Spectral Methods for 3-D Motion Segmentation of Sparse Scene-Flow

Diana Mateus * Radu Horaud
PERCEPTION group, INRIA Rhône-Alpes,France
{ mateus, horaud@inrialpes.fr }

Abstract

The progress in the acquisition of 3-D data from multi-camera set-ups has opened the way to a new way of loking at motion analysis. This paper proposes a solution to the motion segmentation in the context of sparse scene flow. In particular, our interest focuses on the disassociation of motions belonging to different rigid objects, starting from the 3-D trajectories of features lying on their surfaces. We analyze these trajectories and propose a representation suitable for defining robust-pairwise similarity measures between trajectories and handling missing data. The motion segmentation is treated as graph multi-cut problem, and solved with spectral clustering techniques (two algorithms are presented). Experiments are done over simulated and real data in the form of sparse scene-flow; we also evaluate the results on trajectories from motion capture data. A discussion is provided on the results for each algorithm, the parameters and the possible use of these results in motion analysis.

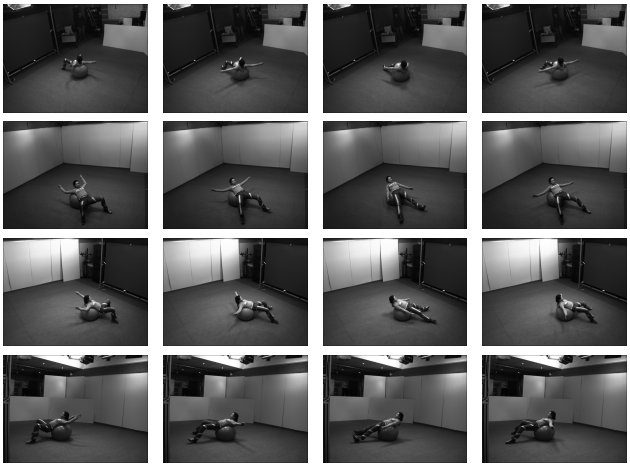

Figure 1: One of the SFT sequences. Each row correspond to a different point of view and each column to a different time step.

1. Introduction

The analysis and understanding of complex object motion from image sequences constitutes a powerful source of information. The problems of capturing, modeling, and rendering motion observations are relevant to an increasing number of applications such as video surveillance, robot and autonomous vehicle navigation, video post-processing, and so forth. The vast majority of motion-segmentation approaches employ 2-D motion observations, such as *optical flow*. It is only recently that interest has shifted towards 3-D motion observations like *scene flow*. In this paper, we address the problem of clustering sparse scene flow (i.e., 3-D trajectories of sparse features that are tracked over time) into several groups, each representing an object undergoing rigid motion. We will refer to this problem as 3-D *motion segmentation*.

Existing motion segmentation techniques fall into one of the following categories: frame-to-frame methods,multiple-frame methods, and clustering methods.

Frame-to-Frame methods: They use *optical flow* as input. There exists a broad list of algorithms for motion estimation from a single image sequence, based on finding flow discontinuities, layered representations or on fitting mixture models. However this remains a difficult problem due to the ambiguities inherent to 2-D velocity fields (i.e. occlusions and transparencies). More recently, the use of multi-camera geometry lead to approaches based on the multiple-body epipolar constraint [16]. In [15], Vidal and Ma have recently proposed an algebraic approach for 2-D and 3-D motion segmentation based on two-view point correspondences or optical flow; its purpose is to fit a single multi-body motion model to the image measurements, which one can derivate to obtain the motion of a particular object.

Multiple-frame methods: In the more general approach, tracking of sparse 2-D features over a long image sequence is followed by multi-body factorization algorithms. Trajectories of image features are stacked and used to build a *Shape interaction matrix SIM*. Once in its canonical form, the SIM gives the segmentation of features into rigid objects. Various algorithms have been proposed to solve the segmentation from the SIM, such as permutations [2], QR iterative diagonalization [6], etc. These methods heavily depend on a reliable 2-D feature tracker which is a difficult and unsolved problem in itself.

