A General Framework for Motion Segmentation: Independent, Articulated, Rigid, Non-rigid, Degenerate and Non-degenerate*

Jingyu Yan and Marc Pollefeys

Department of Computer Science,
The University of North Carolina at Chapel Hill,
Chapel Hill, NC 27599
{yan, marc}@cs.unc.edu

Abstract. We cast the problem of motion segmentation of feature trajectories as linear manifold finding problems and propose a general framework for motion segmentation under affine projections which utilizes two properties of trajectory data: geometric constraint and locality. The geometric constraint states that the trajectories of the same motion lie in a low dimensional linear manifold and different motions result in different linear manifolds; locality, by which we mean in a transformed space a data and its neighbors tend to lie in the same linear manifold, provides a cue for efficient estimation of these manifolds. Our algorithm estimates a number of linear manifolds, whose dimensions are unknown beforehand, and segment the trajectories accordingly. It first transforms and normalizes the trajectories; secondly, for each trajectory it estimates a local linear manifold through local sampling; then it derives the affinity matrix based on principal subspace angles between these estimated linear manifolds; at last, spectral clustering is applied to the matrix and gives the segmentation result. Our algorithm is general without restriction on the number of linear manifolds and without prior knowledge of the dimensions of the linear manifolds. We demonstrate in our experiments that it can segment a wide range of motions including independent, articulated, rigid, non-rigid, degenerate, non-degenerate or any combination of them. In some highly challenging cases where other state-of-the-art motion segmentation algorithms may fail, our algorithm gives expected results.

1 Introduction

Motion segmentation of trajectory data has been an essential issue in understanding and reconstructing dynamic scenes. Dynamic scene consists of multiple moving objects with a static or moving camera. The objective is to segment the feature trajectories according to the motions in the scene.

Ever since Tomasi and Kanade[17] introduced the factorization method based on the idea that trajectories of a general rigid motion under affine projection

* The support of the NSF ITR grant IIS-0313047 is gratefully acknowledged.

A. Leonardis, H. Bischof, and A. Pinz (Eds.): ECCV 2006, Part IV, LNCS 3954, pp. 94–106, 2006.
© Springer-Verlag Berlin Heidelberg 2006

span a 4-dimensional linear manifold, this geometric constraint has been used extensively in motion segmentation, especially for independently moving objects whose trajectories have a nice property that they are from independent subspaces. Most notably, Costeria and Kanade[2] constructs a shape interaction matrix from this fact and uses the zero product between independent trajectories as a segmentation criteria. More recently, Yan and Pollefeys[21], Tresadern and Reid[22] studied articulated motions, another paradigm of dynamic scenes, and drew a conclusion that the motions of linked parts are dependent and their subspaces are intersecting on 1 or 2 dimensions depending on whether the link is a joint or an axis. Besides rigid motions, Bregler et al.[3] and Brand[4] showed that non-rigid motions like human facial motion etc. can be approximated using a higher dimensional linear subspace.

To sum up, motion segmentation of a dynamic scene that consists of multiple motions, either independent, articulated, rigid, non-rigid, degenerate or non-degenerate, can be casted as a linear manifold finding problem. The challenges are from the unknowns like dimensionality and dependency of these linear manifolds.

We propose a general framework for motion segmentation under affine projections. Our algorithm estimates a number of linear manifolds of different dimensions and segment the trajectories accordingly. It first estimates a local linear manifold for each trajectory by local sampling; then it derives an affinity matrix based on principal angles between each pair of estimated linear manifolds; spectral clustering is then applied to the matrix and segments the data. Our algorithm is general without restriction on the number of linear manifolds or their dimensionalities. So it can segment a wide range of motions including independent, articulated, rigid, non-rigid, degenerate, non-degenerate or any combination of them.

Due to the large volume of works of motion segmentation, we need to draw the distinction between our work and the previous ones. Most of the previous works assume independency between motion subspaces (Boult and Brown [8], Gear[9], Costeria and Kanade[2], Kanatani[11], Ichimura[10]) while our goal is to deal with a mixture of dependent and independent motions in a unified way.

