmark [42]). This involves several applications such as surveillance [19], scene understanding [34] and autonomous driving [9]. We name this category "trajectory clustering" (see Fig. 1).

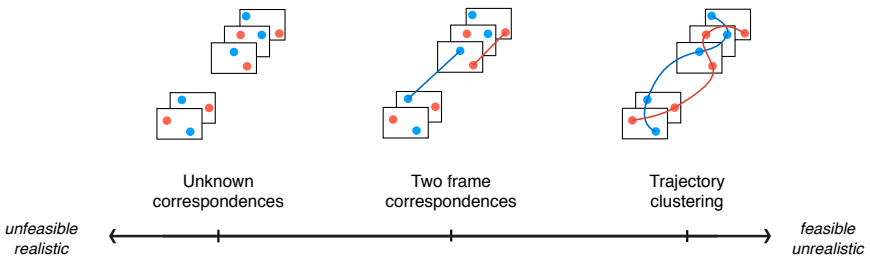


Fig. 1: The proposed taxonomy divides existing approaches into three categories: trajectory clustering; segmentation with two-frame correspondences; segmentation with unknown correspondences. When moving from right to left the problem becomes more difficult to solve since assumptions are weaker (but more realistic). This paper comes under the middle category.

The second category considers the problem of clustering image points (e.g., SIFT keypoints [26]) into different motions, assuming that matches between pairs of images (i.e., *two-frame* correspondences) are available only. This task is poorly studied and there are only a few works addressing it [3, 2]. The typical scenario involves unstructured/unordered image sets where there are large motions between different frames (e.g., the indoor scenes used in [3, 2]). This finds application in multi-body structure from motion [36], where the objective is to reconstruct a 3D scene containing multiple moving objects. This category is represented in the middle of Fig. 1.

The third category, shown in the left part of Fig. 1, assumes that a set of image points (e.g., SIFT keypoints) is given and considers the case of *unknown* correspondences. This problem is addressed in [15, 46] only, where the authors aim at computing multi-frame correspondences while at the same time classifying those trajectories into different groups. However, such approaches are not suitable for practical applications: only sequences with (at most) 200 tracks are analyzed in [15, 46] due to algorithmic complexity.

In the first case, multi-frame correspondences are needed *before* motion segmentation. Note that recovering trajectories in the presence of multiple moving objects is a hard task [16]. To overcome such a difficulty, correspondences are usually cleaned with manual operations (see, e.g., the Hopkins benchmark), hence they are not realistic at all. In the second case, multi-frame correspondences are not computed explicitly. However, they could be recovered *after* motion segmentation as follows: single-body techniques (e.g., RANSAC [8]) can be used to clean the input two-frame correspondences for each motion; then, existing solutions (e.g., QuickMatch [43] or StableSfM [29]) can be used for getting tracks starting from those (refined) two-frame correspondences. In the third case, multi-frame correspondences are computed *during* motion segmentation. However, addressing segmentation under such a weak assumption is very challenging due to the large number of unknowns, and existing solutions [15, 46] are not practical yet. Note that methods belonging to a specific category are, in general, sub-optimal

when applied to the task associated with another category. To sum up, the second category lies at the middle between the first one and the third one, hence it can be viewed as a good trade-off between making realistic assumptions and addressing a feasible/practical task. This motivates our interest in those methods, which are reviewed in Sec. 1.1.

1.1 Related Work

The most related work [3, 2] address motion segmentation in two steps:

1. motion segmentation is solved independently on different image pairs;
2. such partial results are combined in order to get a multi-frame segmentation.

Concerning the first step, multiple fundamental matrices are fitted to corresponding points via Robust Preference Analysis (RPA) [27]. Concerning the second step, different techniques are proposed.

In [2] all the two-frame segmentations produced by Step 1 are represented as binary matrices, and they are collected in a big block-matrix. Then, the unknown multi-frame segmentation is recovered from the spectral decomposition of such a matrix, followed by a rounding procedure. This method – named SYNCH– can be viewed as a "synchronization" of binary matrices [1] or as a special case of "spectral clustering" [45].

In [3] it is observed that all the two-frame segmentations involving a fixed image provide – up to a permutation of the motions – a possible solution for clustering points in that image. In order to resolve such ambiguity, *permutation synchronization* [31] is performed. Then, each point is assigned to the most frequent label (i.e., the mode) among all the possible solutions coming from different two-frame segmentations. For this reason the method is named MODE.

1.2 Contribution

In this paper we propose two methods that tackle motion segmentation by exploiting the trifocal tensor, motivated by the fact that the latter constitutes a stronger model than the fundamental matrix. Indeed, it is well known that the trifocal tensor can be used to determine the exact position of a point in a third image (given its position in the other two images), hence there are fewer mismatches over three images than there are over two [11]. In the case of the fundamental matrix, instead, there is only the weaker constraint of an epipolar line against which to verify a possible correspondence.

Our methods are outlined in Fig. 2. They belong to the second category (i.e., the case of two-frame correspondences represented in the middle of Fig. 1) and are inspired by [3]. The first approach – named TRISEG– addresses motion segmentation in two steps:

1. motion segmentation is solved independently on different triplets of images;
2. such partial results are combined in order to get a multi-frame segmentation.

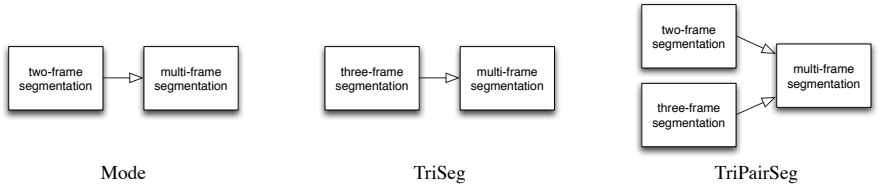


Fig. 2: The method developed in [3] and our approaches combine segmentation results independently obtained from subsets of images: Mode [3] considers pairs of images (i.e., the fundamental matrix); TriSeg considers triplets of images (i.e., the trifocal tensor); TriPairSeg considers both pairs and triplets of images (i.e., both the fundamental matrix and the trifocal tensor).