Clustering methods: Motion segmentation can also be

*D. Mateus is supported by a grant from the European Community under the EST Marie-Curie Project Visitor.

viewed as a clustering problem: the goal is to assign a unique object label to features undergoing the same rigid motion. Some methods overlap with the previous category [7], since they use the SIM as a pairwise similarity matrix as input to the clustering step. Others ([4, 8, 17, 11]), use spectral mappings on the features before performing the clustering. The method proposed in this paper belongs to this category. In [13], the clustering problem is addressed differently. First 3-D points are obtained at each time instant using a stereoscopic technique. A 3-D rotation is computed from matched triplets of points and represented in a 3-parameter space. Clustering is performed using the mean-shift algorithm on this space. The main drawback of this method is that many artifact motions are generated because there is no simple way to associated triplets of points. It is therefore possible to find a rigid motion between two sets of points even if they do not belong to the same object.

1.1. Methodology of the proposed solution

In this work we use data gathered with a multiple-camera system, see Fig1. A set of interest points are reconstructed in 3-D and tracked along a certain period of time. Each tracked feature is represented as a 3-D trajectory describing the 3-D position of each feature in a given frame. The set of these trajectories will be refered to as *sparse scene flow*. Although the problem of motion segmentation has been thoroughly studied in computer vision, there aren't many methods that work directly onto 3-D trajectories, probably because of the difficulty to accurately recover scene flow. The originality of our work resides on addressing the motion segmentation problem in the 3-D domain, taking as input multiple-frame scene-centered point trajectories.

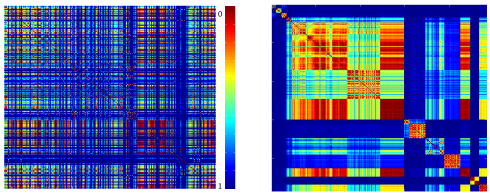

Figure 2: Methodology: From the scene-flow trajectories (see Fig.1), a) Build a similarity matrix from pairwise distances between trajectories. b)Embed and cluster: the ordered similarity matrix after the clustering is finished. Color corresponds to similarity values. A successful clustering should permute the input matrix to a block diagonal one

In order to group trajectories following the same motion, it is important to describe a pairwise distance measures and a space where the comparison is feasible. This measure should make easy to distinguish when a couple of trajectories are under the effects of the same motion, i.e. when the trajectories describe the motion of two features attached to the same rigid object. According to the rigidity constraint, two points on a rigid object will allways preserve an constant Euclidean distance. We use this fact to define a distance measure between every pair of trajectories, where a null distance stands for features rigidly linked and a large distance corresponds to independently moving features.

In our approach, the multi-body motion segmentation is treated as a clustering problem; we propose the use of spectral clustering techniques to solve it. The motivation of this choice is the lifespan of the features: they may be initialized at any frame, and can be lost when occlusions occur or when the incertitude in tracking has grown to much. As a consequence, trajectories can not be easily represented in a vector space. Fortunately the spectral mapping methods are able to construct a proper embedding for clustering for any data whenever there is an adequate similarity pairwise distance defined between features.The two proposed algorithms are capable of determining the number of rigid objects moving independently in the scene and labeling them accordingly

Our method differs from previous motion segmentation approaches using spectral mappings in both the type of data used to cluster and the measure of similarity for embedding it. We define a robust similarity measure from the rigidity pairwise distance. The proposed solution has been tested with three different datasets: simulated data, motion capture data, and scene-flow data. Results show that our motion segmentation method performs well with indepedently moving objects and gives interesting results for articualted objects such as humans.

Paper organization The remainder of this paper is organized as follows: Section 2 explains how to obtain pairwise measurements between trajectories and how to construct an appropriate similarity matrix. We also provide an interpretation of the motion segmentation problem from the viewpoint of graph clustering. Section 3 overviews a spectral clustering theory and methods and gives insights on how to apply them to motion segmentation. It presents the proposed algorithms algorithms and the estimation of the similarity scale parameter. In section 4 we describe the experimental data sets and the quality measures used for evaluation, and present the relevant results. The paper concludes with a discussion on the results in Section 5.