Zelnik-Manor and Irani[12] addresses the dependency problem between motion subspaces and deals with it using a method with the same nature as [2] but an elevated perspective from Weiss[14] to derive an affinity matrix, followed by the technique of [11] to separate the dependent motions. In their case, the angle between every pair of vectors, expressed by a dot product, are used as the affinity measurement. However, unlike the independent motion subspace cases, angles, or any other distance measurement between the data, do not reflect the true geometric constraints, the subspace constraints, that we use to segment the data. Instead, our method uses the distance between two locally estimated subspaces of each data, expressed by subspace principal angles, as the affinity measurement. This new affinity measurement reflects the true nature of the constraint for segmentation and leads to more accurate results, which is confirmed by our experiments.

Vidal et al.[18][19][20] propose an algebraic framework called GPCA that can deal with dependent and independent subspaces with unknown dimensionality uniformly. It models a subspace as a set of linear polynomials and a mixture of n subspaces as a set of polynomials of degree n. Given enough sample points in the mixture of subspaces, the coefficients of these high degree polynomials can be estimated. By differentiating at each data point, the normals of each data can be estimated. Then it also uses standard methods like principal angles and spectral clustering to segment the data. However, because GPCA first brings the problem to a high degree nonlinear space and then solves it linearly, the number of sample points required by GPCA to estimate the polynomial coefficients becomes its Achilles' heel, which grows exponentially with the number of subspaces and the dimensions ($\bigcirc((d+1)^n)$, d is the dimension of the largest underlying subspace and n is the number of subspaces). In practice, the number of trajectories can hardly satisfy GPCA's requirement for it to handle more than 3 subspaces. And for non-rigid motion subspaces whose dimensions are more than 4, the situation gets even worse. Our method requires $\bigcirc(d \times n)$ trajectories which makes it practical to handle not only multiple motions but also non-rigid motions that have a higher dimension.

Our approach has not been attempted in motion segmentation. Under a different context [13] uses local sampling and clustering to identify discrete-time hybrid systems in piecewise affine form. We need to point out the differences: first, motion data is not in piecewise form; second, the first step of our approach that projects motion data onto a sphere is important in order to "localize" data of the same underlying subspace while [13] assumes that the data is piecewise beforehand. Our approach is motivated and derived independently, specifically aiming at motion segmentation.

The following sections are organized as followed: Section 2, detailed discussion of motion subspaces of all kinds; Section 3, the algorithm and its analysis; Section 4, experimental results; Section 5, conclusions and future work.

2 The Motion Subspaces

We are going to show that the trajectories of different kinds of motions lie in some low-dimensional linear manifolds under affine projection which models weak and paraperspective projection.

- For rigid motions, the trajectories of a rigid object forms a linear subspace of dimensions no more than 4 ([17]).

$$M_{2f\times p} = [R_{2f\times 3}|T_{2f\times 1}] \begin{bmatrix} S_{3\times p} \\ \mathbf{1}_{1\times p} \end{bmatrix} \tag{1}$$

  f is the number of frames and p, the number of feature trajectories.
- For independent motions, $[R_i|T_i]$ is independent for each motion i, so each motion $M_i = [R_i|T_i] \begin{bmatrix} S_i \\ \mathbf{1} \end{bmatrix}$ lies in an independent linear subspace of dimension no more than 4 ([2]).

– For articulated motions ([21][22]),
  - If the link is a joint, $[R_1|T_1]$ and $[R_2|T_2]$ must have $T_1 = T_2$ under the same coordinate system. So M_1 and M_2 lie in different linear subspaces but have 1-dimensional intersection.
  - If the link is an axis, $[R_1|T_1]$ and $[R_2|T_2]$ must have $T_1 = T_2$ and exactly one column of R_1 and R_2 being the same under a proper coordinate system. So M_1 and M_2 lie in different linear subspaces but have 2-dimensional intersection.
– The trajectories of a non-rigid object can be approximated by different weighings of a number of key shapes ([3][4][5]) and, as shown below, lie in a linear subspace of dimension no more than $3k + 1$.

$$M = \begin{bmatrix} c_1^1 R_{2\times 3}^1 | ... | c_k^1 R_{2\times 3}^1 | T_{2\times 1}^1 \\ ... \\ c_1^f R_{2\times 3}^f | ... | c_k^f R_{2\times 3}^f | T_{2\times 1}^f \end{bmatrix} \begin{bmatrix} S_{3\times 1}^1 \\ ... \\ S_{3\times p}^k \\ \mathbf{1}_{1\times p} \end{bmatrix} \tag{2}$$

c_j^i $(1 \le i \le f, 1 \le j \le k)$.

