# Motion Segmentation with Missing Data using PowerFactorization and GPCA

René Vidal[1,2]
[1]Center for Imaging Science
Johns Hopkins University
rvidal@cis.jhu.edu

and

[2]National ICT Australia

Richard Hartley[2,3]
[3]Dept. of Systems Engineering
Australian National University
Richard.Hartley@anu.edu.au

## Abstract

*We consider the problem of segmenting multiple rigid motions from point correspondences in multiple affine views. We cast this problem as a subspace clustering problem in which the motion of each object lives in a subspace of dimension two, three or four. Unlike previous work, we do not restrict the motion subspaces to be four-dimensional or linearly independent. Instead, our approach deals gracefully with all the spectrum of possible affine motions: from two-dimensional and partially dependent to four-dimensional and fully independent. In addition, our method handles the case of missing data, meaning that point tracks do not have to be visible in all images. Our approach involves projecting the point trajectories of all the points into a 5-dimensional space, using the PowerFactorization method to fill in missing data. Then multiple linear subspaces representing independent motions are fitted to the points in $\mathbb{R}^5$ using GPCA. We test our algorithm on various real sequences with degenerate and nondegenerate motions, missing data, perspective effects, transparent motions, etc. Our algorithm achieves a misclassification error of less than 5% for sequences with up to 30% of missing data points.*

## 1. Introduction

The past few decades have witnessed significant advances on the understanding of the geometry and reconstruction of *static scenes* observed by a moving camera. More recently, there has been an increasing interest on the geometrical and statistical models for the understanding of *dynamic scenes*, in which both the camera and multiple objects move.

The case of multiple moving objects seen by two perspective views was recently studied by Vidal et al. [19, 20], who proposed a generalization of the 8-point algorithm based on the so-called *multibody epipolar constraint* and its associated *multibody fundamental matrix*. The method simultaneously recovers multiple fundamental matrices using multivariate polynomial factorization, and can be extended to most two-view motion models in computer vision, such as affine, translational and planar homographies, by fitting and differentiating complex polynomials [16]. The case of multiple moving objects seen by three perspective views has also been recently solved by exploiting the algebraic and geometric properties of the *multibody trifocal tensor* [6].

To the best of our knowledge, motion segmentation from multiple views has only been studied in the case of affine cameras, because in this case the motion of each one of the rigidly moving objects lives in a four-dimensional subspace [1, 14]. That is, if $\boldsymbol{x}_{fp}$ is the image of point $\boldsymbol{X}_p \in \mathbb{P}^3$ in frame $f$ and $\mathtt{A}_f \in \mathbb{R}^{2\times 4}$ is the affine camera matrix in frame $f$, then $\boldsymbol{x}_{fp} = \mathtt{A}_f \boldsymbol{X}_p$. Therefore if we stack all the image measurements into a $2F \times P$ matrix $\mathtt{W}$, then we have

$$\mathtt{W} = \mathtt{M}\mathtt{S}^\top$$

$$\begin{bmatrix} \boldsymbol{x}_{11} \cdots \boldsymbol{x}_{1P} \\ \vdots \quad\quad \vdots \\ \boldsymbol{x}_{F1} \cdots \boldsymbol{x}_{FP} \end{bmatrix}_{2F\times P} = \begin{bmatrix} \mathtt{A}_1 \\ \vdots \\ \mathtt{A}_F \end{bmatrix}_{2F\times 4} \begin{bmatrix} \boldsymbol{X}_1 \cdots \boldsymbol{X}_P \end{bmatrix}_{4\times P}. \quad (1)$$

Therefore $\text{rank}(\mathtt{W}) \leq 4$ and the motion of each object lives in a four-dimensional subspace of $\mathbb{R}^{2F}$. This subspace constraint was used by Boult and Brown [1] to propose the first multiframe motion segmentation algorithm based on thresholding the leading singular vector of $\mathtt{W}$. Costeira and Kanade (CK) [2] extended this approach by thresholding the entries of the so-called *shape interaction* matrix

$$\mathtt{Q} = \mathtt{V}\mathtt{V}^\top, \quad (2)$$

which is built from the SVD of the measurement matrix $\mathtt{W} = \mathtt{U}\mathtt{S}\mathtt{V}^\top$ and has the property that

$$\mathtt{Q}_{ij} \begin{cases} = 0 & \text{if } i \text{ and } j \text{ correspond to the same motion} \\ \neq 0 & \text{otherwise} \end{cases}. \quad (3)$$

Unfortunately, thresholding the entries of the shape interaction matrix is very sensitive to noise [8, 22]. Wu et al. [22] reduce the effect of noise by building a similarity matrix from the distances among the subspaces obtained by the CK algorithm. Kanatani modifies the entries of the interaction matrix using the geometric information criterion [8] and also incorporates the fact that the subspaces are affine [10].