2. Similarity between Scene-flow Trajectories

Assume a scene observed by several cameras, where K rigid objects are moving freely. At the first frame, N feature points $p_1, p_2 ... p_N$, $p_i \in \Re^3$ are detected on the surface of the objects and tracked during at-most F frames. Stacking the scene vectors flows from a frame to the next we build, the trajectory of each feature during the sequence is reconstructed.

Let S be the set of N trajectories $S = \{s_1, s_2, ..., s_N\}$ in a time sequence of F frames. The content of the i^{th} trajec-

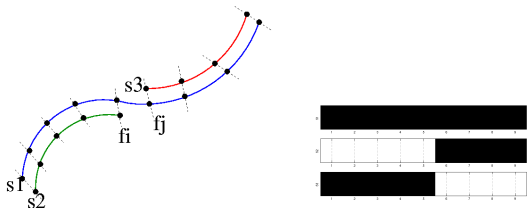

Figure 3: Three trajectories s_1, $s2$ and $s3$ and their mask. The distance between s_2 and s_3 is set to ∞ since lifespans dont intersect

tory at a particular time frame f, $s_i^f \in \Re^3$, corresponds to the 3-D global coordinates of the point p_i at frame f. Since at any time frame, features can be added or lost, a binary *activity mask* $m_i \in [0,1]$ is attached to each trajectory to indicate those frames where trajectory is active. We define the *lifespan* of a feature as the set of frames during which it is active. See Fig.3.

For the clustering purposes, we must define a pairwise measure of the rigidity. For each pair of trajectories s_i and s_j, we calculate the euclidean distance between their coordinates $e^f = \| s_i^f - s_j^f \|$ at every time frame f; as a result we have a vector of distances $\vec{e}$ of length F. Since two points following the same rigid motion preserve a constant distance, examining how the elements of $\vec{e}$ vary along the sequence will allow us to determine if s_i and s_j are points in the surface of the same rigid object. In consequence, we propose the *variance* of the $\vec{e}$ vector as rigidity distance measure $d(s_i, s_j)$, considering that the distance $e_{i,j}^f$ is only calculated for active entries ($m_i^f = 1$), and between intersecting trajectories (see eq.1). While the intersection exists, the measure is metric.

$$e^f(s_i, s_j) = \begin{cases} \| s_i^f - s_j^f \| & \text{if } m_i^f m_j^f = 1 \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

$$d(s_i, s_j) = \begin{cases} var(\vec{e}) & \text{if } \sum_{f=1}^{F} m_i^f m_j^f \neq 0 \\ \infty & \text{otherwise} \end{cases} \quad (2)$$

2.1. Graph interpretation

Using the distance measure, the overall problem can be represented with a completely connected graph with undirected weighted links. A node corresponds to a trajectory s and weights $w_{ij} \in [0,1]$ show the degree in which the motion between trajectories is similar. Gathering these pairwise distances we form the adjacency matrix of the graph (also called affinity or similarity Eq.3). For groups with clear independent motions, the symmetric adjacency matrix associated to the graph is a permutation of a diagonal block matrix.

$$W = [w_{ij}], \text{ with} \quad (3)$$

$$w_{ij} = exp\Big(\frac{-d(s_i, s_j)^2}{2\sigma}\Big) \quad (4)$$

The σ is scale parameter that must be set carefully. We describe in section 3.3 how to get a proper estimation according to the data set.

As mentioned before, the problem of motion segmentation can be interpreted as the clustering of the trajectories in K groups. Spectral methods are particulary well suited to solve the clustering problem since they only need a pairwise similarity matrix to separate groups of features.

Motion segmentation methods based on spectral clustering techniques usually employ the SIM matrix as similarity matrix, i.e [4] [8]. In [17] two similarity measures are proposed; the first, based on SIM, attempts to enhance the performance by cutting-off at once several highly "sensitive" weights. The second measure is directly built from optical-flow displacements and fails, by definition, under rotations. Furthermore, it depends on the magnitude and duration of the motion, giving higher affinity values to features that move fast and that last long. Park [11] uses the QR algorithm instead of the SVD for building the shape interaction matrix (invariant to both the object motions and the selection of coordinate systems). It also gives confidence levels to the membership of each feature.