To sum up, the trajectories of a mixture of motions lie in a mixture of linear manifolds of different dimensions. If we can estimate these underlying linear manifolds accurately enough, we can segment the trajectories accordingly.

3 The Algorithm

In this section, we first outline our algorithm and discuss the details of each step. In the end, we will discuss the issue of outliers.

Our algorithm first transforms the trajectory data; then it estimates a local linear manifold for each trajectory by local sampling; it derives an affinity matrix based on principal subspace angles between each pair of local linear manifolds; spectral clustering is applied to the matrix and gives the segmentation result.

3.1 Motion Data Transformation

Given a motion matrix $W_{2f\times p}$, decompose W into $U_{2f\times K}$,$D_{K\times K}$ and $V_{K\times p}^T$ by SVD, assuming $rank(W)$ is K (A practical algorithm for rank detection is described in (Section 3.5)). Normalize each column of $V(:, 1:K)^T$. Each column unit vector $v_i (i = 1...p)$ becomes the new representation of the corresponding trajectory.

This transformation is an operator that projects a $\mathbf{R^{2f}}$ vector w_i(the ith column of W) onto the $\mathbf{R^K}$ unit sphere which preserves the subspace property, which is that any subset of $w_i (i = 1...p)$ spans a subspace of the same rank of the corresponding subset of $v_i (i = 1...p)$.

The purposes of transforming the trajectories into a unit sphere are:

– Dimension reduction. Notice each trajectory is a $2f \times 1$ vector. Most of the dimensions are redundant and can be effectively reduced by linear transformations.

– Normalization of the data.
– Preparation for the local sampling in the next step. Locality of trajectory data in our algorithm is not defined in the image coordinate space, i.e. proximity in images, but defined on the sphere which has simple geometric meanings.

We are going to perform the segmentation on these unit vectors. It is equivalent to state that we are trying to find a set of R^t spheres $(1 \leq t < K)$ whose union is the R^K sphere. And each vector is grouped according to this set of spheres unless it lies at the intersection of some spheres, in which case it can be grouped to either of these intersecting spheres (Fig. 1).

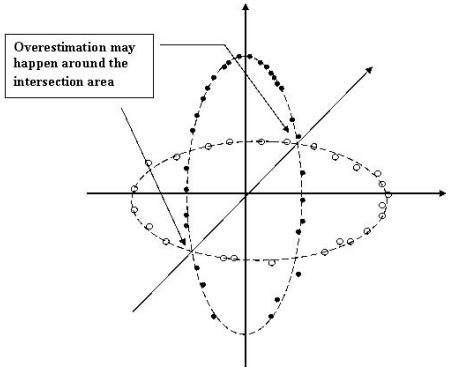

Fig. 1. There are two underlying subspaces of dimension 2 for the data which are transformed onto the $\mathbf{R^3}$ unit sphere. The empty dots represent a group of transformed data belonging to one subspace and the black dots represent another. Due to noise, the dots may not lie exactly on the $\mathbf{R^2}$ spheres. And the intersection area is where "overestimation" may happen, by which we mean that local sampling results in a local subspace estimation that crosses different underlying subspaces.