Concerning Step 1, we exploit RPA [27] in order to fit multiple trifocal tensors to correspondences. Concerning Step 2, we adapt the method proposed in [3] – which was developed for merging results coming from pairs of images – in order to deal with triplets of images. The relevance of the trifocal tensor for addressing motion segmentation in *three* images was already observed in some early works [41, 40, 10]. However, this is the first paper where the trifocal tensor is exploited in order to solve motion segmentation in *multiple* images.

The second approach – named TriPairSeg– is made of three steps:

1. motion segmentation is solved independently on different pairs of images;
2. motion segmentation is solved independently on different triplets of images;
3. the partial results derived in the first two steps are combined in order to get a multi-frame segmentation.

Concerning Step 1, multiple fundamental matrices are fitted to corresponding points via RPA, as done in [3]. Concerning Step 2, multiple trifocal tensors are fitted to correspondences via RPA, as done by TriSeg. Concerning Step 3, we explain how TriSeg can be easily adapted in order to deal with both pairs and triplets of images. The idea of merging results coming from different models is also present in [47], where the authors consider both the homography, the fundamental matrix and the affine subspace. Such approach, however, differs from ours in three respects. First of all, it addresses a different task for it belongs to the first category of methods. Secondly, the analysed models are not used one at a time to provide a possible segmentation involving a subset of images – as happens for our method – but they are used all together to build an accumulated affinity matrix. Finally, the trifocal tensor is not used in [47].

The proposed solutions were validated on previous datasets and compared to the state of the art. Moreover, a new image collection was created, which comprises six indoor scenes with three or four motions. Results show that: our methods outperform both Mode [3] and Synch [2] in terms of misclassification error; they successfully handle sequences with four motions, whereas the competing methods either fail on a few images or produce useless results; TriSeg usually classifies less points than TriPairSeg with higher accuracy.

The paper is organized as follows. Section 2 is devoted to our solutions to motion segmentation: Sec. 2.1 describes TriSeg whereas Sec. 2.2 presents TriPairSeg. Experimental results are reported in Sec. 3 and Sec. 3.1 explains how the RPA algorithm can be used in order to fit multiple trifocal tensors. The conclusion is drawn in Sec. 4.

2 Proposed Methods

Let us introduce some useful notation. Let n denote the number of images and let d denote the number of motions, which is known by assumption. Similarly to [3, 2], we assume that a set of points is given in all the images and correspondences between points in image pairs have been established (using SIFT [26] for instance). Let p_i denote the number of points in image i, and let $p = \sum_{i=1}^{n} p_i$ denote the total amount of points over all the images. Let $\mathbf{s}_i \in \{0, 1, \dots, d\}^{p_i}$ be a vector – named the *total segmentation* of image i – representing the labels of points in image i. Labels from 1 to d identify the membership to a specific motion, while the zero label identifies those points (also known as *unclassified* points) whose cluster can not be established due to high corruption in the correspondences. The goal here is to estimate $\mathbf{s}_i$ for all $i = 1, \dots, n$. Two approaches are developed to accomplish such a task, which are presented in Sec. 2.1 and 2.2.

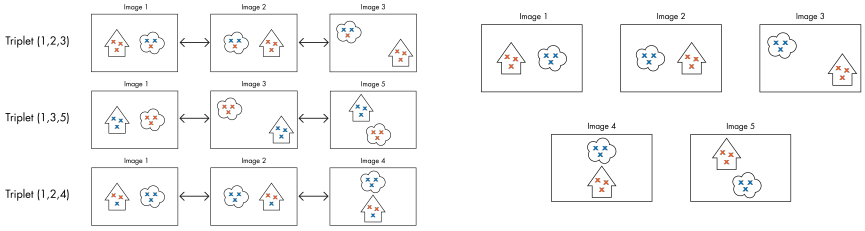
2.1 TriSeg

Let $\alpha = (i, j, k)$ be a triplet of images and let $\mathbf{t}_\alpha \in \{0, 1, \dots, d\}^{m_\alpha}$ be a vector – named the *partial segmentation* of triplet α – representing the labels of corresponding points in images i, j and k, where $m_\alpha \leqslant \min\{p_i, p_j, p_k\}$ denotes the number of correspondences in the triplet. Hereafter Greek letters are used to denote triplets of images. In practice each partial segmentation is computed by fitting multiple trifocal tensors to correspondences with RPA [27], as explained in Sec. 3.1, where points labelled as outlier (if any) are given the zero label. Note that the usage of the trifocal tensor is a relevant difference with respect to [3], where fundamental matrices are used. Such difference brings significant improvement in performance, as shown in Sec. 3. The goal here is to estimate the total segmentations starting from a redundant set of partial segmentations, as shown in Fig. 3. In this respect, two issues have to be addressed:

- each partial segmentation considers its own labelling of the motions, i.e., the same motion may be given a different label in different triplets;
- each partial segmentation may contain some errors, which can be caused either by wrong correspondences or by failure of the RPA algorithm.