3. Overview of Spectral theory and Spectral Clustering

Spectral clustering methods belong to the family of mode analysis techniques which apply eigenbased mappings of the data onto spaces where a desired aspect of data can be easily highlighted. In particular, they are capable of grouping features based only on pre-established pairwise measures of similarity between them. The measures do not have to be necessarily metric, opening the solution to applications with non-linear or complex spaces. They are particularly useful for problems that do not accept the assumption of clusters with gaussian-shape, since they do not require explicitly estimating the model of data distribution. They also support features in infinite-spaces (as long as a measure of similarity can be established). They are called spectral because they rely upon the analysis of the spectrum of the similarity matrix. Several operations are applied to this input matrix, including at some point an eigen decomposition step. The searched embedding is made from a selection of the transformed eigenvectors. If an appropriate similarity measure has been properly chosen all the features members of a given cluster will be mapped to a single point.

The sparse scene-flow trajectories can not be easily represented in a vector space, due to their lifespans. Even if

they were defined all over the sequence, the trajectory space itself would not be meaningful, since the spatial proximity does not necessarily mean motion similarity. For these reasons we rule out PCA-like embeddings, and instead prefer spectral mappings. Other embeddings as local linear embedding or multidimensional scaling could also be used.

Spectral graph theory relies on the analysis of the adjacency matrix W of a graph, although more often is the graph's *Laplacian matrix* $\mathcal{L}$ which is considered for stability reasons [1, 14]. There exists several definitions of the $\mathcal{L}$, we have chosen to work with a definition that gives a good numerical stability [14]. Defining a diagonal matrix D, with each diagonal entry d_{ii} corresponding to the *degree* of a node, the *Laplacian* is described by : $\mathcal{L} = D^{-1/2}WD^{-1/2}$ and its eigendecomposition takes one of the following forms(with $u = D^{1/2}v$):

$$D^{-1/2}WD^{-1/2}u = \lambda u \quad (5)$$

$$Wv = \lambda Dv \quad (6)$$

$\mathcal{L}$, has the properties of being symmetric and semidefinite positve; therefore, its eigen-decomposition leads to real eigenvalues and a complete set of orthonormal eigenvectors. Spectral clustering use this properties together with the fact that the spectrum of a graph G is the union or the spectra of its connected components.

The ideal graph for segementation should have a series of connected components with no link between them. The eigenvectors would form an eigenspace of geometric multiplicity equal to the number of groups, and clustering in this space should be straightforward. In the real case, the Laplacian corresponds to a connected graph from which we want to extract the stronger connected components. The largest eigenvalue λ_0 is not even guarantee to be multiple so, the most we can expect is to obtain eigenvectors that are close to being piecewise constant [9]. We can see eigenvectors as functions assigning values to each feature. In a bi-partition problem, this means eigenvectors v assigning different values a and b to features in different clusters: $v(i) \approx a$ if $i \in A$ and $v(i) \approx b$ if $i \in B$. This can be extended to a multiple partition by watching the embedding; features in the same cluster should be closely mapped (ideally to the same point). The main difference between algorithms rely on the post-processing of these eigenvectors, in order to enhance the piecewise condition and the method used to obtain the labeling.

3.1. Spectral Clustering algorithms

Early approaches tried to iteratively split features in two clusters, according to a particular eigenvector (i.e. the greatest of W [4], the second smallest generalized eigenvector of $\mathcal{L}'$ [12]). For more than two clusters, the algorithm could be applied iteratively.

The first intent to give a good mathematical justification to these techniques is the Shi and Malik's work [12] within an image segmentation application; in particular, they show that λ_1 is an approximated solution to the optimization of the normalized-cut criterion for clustering. See [18] for a comparison of this preliminary algorithms.

Most of current approaches, could be interpreted as an spectral embedding of the features in a space where standard low-cost clustering algorithms can take place; in other words, this algorithms consider and transform a whole set of eigenvectors of interest. Under good conditions, the algorithms acting on several eigenvectors at the same time, should encounter that the first k eigenvectors of their system are (almost) piecewise constant. The well-known Ng, Jordan and Weiss' work [10] being a pillar. For a practical comparison of spectral clustering methods see [14]. Later, the link with the stochastic processes was uncovered [9]. There are several proposals to enhance the initial results, notably by adding pre-processing tasks improving the similarity matrix ([1, 17]). There are also some publications on how this techniques are related to other eigen-like clustering procedures (i.e, Kernel-PCA) and embedding methods.