3.2 Subspace Estimation by Local Sampling

In the transformed space (e.g. see Fig. 1), most points and their closest neighbors lie on the same underlying subspace, which allows us to estimate the underlying subspace of a point α by local samples from itself and its n closest neighbors, i.e. computing the subspace of $[\alpha, \alpha_1, ..., \alpha_n]_{K\times(n+1)}$. This can be easily achieved using SVD (See Section 3.5 for rank detection). Because all the points lie in a $\mathbf{R^K}$ unit sphere, we can use either the Euclidean distance $\|\alpha - \beta\|_2 \in [0, 2]$ or the angle $\arccos(\alpha^T\beta \in [0, \pi]$ to find the n closest neighbors. Our algorithm is not very sensitive to the choice of n as long as $n + 1$ must not be less than the dimension of its underlying subspace.

Two naturally raised questions: what happens to a point near an intersection of subspaces, whose neighbors are from different subspaces (Fig. 1). Secondly, what if a point and its n neighbors do not span the whole underlying subspace? We will discuss these two important situations in the following section after introducing the concept of distance between subspaces.

The subspace constraint of the points is a reliable geometric property for segmentation while the "distance" between the points is not. Most previous works, e.g. [2][10][11][12], use the dot product of the trajectories or some normalized form as the affinity measurement for clustering. The dot product actually measures the angle between the trajectories and is a "distance" in essence. They assume that points of the same subspace are closer in "distance". This assumption mostly stems from works for independent motions [2], in which the dot product is always 0. But for dependent motions whose subspaces intersect like in Fig. 1, this assumption is invalid. Our affinity definition, which is the distance between two local estimated subspaces described by principal angles in the next section, reliably base the segmentation on the criteria of subspace constraint that the points conform to.

3.3 Principal Angles Between Local Subspaces

The distance between two subspaces can be measured by principal angles. The principal angles between two subspaces P and Q are defined recursively as a series of angles $0 \leq \theta_1 \leq,...,\leq \theta_M \leq \pi/2$ (M is the minimum of the dimensions of P and Q):

$$\cos(\theta_m) = max_{u \in S^1, v \in S^2} u^T v = u_m^T v_m$$

where

$$\begin{aligned} \|u\| = \|v\| &= 1 \\ u^T u_i &= 0 \quad i = 1, ..., m-1 \\ v^T v_i &= 0 \quad i = 1, ..., m-1 \end{aligned}$$

We define the affinity of two points, α and β, as the distance between their estimated local subspaces denoted $S(\alpha)$ and $S(\beta)$.

$$a(\alpha, \beta) = e^{-\sum_{i=1,...,M} sin^2(\theta_i)}$$

where θ_1,...,θ_M are the principal angles. Thus, we can build an affinity matrix for spectral clustering described in the following section.

Before we proceed to the next section, let us take a closer look at the two scenarios pointed out at the end of Section 3.2.

– When an estimated local subspace crosses different underlying subspaces, which happens to points near an intersection as shown in Fig. 1 (this usually happens to the trajectories of features very close to or at an articulated axis or joint), we call this estimation "overestimated". An overestimated subspace is usually distanced from the underlying subspaces that it crosses because it has dimensions from other underlying subspace(s). However, points near an intersection are usually small in amount compared to the total. So overestimated subspaces do not have a dominant effect for clustering. Besides, which cluster a point near an intersection may be classified to relies

on which underlying subspaces have a larger portion of its overestimated subspace. So in the end it tends to cluster the point to its real underlying subspace. If not, it results in a misclassification. Our experiments show that if there are misclassifications, mostly it happens to points that are close to an intersection.

– When the estimated local subspace is a subspace of the underlying subspace, we call this estimation "underestimated" since it only estimates a part of it. This occurs when the local neighbors may not span the whole underlying subspace. However, this will not affect the effectiveness of the segmentation introduced in the following section. To explain why, we use an example in rigid motions and allow other cases. Suppose two underlying subspaces of dimension 4 having a 2-dimension intersection. This happens when two articulated parts are linked by an axis [21][22]. The total dimension is 6. The two underlying subspaces, S^1 and S^2, and their underestimated subspaces, A, B, C and D, E, F, of dimension 3 (2 is rare because the features corresponding to the point and its neighbors need to be exactly on a line for that to happen) are as follows.