We now explain how to address the first challenge. Note that $\mathbf{t}_\alpha \in \{0, 1, \dots, d\}^{m_\alpha}$ gives rise to three vectors

$$\begin{aligned} \mathbf{s}_i^\alpha &\in \{0, 1, \dots, d\}^{p_i} \\ \mathbf{s}_j^\alpha &\in \{0, 1, \dots, d\}^{p_j} \\ \mathbf{s}_k^\alpha &\in \{0, 1, \dots, d\}^{p_k} \end{aligned} \tag{1}$$



(a) three-frame segmentation (b) multi-frame segmentation

Fig. 3: The task of TriSeg is to assign a label (blue or red) to each point in multiple images based on the moving object (house or cloud) it belongs to. The starting point is a set of partial results obtained by solving motion segmentation on different triplets. Observe that such results may contain errors and they are not absolute: the house is given the red label in the first triplet but it is given the blue label in the second triplet.

which contain labels of corresponding points in images i, j and k, where missing correspondences are given the zero label. Observe that the superscript in Eq. (1) refers to the triplet whereas subscripts refer to the images in the triplet. Let us construct a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with vertex set $\mathcal{V}$ and edge set $\mathcal{E}$ as follows:

- each vertex corresponds to one triplet;
- an edge is drawn between two vertices each time the associated triplets have one image in common.

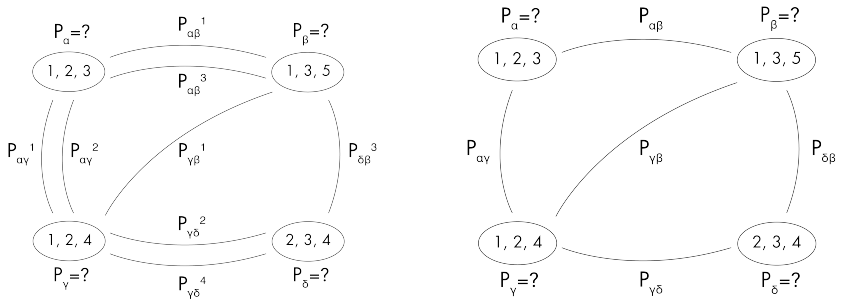
Note that $\mathcal{G}$ is a *multigraph*, i.e., a graph with multiple edges: in the case where two triplets share two images there will be two edges between the corresponding vertices, as shown in Fig. 4a. Observe that a multigraph is not constructed in [3], since different pairs can not share two images but (at most) one.

Each vertex in the multigraph is associated with an *unknown* permutation and each edge is associated with a *known* permutation[4]. Let P_α denote the $d \times d$ permutation matrix associated with vertex α, which corresponds to triplet α. The interpretation is that – after applying P_α to the partial segmentation $\mathbf{t}_\alpha$ – the ambiguity in the local labelling of motions is fixed, i.e., the same motion has the same label in different triplets. Let k denote a common image between triplets α and β (i.e., $k \in \alpha \cap \beta$) and let $P^k_{\alpha\beta}$ denote the $d \times d$ permutation matrix associated with the k-th edge between vertices α and β. Such a matrix represents the permutation that best maps the vector $\mathbf{s}^\alpha_k$ (i.e., labels of image k in triplet α) into the vector $\mathbf{s}^\beta_k$ (i.e., labels of image k in triplet β):

$$P^k_{\alpha\beta} = \text{bestMap}\,(\mathbf{s}^\alpha_k, \mathbf{s}^\beta_k). \tag{2}$$

Recall that $\mathbf{s}^\alpha_k$ and $\mathbf{s}^\beta_k$ are recovered from $\mathbf{t}_\alpha$ and $\mathbf{t}_\beta$ respectively, as stated by Eq. (1). Finding $P^k_{\alpha\beta}$ is a *linear assignment* problem, which can be solved with the Hungarian algorithm [21].

[4] Observe that these permutations are represented as *square* matrices since we are assuming that the number of motions is known and constant over all the frames.



(a) multigraph (b) simple graph

Fig. 4: The relations between different triplets of images can be represented as multigraph or a simple graph. In both cases each vertex corresponds to one triplet. In the multigraph an edge connects two triplets all the times they share one image. Triplets $(1, 2, 3)$ and $(1, 3, 5)$, for instance, are linked by two edges since they have two common images. In the simple graph, instead, one (single) edge is drawn between two triplets if and only if they share (at least) one image. Vertices correspond to unknown permutations and edges correspond to known permutations, as explained in the text.

Now we turn $\mathcal{G}$ into a *simple graph* (i.e., a graph without multiple edges) in order to have (at most) one single measure between each pair of vertices (instead of multiple measures), as shown in Fig. 4b. Thus the task is to find a permutation $P_{\alpha\beta}$ associated with edge (α, β) that best represents (or, in other words, that "averages") the set $\{P^k_{\alpha\beta}$ s.t. $k \in \alpha \cap \beta\}$:

$$P_{\alpha\beta} = \text{mean}\{P^k_{\alpha\beta} \text{ s.t. } k \in \alpha \cap \beta\}. \tag{3}$$

Finding $P_{\alpha\beta}$ can be cast to a linear assignment problem, as explained in the supplementary material, which can be solved with the Hungarian algorithm [21].

Now we have to face the problem of computing an unknown permutation P_α for each vertex $\alpha \in \mathcal{V}$ starting from a redundant set of permutations $P_{\alpha\beta}$ with $(\alpha, \beta) \in \mathcal{E}$, where $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a simple graph. It can be seen that such matrices satisfy the following consistency relation

$$P_{\alpha\beta} = P_\alpha P_\beta^\mathsf{T} \tag{4}$$

which defines a *permutation synchronization* problem [31]. Equation (4) can be solved via spectral decomposition (see [31, 38] for more details). At this point the permutation P_α is applied to the partial segmentation $\mathbf{t}_\alpha$ for each triplet α. This has the effect of (possibly) reshuffling the labels of motions in individual triplets so that the permutation ambiguity is fixed, i.e., the same motion has the same label in different triplets.