3.2. Clustering trajectories with the spectral methods

We propose two algorithms to carry out the spectral clustering of our 3-D trajectories.The first follows the embedding form of [10] but using the stable eigen-decomposition from [14]. It works in a hierarchical way for searching different degrees of correlation between the data, as in the case of articulated motion . The second one takes an iterative form as in [12] but evaluating the MNCut (Multiway Normalized Cut) [14] quality measure at each iteration.

Algorithm I

1. Calculate all the $d_{i,j}$ for each pair of trajectories eq.(1).
2. Estimate the scale parameter σ.
3. Do the eigen-decomposition of the generalized system $Wv = \lambda Dv$.
4. Find K the number of cluster to be found, using a coherence measure.
5. Select the K eigenvectors corresponding to the greatest eigenvectors and pile them to form the columns of the matrix Y.
6. Apply k-means clustering on the normalized rows of the matrix Y
7. Repeat the procedure from step 2. for each cluster with correlated data

Algorithm II

1. - 3. as in Algorithm I
4. Find $u_1 = D^{1/2}v_1$, verify the stability of the vector and split the trajectories in two groups accordingly. The split point is the midpoint between two values of u_1 minimizing the Multiway Ncut criterion
5. Repeat until a maximum $\frac{MNCut}{K_i}$ is achieved, where K_i stands for the number of clusters found so far.

The MNCut criterion, measures the quality of the cluster (for a given K)[14], under the principle that a good clustering should be able to maximize the intra-similarity for each group while maximizing the dissimilarity between groups. The normalization by the volume [1] avoids the measure to give a preference to small sets of isolated nodes. For a graph C with a a clustering (Δ) of K groups: $\Delta = \{C_1, C_2, ..., C_K\}$ the MNCut criterion is defined as:

$$MNCut(\Delta) = \sum_{k=1}^{K}\sum_{l=1}^{K} ncut(C_k, C_l) \quad (7)$$

$$ncut(C_k, C_l) = \frac{cut(C_k, C_l)}{vol(C_k, C)} + \frac{cut(C_k, C_l)}{vol(C_l, C)} \quad (8)$$

$$cut(C_k, C_l) = \sum_{i \in C_k, j \in C_l} w(i, j) \quad (9)$$

Since MNCut is not a suitable measure for comparing clusterings of different K (as it grows with K, it always gives preference to small clusterings), another criterion needs to be considered to choose the dimension of the embedding K. Our measure is one of the [14] *coherence* criteria. It is based on the distance to the lower bound $l(K)$ of MNCut for a given K (see eq.10) and takes into account the fact that when noise grows, the distance to the bound $l(K)$ also tends to increase. For a given clustering Δ:

$$coh(\Delta) = gap(\Delta)/l(\mid \Delta \mid) \quad (10)$$

$$gap(\Delta) = MNCut(\Delta) - l(K) \quad if \mid \Delta \mid = K \quad (11)$$

$$l(K) = K - \sum_{k=1}^{K} \lambda_k \quad (12)$$

Algorithm II, needs several considerations. First, the splitting point of the λ_1 vector. We order the elements of v_i and test splitting points in between each pair of values. The Ncut criterion is used to compare and choose the best clustering. Second, the stop criterion for the iteration. A max $MNCut/K$ criterion is established, this parameter is not really sensitive and a big value can be set if no a-priori information of the level of segmentation is available. For safety we use the histogram-based stability criterion [12] to avoid from splitting clusters from eigenvectors far from being piecewise constant.

[1] The volume of a cluster k is $vol(C_k) = \sum_{i \in C_k} d_i$.

3.3. Estimating the scale parameter

We attach a significant importance to the scale parameter, since in our experiments we have found that motion from real objects can have different degrees of correlation.