Subspace \ *Dimensions*	1	2	3	4	5	6
S^1	X	X	X	X		
A	X	X	X			
B		X	X	X		
C	X		X	X		
S^2			X	X	X	X
D			X	X	X	
E				X	X	X
F			X		X	X

(3)

The number of non-zero principal angles between these subspaces and their underestimated subspaces are shown as follows.

Subspaces	S^1	A	B	C	S^2	D	E	F
S^1	0	0	0	0	2	1	2	2
A	0	0	1	1	2	2	3	2
B	0	1	0	1	1	1	2	2
C	0	1	1	0	1	1	2	2
S^2	2	2	1	1	0	0	0	0
D	1	2	1	1	0	0	1	1
E	2	3	2	2	0	1	0	1
F	2	2	2	2	0	1	1	0

(4)

Generally, intra-subspaces have smaller number of non-zero principal angles compared to inter-subspaces. So expectedly, an underestimated subspace tends to be closer to all possible estimated subspaces of its underlying subspace than to those of another underlying subspace.

3.4 Spectral Clustering

We can apply spectral clustering, e.g.[16][15], to the affinity matrix and retrieve N clusters. We advocate recursive 2-way clustering detailed in [16]. Thus we can re-estimate the local subspaces within the current clusters so that points belonging to different clusters will not affect each other any more. Secondly, recursive 2-way clustering gives a more stable result because k-way clustering like [15] depends on k-means which in turn depends on some random initialization. The recursive 2-way clustering is as follows, given N is the total number of underlying subspaces:

- Compute the affinity matrix using the approach in Section 3.2 and 3.3 above and segment the data into two clusters $\{C_1, C_2\}$ by spectral clustering.
- While $NumOfClusters\{C_1, ..., C_n\} < N$, compute the affinity matrix for each cluster C_i $(i = 1, ..., n)$ from the points within the cluster; divide C_i into two clusters, C_i^1 and C_i^2; evaluate the Cheeger constant[15] of each pair of C_i^1 and C_i^2 and decide the best subdivision, C_J^1 and C_J^2 ($1 \leq J \leq n$); replace C_J with them.

3.5 Effective Rank Detection

In practice, a data matrix may be corrupted by noise or outliers and thus has a higher rank. We may use a model selection algorithm inspired by a similar one in [20] to detect an effective rank.

$$r_n = \arg\min_r \frac{\lambda_{r+1}^2}{\sum_{k=1}^{r} \lambda_k^2} + \kappa\, r$$

with λ_i, the i^{th} singular value of the matrix, and κ, a parameter. If the sum of all λ_i^2 is below a certain threshold, the effective rank is 0. The higher the noise level is, the larger κ we should set.

For rank detection of local estimated subspaces, due to small number of samples, noise level is higher, so we prefer a larger κ.

3.6 Outliers

In practice, the trajectories may have some outliers. We are going to discuss their effects under the context of our algorithm.

First of all, an outlier will be classified to one of the segments, which depends on its locally estimated subspace. We suggest that outliers can be better dealt with after the segmentation because the segmented subspace offers less freedom for outliers and makes it easier to detect and reject them.

Second, an outlier will corrupt the estimation of local subspaces of a nearby point. However, this bad effect will not propagate under our algorithm and only remains on those points whose neighbors include the outlier. Misclassification may happen to these points. But as long as the outliers are not dominant in number, our algorithm is robust.

4 Experiments

We test our approach in various real dynamical scenes with 2 to 6 motions and a combination of independent, articulated, rigid, non-rigid, degenerate and non-degenerate motions.

For the experiments, we choose $\kappa = 10^{-6}$ to 10^{-5} for trajectory rank estimation depending on the noise level of the trajectories, and $\kappa = 10^{-3}$ for local subspace rank estimation (See Section 3.5 for more detail). We let $n = d$ where n is the number of neighbors for local sampling and d is the highest possible dimension of the underlying subspaces. That is 4 for rigid motions and 7 for non-rigid motions in our experiments.

Miscalssification errors vs. total number of trajectories and the number of outliers vs. total number of trajectories for the experiments is summaried in Table 1. Outliers may be clustered to any segments and are not counted as misclassification errors.