We now explain how to deal with errors in individual partial segmentations, thus addressing the second challenge mentioned above. Recall that Eq. (1) means that each partial segmentation provides a possible solution for the total segmentation of the three images involved in the triplet. Hence, for a given image,

several estimates are available for its total segmentation. If $\mathcal{T}_i$ denotes the set of triplets involving image i, then such estimates are given by $\{\mathbf{s}_i^\alpha \text{ s.t. } \alpha \in \mathcal{T}_i\}$. In order to assign a unique label to each point, the following criterion [3] is used

$$\mathbf{s}_i[r] = \text{mode } \{\mathbf{s}_i^\alpha[r] \text{ s.t. } \alpha \in \mathcal{T}_i,\ \mathbf{s}_i^\alpha[r] \neq 0\} \tag{5}$$

with $r = 1, \ldots, p_i$ and $i = 1, \ldots, n$. The idea is that the most frequent label (i.e. the *mode*) is, in general, correct in the presence of moderate noise. The condition $\mathbf{s}_i^\alpha[r] \neq 0$ means that both missing correspondences and points labelled as outlier (if any) by RPA are ignored, and the mode is computed over remaining points. We set $\mathbf{s}_i[r] = 0$ (i.e., point r in image i is labelled as unknown) in the case where $\mathbf{s}_i^\alpha[r] = 0$ for all $\alpha \in \mathcal{T}_i$, meaning that the point is either missing or deemed as outlier in *all* the triplets. For the sake of robustness, we further require that the mode is equal to (at least) two measures, otherwise the point is labelled as unknown.

To summarize, our method – named TriSeg– is made of the following steps:

i) for each triplet α, the partial segmentation $\mathbf{t}_\alpha$ is computed by fitting multiple trifocal tensors with RPA (see Sec. 3.1); the three vectors in Eq. (1) are derived from $\mathbf{t}_\alpha$;
ii) for each pair (α, β) of triplets with some images in common, the following operations are performed: first, the permutation matrix $P_{\alpha\beta}^k$ is computed from Eq. (2) for all $k \in \alpha \cap \beta$ (linear assignment problem); then, the permutation matrix $P_{\alpha\beta}$ is computed from Eq. (3) (linear assignment problem – see supplementary material);
iii) the permutation matrices $P_\alpha, \ldots, P_\beta$ are computed simultaneously for all the triplets from Eq. (4) (permutation synchronization);
iv) for each triplet α, the permutation matrix P_α is applied to the partial segmentation $\mathbf{t}_\alpha$; the three vectors in Eq. (1) are derived from $\mathbf{t}_\alpha$;
v) for each image i, the total segmentation $\mathbf{s}_i$ is derived from Eq. (5).

Step i) is a pre-processing phase where motion segmentation is solved on triplets of images. Steps ii)-iv) aim at expressing all such partial/local results with respect to the same numbering of motions (see the first challenge mentioned at the beginning of this section). Step v) explains how to robustly assign a unique label to each point starting from multiple measures possibly corrupted by noise (see the second challenge mentioned at the beginning of this section).

2.2 TriPairSeg

We introduce here another technique – named TriPairSeg– which computes the total segmentations starting from partial segmentations of two different types, as shown in Fig. 2. Such partial results are derived by fitting either fundamental matrices (in the case of image pairs) or trifocal tensors (in the case of triplets of images) via RPA. The idea is that, by using models of two different types, the advantages of both are inherited, as shown in Sec. 3.

It is straightforward to see that the approach developed in Sec. 2.1 applies equally well to this case, with the provision that the multigraph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is now constructed as follows: each vertex can be either an image pair or a triplet of images; an edge is present between two pairs if and only if they share one image (i.e., there are no multiple edges between two pairs); an edge is present between a triplet and a pair (or between two triplets) each time they have one image in common. After constructing the multigraph, TriPairSeg proceeds in the same way as TriSeg: it first solves linear assignment problems, it then performs permutation synchronization and it finally computes the mode.

3 Experiments

In this section we report experimental results on both existing datasets and new image collections. We implemented TriSeg and TriPairSeg in Matlab and we made our code publicly available[5]. We compared our approaches to previous techniques belonging to the same category (see Fig. 1), namely methods working under the mild assumption of two-frame correspondences (Mode [3] and Synch [2]), whose implementation is available online[6]. All the analysed techniques assumed that the number of motions d was known.

3.1 Implementation details

Given a set of two-frame correspondences, we proceed as follows in order to compute the partial segmentations, which constitute the input to TriSeg. First, triplets of images are identified: for the smallest sequences (i.e., $n < 10$) all the possible triplets are considered; in the remaining cases, a fixed number of triplets is sampled [32]. Such number is set equal to twice the number of image pairs. Then, for each triplet, a set of trajectories is computed by chaining two-frame correspondences. Note that these are *not* multi-frame correspondences, for they involve three images at a time. Moreover, observe that they are, in general, much noisier than the input two-frame correspondences: a mismatch between two images in the triplet propagates also to the third one. Finally, motion segmentation is solved in each triplet by fitting multiple trifocal tensors to those trajectories via RPA [27].

Robust Preference Analysis (RPA) is a general technique[7] for fitting multiple instances of a model to data corrupted by noise and outliers. Three main steps can be singled out. First, points are described in a conceptual space as vectors of "preferences", which measure how well they are fitted by a pool of provisional models instantiated via random sampling. Specifically, model hypotheses are instantiated from a minimal sample set (i.e., the minimum number of points necessary to fit a model), residuals are computed for every model, and the preference a point grants to a model is expressed in terms of its residual using the

[5] `https://github.com/federica-arrigoni/ECCV_20`
[6] `https://github.com/federica-arrigoni/ICCV_19`
[7] `http://www.diegm.uniud.it/fusiello/demo/rpa/`

Cauchy weight function [13]. Vectors are hence collected in a matrix that is segmented leveraging on robust principal component analysis [24] and symmetric non negative factorization [20]. A model is fitted to every cluster using robust statistics and the segmentation is accordingly refined. Possible applications of RPA include fitting geometric primitives (e.g., lines or circles) to points in the plane and fitting geometric models (e.g., fundamental matrices or homographies) to correspondences in an image pair. However, fitting trifocal tensors (and hence, performing motion segmentation in three images) has not been explored in [27]. In order to use RPA for such a task, we proceed as follows:

- we randomly sample subsets of 7 points and use them to instantiate a tentative trifocal tensor via linear estimation [12];
- residuals between points and a tensor are expressed using the reprojection error, as explained in [11];
- the final models are refined using Gauss-Helmert optimization with Ressl parametrization[8], as suggested in [18];
- the parameter σ_n (representing the standard deviation of the residuals of the inliers [27]) is set equal to 0.1 in all the experiments[9].