We propose to use of two scale definitions. The first one based on the Huber robust measure of scale (See Eq.15). In the presence of noise there maybe outliers that we attempt to reject when estimating our final σ by considering only the standart deviation of those d_{ij} under $3\sigma_{mad}$.

$$\sigma_{mad} = 1.4826 mad(w_{ij}) \quad (13)$$

$$= 1.4826 med(\parallel w_{ij} - med(w_{ij}) \parallel) \quad (14)$$

$$\sigma = \sqrt{var(w_{ij})} \quad \forall w_{ij} < 3\sigma_{mad} \quad (15)$$

Fischer [5] and Zelnik [19] have proposed local scaling for adapting the scale to a neighborhood. We use the [19] definition, where the similarity matrix is built from $\sigma' = \sigma_i \sigma_j$, with $\sigma_i = d(s_i, s_{knn})$, where knn stands for the k^{th} nearest neighbor of the trajectory i; plus a step to recover the symmetry. We establish the neighborhood as a percentage of the number of input trajectories (10% for the experiments presented in the paper). σ' is able to uncover the rigid motion of articulated parts but, it dependes on the chosen number of the neighbors. An exahustive search over σ may give better results but is expensive and ours is satisfactory enough for highlighting similar motions. In the next section, we show the performance of our algorithms over different datasets.

4. Experimental Evaluation

Experimental evaluation was performed over three different kinds of datasets. The first one, that we called *SIMIL*, is an entirely simulated dataset, consisting on trajectories of groups of scattered points following randomly-changing rigid transformations. It contains cases of independent and/or articulated groups of points, motion is corrupted with different levels of noise and missing data. *ARTI*, the second dataset is based on motion capture data, mainly of human bodies. Trajectories correspond to reconstructed 3-D motion obtained from image features matched over several cameras. The final data set *SFT* (Scene-Flow Tracker), it was obtained by processing real video sequences from a multi-camera calibrated system, observing scenes where different objects moved (including rigid objets and people) and tracking features in 3-D [3] (See Fig. 5 for a quick view of the data acquisition method)

Clustering evaluation w.r.t ground truth is possible for the SIMIL and ARTI datasets. We do not have real labeling for the SFT data set, it is specially difficult to build given

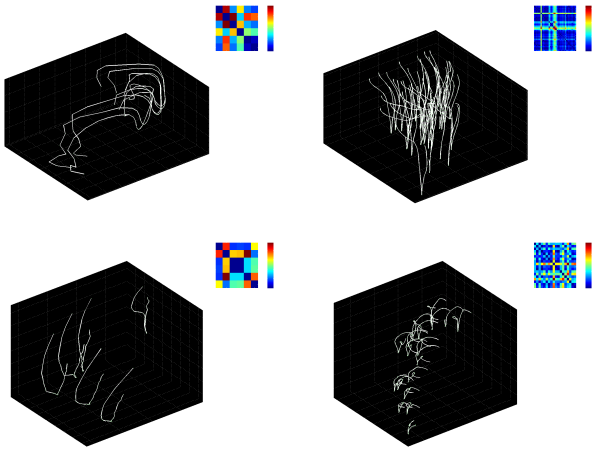

Figure 4: Three of the 17 splitted clusters of the input sequence of Fig.1.The small matrices stand for the similarity matrices inside the cluster. Motion inside each cluster is indeed coherent.

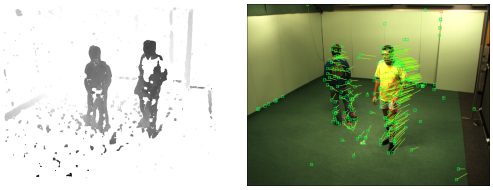

Figure 5: Obtaining real sparse scene-flow trajectories SFT. a) Feature 3-D points are initialized from a stereo reconstruction. b) 3-D tracking (projection of scene-flow)

the incremental uncertainty with time and the fact that 3-D features may switch from one cluster to another during the sequence.