Table 1. A comparison between our method, GPCA and trajectory angle based method

Experiment	**Our Method**	**GPCA**	**Angle based**	**Outliers**
Truck	0/83	5/83	16/83	0/83
Head and Body	1/99	10/99	9/99	6/83
Booklet	0/38	2/38	2/38	1/38
Two bulldozers	1/94	4/94	24/94	8/94
One bulldozers	4/85	6/85	11/85	9/85
Dancing	21/268	not enough samples[1]	78/268	7/268

The first experiment is from a scene with non-degenerate data of an articulated object with a rotating axis. The detected rank of the trajectories is 6. The segmentation result is shown in Fig. 2. The ranks of the segmented trajectories are both 4.

The second experiment is from a scene of 2 non-degenerate motions of an articulated body with a joint. The detected rank of the trajectories is 7. There are one misclassification, the red dot on the left shoulder. The other red dot on the left arm is an outlier. The segmentation result is shown in Fig. 3. The ranks of the segmented trajectories are both 4.

The third experiment is from a scene of 2 degenerate shapes of an articulated object. Two pages of a booklet is being opened and closed. The detected rank of the trajectories is 4. The segmentation result is shown in Fig. 4. The ranks of the segmented trajectories are both 3.

[1] For the last experiment of 6 motions, GPCA requires a huge number of trajectories for it to work. Roughly, it needs $\bigcirc((d+1)^6)$. d is the dimension of the largest subspace. For non-degenerate rigid motions, $d = 5$ [18]; for non-rigid motions, d is even larger. That many number of trajectories are normally not available in practice.

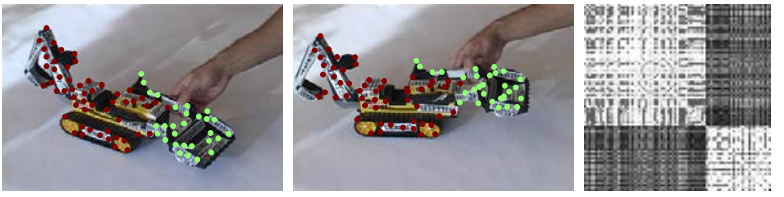

Fig. 2. (*left and middle*) A sequence of a truck moving with the shovel rotating around an axis. The color of a dot, red or green, shows the segmentation result. (*right*) The affinity matrix of local estimated subspaces is shown. The row and columns are rearranged based on the segmentation.

Fig. 3. (*left and middle*) A sequence of a person moving with his head rotating around the neck. The color of a dot, red or green, shows the segmentation result. There is one misclassification, the red dot on the left shoulder. The other red dot on the left arm is an outlier. (*right*) The affinity matrix of locally estimated subspaces is shown. The row and columns are rearranged based on the segmentation.

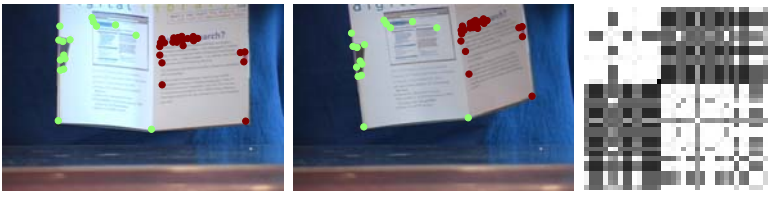

Fig. 4. (*left and middle*) A sequence of a booklet whose two pages are being opened and closed around an axis. The color of a dot, red or green, shows the segmentation result. There is no misclassification error. The green dot on the rotating axis can be grouped to either page. (*right*) The affinity matrix of local estimated subspaces is shown. The row and columns are rearranged based on the segmentation.

The fourth experiment has 3 motions. It comes from a scene of 2 independently moving bulldozers, one of which has an articulated arm rotating around an axis. Only the side of the articulated arm can be seen so it is a degenerate shape. The detected rank of the trajectories is 8 before the first segmentation. Both of the segments have rank 4. The next subdivision is automatically detected (See Section 3.4) for points from the right bulldozer and the segmented trajectories are of rank 3. The segmentation result is shown in Fig. 5.