Concerning TriPairSeg, partial segmentations of two different types are required as input: the ones associated with triplets of images are computed by fitting trifocal tensors with RPA, as explained above; the ones associated with pairs of images are obtained by fitting fundamental matrices with RPA (using default values for the algorithmic parameters specified in [27]).

3.2 Existing datasets

We considered the benchmark provided in [3, 2] consisting of 12 indoor scenes with two or three motions counting from 6 to 10 images. Image points (with ground-truth labels) and noisy two-frame correspondences are available in this dataset. As done in [3, 2], we computed the *misclassification error* – defined as the percentage of misclassified points over the total amount of *classified* points[10] – and we also considered the percentage of points labelled by each method.

Results are reported in Table 1, showing that TriSeg achievest the lowest misclassification error in 9 out of 12 sequences, outperforming the competing techniques. This clearly shows the benefit of using the trifocal tensor, which is more robust to mismatches than the fundamental matrix. TriPairSeg is slightly better than Mode and significantly better than Synch in terms of accuracy.

[8] `https://github.com/LauraFJulia/TFT_vs_Fund`

[9] This value was optimally determined on a small subset of sequences (Penguin, Flowers, Pencils and Bag [3]). As for the remaining parameters of RPA (e.g. the number of sampled hypotheses), we used default values provided in the code by the authors.

[10] This choice is motivated by the fact that, in the presence of high corruption among the correspondences, one may not expect to classify *all* the points, as explained in [3]. Observe also that this error metric reports the fraction of wrong labelled data, that one wants to minimize in practice.

Table 1: Misclassification error [%] (the lower the better) and classified points [%] (the higher the better) for several methods on the data used in [3, 2]. The number of motions d, the number of images n, and the total number of image points p are also reported for each sequence. The best results are highlighted in boldface. *In this experiment all the correspondences are used.*

				TriSeg		TriPairSeg		Mode [3]		Synch [2]	
Dataset	d	n	p	Error	Classified	Error	Classified	Error	Classified	Error	Classified
Pen [2]	2	6	4550	**0.15**	60.51	0.55	79.56	0.58	80.07	0.82	83.23
Pouch [2]	2	6	4971	**1.07**	33.86	3.09	67.09	3.79	65.34	4.15	69.89
Needlecraft [2]	2	6	6617	**0.53**	45.40	0.84	73.76	0.83	72.81	1.04	76.76
Biscuits [2]	2	6	13158	**0.04**	63.59	0.35	85.72	0.47	84.47	0.51	87.28
Cups [2]	2	10	14664	**0.07**	50.31	0.49	66.37	0.56	65.42	1.01	69.82
Tea [2]	2	10	32612	**0.01**	61.69	0.23	82.37	0.29	81.70	28.12	52.21
Food [2]	2	10	36723	**0.01**	52.87	0.26	77.17	0.36	76.19	0.56	80.66
Penguin [3]	2	6	5865	0.75	34.31	**0.73**	69.70	0.76	69.17	44.21	46.97
Flowers [3]	2	6	7743	**0.05**	51.62	0.86	75.00	1.23	73.65	1.62	77.28
Pencils [3]	2	6	2982	5.04	35.28	**3.73**	65.56	3.80	65.33	27.53	40.44
Bag [3]	2	7	6114	1.40	40.97	**1.37**	64.26	1.52	57.95	25.92	54.27
Bears [3]	3	10	15888	**2.84**	41.21	4.38	74.31	4.82	73.65	38.95	74.59

Note that the latter fails in 5 cases. Concerning the amount of classified data, the best results are achieved by Synch in all the cases where it does not fail. The amount of point labelled by TriPairSeg is slightly better than Mode. The lowest amount is given by TriSeg, which, however, is not surprising: this method actually ignores all the points that have only one correspondence (points that are visible in 3 images are required to estimate the trifocal tensor).

In order to enrich the evaluation, we considered another scenario, reported in Tab. 2. Starting from the data used in [3, 2], the input correspondences were then filtered as follows: all the points that are matched in just one other image were removed. In other words, only correspondences involving (at least) 3 images were kept. In this way the performance of TriSeg remains unchanged in terms of misclassification error, but it is not penalized when counting the percentage of classified data. The output of the remaining methods, instead, generally improves. Observe that the lowest misclassification error is achieved either by TriSeg or TriPairSeg, outperforming the competing techniques, and Synch fails in 4 out of 12 cases. There is no significative difference between all the analysed methods in terms of amount of classified points in this experiment.

3.3 Novel benchmark

In order to study a more challenging scenario, we created a new dataset consisting of six indoor image collections with three or four motions. The benchmark created in [3, 2], instead, counts several sequences with two motions and only one sequence with three motions. Two-frame correspondences were obtained with SIFT [26] without any cleaning procedure, and ground-truth labels of image points were obtained by manual operations. More information about the dataset is provided in the supplementary material.