We use the *Jaccard* and the variation of information criteria [14] to evaluate de performance of our algorithms, when ground truth is available to compare with.

$$
\begin{aligned}
VI(\Delta,\Delta') &= \mathcal{H}(\Delta)+\mathcal{H}(\Delta')+2\mathcal{I}(\Delta,\Delta') \\
\mathcal{H}(\Delta) &= \sum_{k=1}^{K}\frac{n_k}{N}log\left(\frac{n_k}{N}\right) \\
\mathcal{I}(\Delta,\Delta') &= \sum_{k=1}^{K}\sum_{k'=1}^{K'}\frac{\mid C_k\cap C_{k'}\mid}{n_k n_{k'}}log\left(\frac{\mid C_k\cap C_{k'}\mid}{n_k n_{k'}}\right) \\
\mathcal{J} &= \frac{n_{11}}{2N(2N-1/2)-n_{00}}
\end{aligned}
$$

where $\mathcal{H}$ stands for the entropy and $\mathcal{I}$ for the mutual information. In the Jaccard coefficient n_{11}corresponds to the number of pairs of features classified together in the true and tested clusterings; analogously n_{00}, for the pairs assigned to different clusters in each case. $MNCut(\Delta)$ and $coh(\Delta)$. serve as quality measures for data with no ground truth.

dataset	alg	scale	K	VI	J	MNCut
ARTI	I	σ	21	2.68	0.11	
		σ'	12	2.46	0.13	
	II	σ	32	2.80	0.11	
		σ'	17	3.05	0.18	
SFT	I	σ	3	0.067	0.9718	-
		σ'	5	0.5397	0.6299	0.6595
	II	σ	14	1.1701	0.600	0.6
		σ'	4	1.1356	0.3613	0

Table 1: Evaluation results for two sequences.

Each dataset consisted of around 10 different acquisitions/simulations. The clusterings obtained with our methods satisfactorily corresponded to what we were expecting from the distance measure definition. Misclassifications rarely occurred, but over and sub-segmentation happen. For the SIMI dataset, the results are really good, with both algorithms and definitions of sigma, even for high levels of missing data.

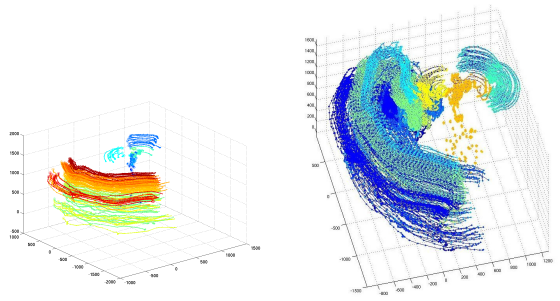

Figure 6: ARTI dataset with two humans in the scene. Different colors correspond to different labels. a) Ground truth b) Results using the algorithm I. Despite there exists a high correlation between body parts, they are correctly splitted appart

The ARTI dataset served to test the algorithm when data is correlated. The definition of the distance and similarity was crucial for uncovering the parts or the articulated body. When more than one independent rigid object body was present, algorithm-I tended to recover only the rigid bodies, while algorithm-II directly explored and clustered inside the correlated data. The numerical evaluation of the ARTI is only approximate, because the ground truth labeling is based on body-parts and not on motion (if a part of an articulated body does not move, the clustering algorithm can not recognize it as an independent group). See Fig.6 for an example of the results obtained with this dataset.

The SFT dataset is highly corrupted by missing data and outliers so its here were the robust measure of scale is really helpful. Both algorithms based on σ do well, but the

ones with σ' sometimes fail to uncover a proper embedding. Here the uncertainty of the scene-flow grows along the sequence so better results are obtained with trajectories that are not too long. See Fig.7 for results.

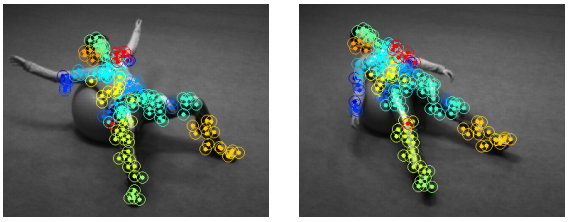

Figure 7: Results of the clustering for the SFT of Fig.1. Projections of the estimated 3-d trajectories at an instant frame on one of the cameras. Colors correspond to the motion segmentation results of analysing 308 trajectories during 35 frames at 15fps with 15% of missing data. setting MNCUT/K = 0.6 and using σ', Algorithm II gives 17 clusters and 66 outliers. Algorithm I only separates body-features from outliers

In general, the values of our quality measures indicate that segmenting scene-flow trajectories with spectral methods is feasible and that the rigidness similarity measure is ideal for discovering motions attached to rigid bodies or rigid-parts of it.