The fifth experiment has 3 motions. It comes from a scene with an articulated object of 3 parts. The bulldozer has its forearm and upper-arm moving articulately rotating around two axes. The detected rank of the trajectories is 6 before

Fig. 5. (*left 2*)A sequence of two bulldozers moving independently, one of which moves articulately with its arm rotating around an axis. The color of a dot, red, blue or yellow, shows the segmentation result. There is one misclassification error which is the red dot on the forearm near the axis. Besides that, there are several outliers. (*right 2*) The affinity matrices for 2-stage segmentations are shown. The row and columns are rearranged based on the segmentation.

Fig. 6. A sequence of a bulldozer with its upper-arm and forearm moving articulately around some axis. The color of a dot, red, blue or yellow, shows the segmentation result. There are 4 misclassification errors. Two are the yellow dots on the forearm and two are the red dots on the upper-arm. All of them are near the axis connecting both arms. Besides these, there are several outliers in the trajectories and they are clustered to one of the segments.

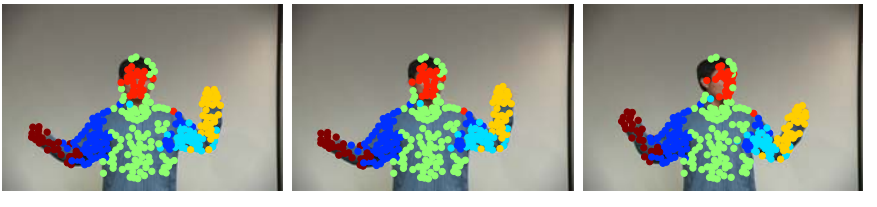

Fig. 7. (*top*) A sequence of a person dancing with his upper body, his head and both of his upper arms and forearms moving. His mouth motion is non-rigid. The color of a dot shows the segmentation result. Besides outliers, there are about 8% misclassifications.

the first segmentation. The segmented trajectories have a rank 3 and 4. The rank-4 cluster gets subdivided into 2 rank-3 clusters. There are outliers in this experiment. They have been clustered to one of the segments. Besides outliers, there are 4 misclassifications, two of which are the yellow dots on the forearm and two of which are the red dots on the upper-arm and all of which are near the axis. The segmentation result is shown in Fig. 6.

The final experiment has 6 motions. It comes from a scene with a person dancing with his upper body, his head and both of his upper arms and forearms mov-

ing. Besides, his mouth movement generates a non-rigid motion. This is a highly challenging case not only because of the total number of motions but also because of the dependency between these articulated motions and the non-rigid kind of motion on the person's face. The detected rank of the trajectories is 12 before the first segmentation. Both of the segmented rank-7 and rank-6 clusters get subdivided into rank-6 and -3 clusters, and rank-4 and -3 clusters repectively. In the end, the rank-4 cluster gets subdivided into two rank-3 subspaces. There are outliers and there are about 8% misclassifications, most of which are near the articulated axes or joints. Interestingly, the green dots on the head are those features not turning as the head turns. Instead, they move like the features on the upper body. And indeed, our algorithm classifies them to those features on the upper body. A second interesting observation is the misclassification of the dark blue dots near the joint between the body and the person's right arm. Though they are far away from the left upper arm of the person, they are actually very close to the intersection between the motion subspaces of the left and right upper arms because they both are linked to the body. The segmentation result is shown in Fig. 7.

4.1 Comparisons

We compare our method with GPCA[18] and trajectory angle based approach, e.g. [12] except for that we use spectral clustering to segment the affinity matrix (Table 1). The numbers in the table are misclassification errors vs. the total number of trajectories and outliers vs. trajectories. Outliers are not counted as misclassification.

5 Conclusions and Future Work

We propose a general framework for motion segmentation of a wide range of motions including independent, articulated, rigid, non-rigid, degenerate and non-degenerate. It is based on local estimation of the subspace to which a trajectory belongs through local sampling and spectral clustering of the affinity matrix of the these subspaces. We demonstrate our approach in various situations. In some highly challenging cases where other state-of-the-art motion segmentation algorithms may fail, our algorithm gives expected results.