Table 2: Misclassification error [%] (the lower the better) and classified points [%] (the higher the better) for several methods on the data used in [3, 2]. The number of motions d, the number of images n, and the total number of image points p are also reported for each sequence. The best results are highlighted in boldface. *In this experiment all the trajectories of length two are removed.*

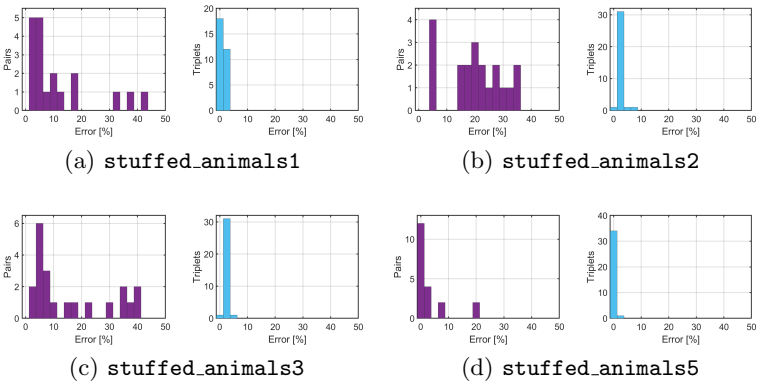
				TriSeg		TriPairSeg		Mode [3]		Synch [2]	
Dataset	d	n	p	Error	Classified	Error	Classified	Error	Classified	Error	Classified
Pen [2]	2	6	3208	**0.15**	83.98	0.17	88.29	0.24	88.77	0.42	93.47
Pouch [2]	2	6	2227	1.07	75.57	**0.65**	76.29	1.94	74.14	3.16	79.48
Needlecraft [2]	2	6	3733	0.53	80.47	**0.45**	82.67	0.56	81.94	1.21	88.51
Biscuits [2]	2	6	9306	0.04	89.91	**0**	91.91	0.07	91.15	0.20	94.87
Cups [2]	2	10	10452	**0.07**	70.58	0.21	78.10	0.26	77.58	0.84	83.31
Tea [2]	2	10	26134	**0.01**	76.98	0.09	88.22	0.15	88.02	24.48	63.08
Food [2]	2	10	27021	**0.01**	71.86	0.03	83.61	0.10	83.24	0.34	88.75
Penguin [3]	2	6	3035	0.75	66.29	**0.61**	81.58	0.73	81.29	35.29	51.73
Flowers [3]	2	6	4813	0.05	83.05	**0**	84.50	0.15	83.94	0.52	88.61
Pencils [3]	2	6	1424	5.04	73.88	**1.04**	74.44	1.58	75.49	34.72	45.72
Bag [3]	2	7	3108	1.40	80.60	**0.92**	80.57	1.24	72.52	2.10	82.82
Bears [3]	3	10	9998	2.84	65.48	**2.56**	81.73	3.08	81.13	29.08	65.18

Results are collected in Tab. 3, which reports the misclassification error and the percentage of classified data for all the analysed techniques. Synch presents poor performances on most cases, thus it is not a practical solution to motion segmentation, confirming the outcome of the experiments in Sec. 3.2. Some interesting observations can be made about the remaining methods, which share the same framework but they are based on different models: Mode uses the fundamental matrix; TriSeg uses the trifocal tensor; TriPairSeg uses both the fundamental matrix and the trifocal tensor. Using only the fundamental matrix as underlying model is not enough to segment the most difficult scenes. This aspect can be appreciated from the poor performance of Mode on the sequences with four motions. It is remarkable that TriSeg achieves very good results on all the sequences, outperforming all the analysed techniques. This is due to the usage of the trifocal tensor, which constitutes a stronger model than the fundamental matrix. The percentage of points classified by TriSeg is, in general, lower than the other approaches but it is still acceptable. TriPairSeg achieves reasonably good results (although not comparable to TriSeg) and it is better than Mode both in terms of misclassification error and amount of classified points. Note that TriPairSeg inherits the advantages of both models: on one side, it provides a good segmentation thanks to the presence of the trifocal tensor; on the other side, it classifies a high amount of points thanks to the presence of the fundamental matrix, which does not discard trajectories of length two.

The fact that the trifocal tensor usually provides a better segmentation than the fundamental matrix can also be appreciated in Fig. 5, which shows the distribution of the misclassification error achieved by RPA on all the triplets/pairs. This gives an idea about the quality of the input partial segmentations used by different approaches. It is clear that those produced by trifocal tensor fitting are

Table 3: Misclassification error [%] (the lower the better) and classified points [%] (the higher the better) 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 for each sequence. The best results are highlighted in boldface.

				TriSeg		TriPairSeg		Mode [3]		Synch [2]	
Dataset	d	n	p	Error	Classified	Error	Classified	Error	Classified	Error	Classified
`stuffed_animals1`	4	7	11507	**1.82**	64.99	4.73	86.95	8.92	83.98	40.79	56.88
`stuffed_animals2`	4	7	11159	**3.05**	61.78	6.92	83.87	17.56	79.29	9.96	49.95
`stuffed_animals3`	4	7	10989	**2.20**	59.07	7.39	83.78	15.03	81.49	27.58	73.53
`stuffed_animals4`	4	7	8079	**3.33**	55.80	6.96	78.79	13.37	76.57	32.97	62.02
`stuffed_animals5`	3	7	13851	**0.72**	60.19	2.00	88.10	2.56	87.04	2.20	89.26
`stuffed_animals6`	3	7	12170	**0.60**	58.89	4.68	84.77	5.43	84.80	15.60	65.46