5. Discussion

We have presented a method to perform motion segmentation of sparse scene-flow trajectories. A similarity measure of rigidness is introduced, based on a robust estimation of the distance scale. In opposition to spatial and speed measures (as SIM), our definition of similarity handle rotations. As a result of the representation of the trajectories and the scale estimation, the proposed approach is able to deal with missing data and outliers. Furthermore, this definitions are context independent, which means the method is extendable to any other source of 3-D trajectories.

In all our datasets, results are encouraging. A key result is that both independent and correlated motions can be properly recovered. (See Figures 4,7 and 6) Algorithm-II, is particularly suitable when noisy data and outliers are present. Here, the exploration leads to the identification of singleton classes corresponding to the outliers, while for Algorithm I they were kept merged.

The simplification of the entire trajectories to pairwise similarity measures is effective for reducing the complexity of the clustering that would otherwise happen in a highdimensional space. However, as trajectory information is still available, it can be exploited together with the clustering to refine the results or, for example, recover the motion of each group in time. Once the segmentation results are available, a method such as the one [13] can safely be applied to recover the motion of each part. Since clustering information may be used for generating the samples, the risk of generating artifacts disappears.

References

[1] C. Chennubhotla and A. D. Jepson. Half-lives of eigenflows for spectral clustering. In *NIPS*. MIT Press, 2002.

[2] J. P. Costeira and T. Kanade. A multibody factorization method for independently moving objects. *Int. J. Comput. Vision*, 29(3):159–179, 1998.

[3] F. Devernay, D. Mateus, and M. Guilbert. Multi-camera scene flow by tracking 3-d points and surfels. In *Proc. CVPR*, New York, NY, June 2006. IEEE Comp.Soc.

[4] X. Feng and P. Perona. Scene segmentation from 3d motion. In *CVPR*, page 225, 1998.

[5] I. Fischer and J. Poland. Amplifying the block matrix structure for spectral clustering. In *Machine Learning Conf. Belgium and Netherlands*, 2005.

[6] C. W. Gear. Multibody grouping from motion images. *Int. J. Comput. Vision*, 29(2):133–150, 1998.

[7] A. Gruber and Y. Weiss. Multibody factorization with uncertainty and missing data using the em algorithm. In *CVPR (1)*, pages 707–714, 2004.

[8] K. Inoue and K. Urahama'. Separation of multiple objects in motion images by clustering'. In *ICCV*, 2001.

[9] M. Meila and J. Shi. A random walks view of spectral segmentation. In *AISTATS*, 2001.

[10] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *NIPS*, 2001.

[11] J. Park, H. Zha, and R. Kasturi. Spectral clustering for robust motion segmentation. In *ECCV*, volume 3024 of *Lecture Notes in Computer Science*, pages 390–401. Springer, 2004.

[12] J. Shi and J. Malik. Normalized cuts and image segmentation. *PAMI*, 2000.

[13] O. Tuzel, R. Subbarao, and P. Meer. Simultaneous multiple 3d motion estimation via mode finding on lie groups. In *ICCV*, pages 18–25. IEEE Computer Society, 2005.

[14] D. Verma and M. Meila. A comparison of spectral clustering algorithms. Technical Report uw-cse-03-05-01, university of washington, 2003.

[15] R. Vidal and Y. Ma. A unified algebraic approach to 2-d and 3-d motion segmentation. In *ECCV*, 2004.

[16] R. Vidal, Y. Ma, S. Soatto, and S. Sastry. Two-view multibody structure from motion. *IJCV*, 2005.

[17] H. Wang and P. Culverhouse. Robust motion segmentation by spectral clustering. In *BMVC*, 2003.

[18] Y. Weiss. Segmentation using eigenvectors: A unifying view. In *ICCV (2)*, pages 975–982, 1999.

[19] L. Zelnik-Manor and P. Perona. Self-tuning spectral clustering. In *NIPS*, 2005.