We plan to reconstruct complex dynamical scenes with a variety of objects and motions. An especially interesting case is human motion. Our algorithm can provide a good initialization for a follow-up EM algorithm to improve the segmentation and reject outliers.

References

1. M. A. Fischler, R. C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. Comm. of the ACM, Vol 24, pp 381-395, 1981.
2. J.P. Costeira, T. Kanade, "A Multibody Factorization Method for Independently Moving Objects", *IJCV*, Vol. 29, Issue 3 pp. 159-179, 1998.

3. C. Bregler, A. Hertzmann, H. Biermann, "Recovering Non-Rigid 3D Shape from Image Streams", Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '00), June 2000.
4. M. Brand, "Morphable 3D models from video", CVPR, pp. II:456-463, 2001.
5. J. Xiao, J. Chai, and T. Kanade, "A closed-form solution to non-rigid shape and motion recovery", Proceedings of the European Conference on Computer Vision, 2004.
6. Ullman, S. 1983. Maximizing rigidity: The incremental recovery of 3D structure from rigid and rubbery motion. Technical Report A.I. Memo No. 721, MIT.
7. Sinclair, D. 1993. Motion segmentation and local structure. In Proceedings of the 4th International Conference on Computer Vision.
8. Boult, T. and Brown, L. 1991. Factorization-based segmentation of motions. In Proceedings of the IEEE Workshop on Visual Motion.
9. Gear, C.W. 1994. Feature grouping in moving objects. In Proceedings of the Workshop on Motion of Non-Rigid and Articulated Objects, Austin, Texas
10. N. Ichimura. Motion segmentation based on factorization method and discriminant criterion. In Proc. IEEE Int. Conf. Computer Vision, pages 600605, 1999.
11. K. Kanatani. Motion segmentation by subspace separation and model selection:model selection and reliability evaluation. Intl. J. of Image and Graphics, 2(2):179197, 2002.
12. L. Zelnik-Manor and M. Irani. Degeneracies, dependencies and their implications in multi-body and multi-sequence factorizations. In Proc. IEEE Computer Vision and Pattern Recognition, 2003.
13. G. Ferrari-Trecate, M. Muselli, D. Liberati, and M. Morari. A clustering technique for the identification of piecewise affine and hybrid systems. Automatica, 39:205–217, 2003.
14. Y. Weiss. Segmentation using eigenvectors: A unifying view. In International Conference on Computer Vision, pages 975982, Corfu, Greece, September 1999.
15. A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: analysis and an algorithm. In Advances in Neural Information Processing Systems 14. MIT Press, 2002.
16. J. Shi and J. Malik, Normalized Cuts and Image Segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 2000.
17. C. Tomasi, T. Kanade, "Shape and motion from image streams under orthography: a factorization method", *IJCV*, Vol. 9, Issue 2 pp. 137-154, 1992.
18. R. Vidal and R. Hartley. Motion Segmentation with Missing Data using PowerFactorization and GPCA. IEEE Conference on Computer Vision and Pattern Recognition, 2004
19. R. Vidal, Y. Ma and S. Sastry, "Generalized Principal Component Analysis (GPCA) ", Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'03), June 2003.
20. R. Vidal, Y. Ma and J. Piazzi, "A New GPCA Algorithm for Clustering Subspaces by Fitting, Differentiating and Dividing Polynomials", Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'04), June 27 - July 02, 2004.
21. J. Yan, M. Pollefeys, A Factorization-based Approach to Articulated Motion Recovery, IEEE Conf. on Computer Vision and Pattern Recognition, 2005
22. P. Tresadern and I. Reid, Articulated Structure From Motion by Factorization, Proc IEEE Conf on Computer Vision and Pattern Recognition, 2005
23. Multiple View Geometry in Computer Vision, Richard Hartley and Andrew Zisserman, Cambridge University Press, 2002
24. G. Golub and A. van Loan. Matrix Computations. Johns Hopkins U. Press, 1996