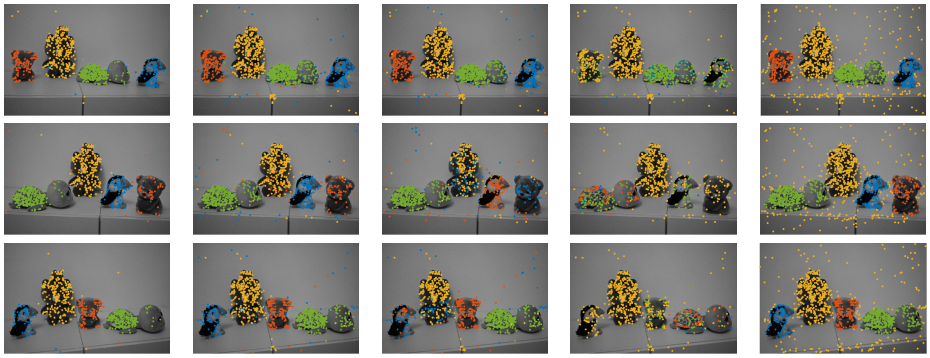
(a) `stuffed_animals1` (b) `stuffed_animals2`

(c) `stuffed_animals3` (d) `stuffed_animals5`

Fig. 5: Histograms of misclassification error achieved by RPA on sample sequences from our dataset. Each point in the horizontal axis corresponds to a possible misclassification error in an individual pair/triplet of images. Each point in the vertical axis corresponds to the number of pairs/triplets where a given error is reached.

the most accurate, since the blu light histograms are concentrated to the left. Those produced by fundamental matrix fitting, instead, are very noisy: note that RPA can even reach a misclassification error larger than 30% in some pairs related to `stuffed_animals1` (see the purple histogram in Fig.5a).

Some qualitative results are reported in Fig. 6 and further analysis is given in the supplementary material. Note that both Synch and Mode presents difficulties in segmenting this scene: the former produces useless results whereas the latter switches two motions in the middle image. TriSeg and TriPairSeg, instead, report good performances.



(a) TriSeg (b) TriPairSeg (c) Mode [3] (d) Synch [2] (e) Ground-truth

Fig. 6: Segmentation results are reported on sample images from `stuffed_animals4` for several methods. Different colours correspond to different motions. For better visualization, unclassified points are not drawn. Ground-truth segmentation is also reported.

4 Conclusion

We presented two novel solutions to motion segmentation that combine local results independently obtained from subsets of images: TriSeg considers triplets of images whereas TriPairSeg considers both triplets and image pairs. In order to tackle segmentation in a triplet, multiple trifocal tensors were fitted to correspondences via robust preference analysis. The usage of the trifocal tensor within motion segmentation was the key to success for our methods, for it is more robust to wrong correspondences than the fundamental matrix. The proposed solutions outperform previous techniques on existing datasets as well as on a new image collection, and they can handle scenes with four motions. The choice of one method between TriSeg– which classifies less points with higher accuracy – and TriPairSeg depends on the task and is left to the reader.

Acknowledgements. This research was supported by the European Regional Development Fund under IMPACT No. CZ.02.1.01/0.0/0.0/15 003/0000468, R4I 4.0 No. CZ.02.1.01/0.0/0.0/15 003/0000470, EU H2020 ARtwin No. 856994, and EU H2020 SPRING No. 871245 Projects.

References

1. Arrigoni, F., Fusiello, A.: Synchronization problems in computer vision with closed-form solutions. International Journal of Computer Vision **128**, 26–52 (2020)
2. Arrigoni, F., Pajdla, T.: Motion segmentation via synchronization. In: IEEE International Conference on Computer Vision Workshops (ICCVW) (2019)

3. Arrigoni, F., Pajdla, T.: Robust motion segmentation from pairwise matches. In: Proceedings of the International Conference on Computer Vision (2019)
4. Barath, D., Matas, J.: Multi-class model fitting by energy minimization and mode-seeking. In: Proceedings of the European Conference on Computer Vision. pp. 229–245. Springer International Publishing (2018)
5. Chin, T.J., Suter, D., Wang, H.: Multi-structure model selection via kernel optimisation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3586–3593 (2010)
6. Delong, A., Osokin, A., Isack, H.N., Boykov, Y.: Fast approximate energy minimization with label costs. International Journal of Computer Vision **96**(1), 1–27 (2012)
7. Elhamifar, E., Vidal, R.: Sparse subspace clustering: Algorithm, theory, and applications. IEEE Transactions on Pattern Analysis and Machine Intelligence **35**(11), 2765–2781 (2013)
8. Fischler, M., Bolles, R.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Morgan Kaufmann Readings Series **24**, 726–740 (1987)
9. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the KITTI vision benchmark suite. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2012)
10. Hartley, R., Vidal, R.: The multibody trifocal tensor: motion segmentation from 3 perspective views. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. vol. 1, pp. I–769–I–775 Vol.1 (June 2004). https://doi.org/10.1109/CVPR.2004.1315109
11. Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press, second edn. (2004)
12. Hartley, R.: Lines and Points in Three Views and the Trifocal Tensor. International Journal of Computer Vision **22**(2), 125–140 (1997)
13. Holland, P.W., Welsch, R.E.: Robust regression using iteratively reweighted least-squares. Communications in Statistics - Theory and Methods **6**(9), 813–827 (1977)
14. Isack, H., Boykov, Y.: Energy-based geometric multi-model fitting. International Journal of Computer Vision **97**(2), 123–147 (2012)
15. Ji, P., Li, H., Salzmann, M., Dai, Y.: Robust motion segmentation with unknown correspondences. In: Proceedings of the European Conference on Computer Vision. pp. 204–219. Springer International Publishing (2014)
16. Ji, P., Li, H., Salzmann, M., Zhong, Y.: Robust multi-body feature tracker: A segmentation-free approach. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2016)
17. Ji, P., Salzmann, M., Li, H.: Shape interaction matrix revisited and robustified: Efficient subspace clustering with corrupted and incomplete data. In: Proceedings of the International Conference on Computer Vision. pp. 4687–4695 (2015)
18. Julià, L.F., Monasse, P.: A critical review of the trifocal tensor estimation. In: Image and Video Technology. pp. 337–349. Springer International Publishing (2018)
19. Kim, J.B., Kim, H.J.: Efficient region-based motion segmentation for a video monitoring system. Pattern Recognition Letters **24**(1), 113 – 128 (2003)
20. Kuang, D., Yun, S., Park, H.: SymNMF: nonnegative low-rank approximation of a similarity matrix for graph clustering. Journal of Global Optimization pp. 1–30 (2014)
21. Kuhn, H.W.: The Hungarian method for the assignment problem. Naval Research Logistics Quarterly 2 **2**, 83 – 97 (1955)

22. Lai, T., Wang, H., Yan, Y., Chin, T.J., Zhao, W.L.: Motion segmentation via a sparsity constraint. IEEE Transactions on Intelligent Transportation Systems **18**(4), 973–983 (2017)
23. Li, Z., Guo, J., Cheong, L.F., Zhou, S.Z.: Perspective motion segmentation via collaborative clustering. In: Proceedings of the International Conference on Computer Vision. pp. 1369–1376 (2013)
24. Lin, Z., Chen, M., Ma, Y.: The augmented lagrange multiplier method for exact recovery of corrupted Low-Rank matrices. eprint arXiv:1009.5055 (2010)
25. Liu, G., Lin, Z., Yan, S., Sun, J., Yu, Y., Ma, Y.: Robust recovery of subspace structures by low-rank representation. IEEE Transactions on Pattern Analysis and Machine Intelligence pp. 171–184 (2013)
26. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision **60**(2), 91–110 (2004). https://doi.org/http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94
27. Magri, L., Fusiello, A.: Robust multiple model fitting with preference analysis and low-rank approximation. In: Proceedings of the British Machine Vision Conference. pp. 20.1–20.12. BMVA Press (September 2015)
28. Magri, L., Fusiello, A.: Multiple models fitting as a set coverage problem. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3318–3326 (June 2016)
29. Olsson, C., Enqvist, O.: Stable structure from motion for unordered image collections. In: Proceedings of the 17th Scandinavian conference on Image analysis (SCIA'11). pp. 524–535. Springer-Verlag (2011)
30. Ozden, K.E., Schindler, K., Van Gool, L.: Multibody structure-from-motion in practice. IEEE Transactions on Pattern Analysis and Machine Intelligence **32**(6), 1134–1141 (2010)
31. Pachauri, D., Kondor, R., Singh, V.: Solving the multi-way matching problem by permutation synchronization. In: Advances in Neural Information Processing Systems 26, pp. 1860–1868. Curran Associates, Inc. (2013)
32. Pavan, A., Tangwongsan, K., Tirthapura, S., Wu, K.L.: Counting and sampling triangles from a graph stream. Proceedings of the VLDB Endowment **6**(14), 1870–1881 (2013)
33. Rao, S., Tron, R., Vidal, R., Ma, Y.: Motion segmentation in the presence of outlying, incomplete, or corrupted trajectories. Pattern Analysis and Machine Intelligence **32**(10), 1832–1845 (2010)
34. Rubino, C., Del Bue, A., Chin, T.J.: Practical motion segmentation for urban street view scenes. In: Proceedings of the IEEE International Conference on Robotics and Automation (2018)
35. Sabzevari, R., Scaramuzza, D.: Monocular simultaneous multi-body motion segmentation and reconstruction from perspective views. In: Proceedings of the IEEE International Conference on Robotics and Automation. pp. 23–30 (2014)
36. Saputra, M.R.U., Markham, A., Trigoni, N.: Visual SLAM and structure from motion in dynamic environments: A survey. ACM Computing Surveys **51**(2), 37:1–37:36 (2018)
37. Schindler, K., Suter, D., Wang, H.: A model-selection framework for multibody structure-and-motion of image sequences. International Journal of Computer Vision **79**(2), 159–177 (2008)
38. Shen, Y., Huang, Q., Srebro, N., Sanghavi, S.: Normalized spectral map synchronization. In: Advances in Neural Information Processing Systems 29, pp. 4925–4933. Curran Associates, Inc. (2016)

39. Toldo, R., Fusiello, A.: Robust multiple structures estimation with J-Linkage. In: Proceedings of the European Conference on Computer Vision. pp. 537–547 (2008)
40. Torr, P.H.S., Zisserman, A.: Concerning bayesian motion segmentation, model averaging, matching and the trifocal tensor. In: Proceedings of the European Conference on Computer Vision. pp. 511–527. Springer Berlin Heidelberg (1998)
41. Torr, P.H.S., Zisserman, A., Murray, D.W.: Motion clustering using the trilinear constraint over three views. In: Europe-China Workshop on Geometric Modellling and Invariants for Computer Vision. pp. 118–125. Springer (1995)
42. Tron, R., Vidal, R.: A benchmark for the comparison of 3-d motion segmentation algorithms. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1–8. IEEE (2007)
43. Tron, R., Zhou, X., Esteves, C., Daniilidis, K.: Fast multi-image matching via density-based clustering. In: Proceedings of the International Conference on Computer Vision. pp. 4077–4086 (2017)
44. Vidal, R., Ma, Y., Sastry, S.: Generalized principal component analysis (gpca). IEEE Trans. Pattern Anal. Mach. Intell. **27**(12), 1945–1959 (2005)
45. Von Luxburg, U.: A tutorial on spectral clustering. Statistics and computing **17**(4), 395–416 (2007)
46. Wang, Y., Liu, Y., Blasch, E., Ling, H.: Simultaneous trajectory association and clustering for motion segmentation. IEEE Signal Processing Letters **25**(1), 145–149 (2018)
47. Xu, X., Cheong, L.F., Li, Z.: Motion segmentation by exploiting complementary geometric models. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2859–2867 (2018)
48. Yan, J., Pollefeys, M.: A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and nondegenerate. In: Proceedings of the European Conference on Computer Vision. pp. 94–106 (2006)