Another disadvantage of the CK algorithm is that equation (3) holds if and only if the motion subspaces are linearly independent [8]. That is, the algorithm is not provably correct for most practical motion sequences which usually exhibit partially dependent motions, such as when two objects have the same rotational but different translational motion relative to the camera, or vice versa. This has motivated the recent work of Zelnik-Manor and Irani [23] who

use the singular vectors of the normalized shape interaction matrix to build a similarity matrix from which the clustering of the features is obtained using spectral clustering techniques (see [21] and references therein). This solution is based on the expectation that on the average the angular displacement of trajectories corresponding to the same motion is smaller than the one of trajectories corresponding to different motions. This assumption is, however, not provably correct. Kanatani [11] has also studied the case of partially dependent (degenerate) motions under the assumption that the type of degeneracy is known, e.g. 2-D similarity motion or pure translation. In these particular cases the motion subspaces, though of smaller dimension, are still independent. Once an initial clustering of the correspondences is obtained, the motion models are estimated using an iterative process that alternates between feature clustering and motion estimation, similarly to the Expectation Maximization (EM) algorithm. Other EM-like approaches to motion segmentation can be found in [15].

### 1.1. Contributions of this paper

In this paper, rather than insisting on heuristics that modify a motion segmentation algorithm that is provably correct only when the motion subspaces are fully independent, we seek an approach that works for all the spectrum of affine motions: from two-dimensional and partially dependent to four-dimensional and fully independent. This is achieved by a combination of PowerFactorization, Generalized Principal Component Analysis (GPCA), and spectral clustering which leads to the following purely geometric solution to the multiframe motion segmentation problem:

1. Project the point trajectories onto a five-dimensional subspace using PowerFactorization, which automatically deals with the case of missing data. A byproduct of this projection is that our algorithm requires a minimum number of *three* views for *any* number of independent motions.[1]
2. Fit a collection of subspaces to the projected trajectories using GPCA and spectral clustering. Specifically,
   (a) Fit a set of homogeneous polynomials representing all motion subspaces to the projected data.
   (b) Obtain a basis for each motion subspace from the derivatives of these polynomials.
   (c) Cluster the data by applying spectral clustering to a similarity built from the subspace angles.

We demonstrate our approach on real and synthetic sequences with degenerate and nondegenerate motions, missing data, perspective effects, transparent motions, etc. Our experiments show that our algorithm achieves a misclassification error of less than 5% for sequences with up to 30% of missing data points.

[1] Previous work required the image points to be visible in $2n$ views for $n$ independent motions, as we will see in the next section.

## 2. Multiframe Motion Segmentation

Let $\{\boldsymbol{x}_{fp} \in \mathbb{R}^2\}_{p=1,\dots,P}^{f=1,\dots,F}$ be a collection of $P$ feature points in $F$ frames, with $\boldsymbol{x}_{fp}$ representing the image of a point $\boldsymbol{X}_p \in \mathbb{P}^3$ in frame $f$. We assume an affine camera model, which generalizes orthographic, weak perspective, and paraperspective projection [4], i.e.

$$\boldsymbol{x}_{fp} = \mathtt{A}_f \boldsymbol{X}_p, \tag{4}$$

where $\mathtt{A}_f \in \mathbb{R}^{2\times 4}$ is the affine camera matrix for frame $f$, which depends on the position and orientation of the camera as well as the internal calibration parameters. It follows from equations (4) and (1) that when all the points $\{\boldsymbol{X}_p\}$ correspond to a single moving object the trajectories generated by those points, that is the columns of the data matrix

$$\mathtt{W} = \mathtt{M}\mathtt{S}^\top \in \mathbb{R}^{2F\times P}, \tag{5}$$

live in a four-dimensional subspace of $\mathbb{R}^{2F}$ spanned by the columns of the *motion matrix* $\mathtt{M} \in \mathbb{R}^{2F\times 4}$.

Consider now the case in which the set of points $\{\boldsymbol{X}_p\}_{p=1}^P$ corresponds to $n$ moving objects undergoing $n$ different motions. In this case, each moving object spans a different four-dimensional subspace of $\mathbb{R}^{2F}$. Therefore, if we knew the clustering of the features we could build a data matrix $\mathtt{W}_i \in \mathbb{R}^{2F\times P_i}$, with $\sum_{i=1}^n P_i = P$, satisfying

$$\mathtt{W}_i = \mathtt{M}_i \mathtt{S}_i^\top \quad i = 1,\dots,n. \tag{6}$$

Since in this paper we assume that the clustering of the features is *unknown*, we only know that the $P$ columns of $\mathtt{W}$ live in a union of $n$ subspaces. Therefore, solving the motion segmentation problem is equivalent to finding a basis for each one of such subspaces. When the rank of the data matrix is $\text{rank}(\mathtt{W}) = 4n$, which implies that $F \geq 2n$ and $P \geq 4n$, the motion subspaces are fully dimensional and independent. Hence the problem can be solved as suggested by the CK algorithm. The same principle applies when the motion subspaces are not necessarily four-dimensional, but independent, that is when $\text{rank}(\mathtt{W}) = \sum_{i=1}^n \text{rank}(\mathtt{W}_i)$.

However, it could be the case that two motions are different, but their associated subspaces are partially dependent, e.g. when they have the same rotation and a different translation. Furthermore, it could be the case that the two motions are fully independent, yet their motion subspaces are partially dependent, e.g. with $n = 2$ four-dimensional motions and $F = 3$ frames we have $\text{rank}(\mathtt{W}) \leq 6 < 4 + 4 = 8$. In general, partially dependent motions satisfy

$$\begin{aligned} &\sum_{i=1}^n \text{rank}(\mathtt{W}_i) > \text{rank}(\mathtt{W}) \qquad \text{and} \\ &\text{rank}([\mathtt{W}_i \;\; \mathtt{W}_{i'}]) > \max\{\text{rank}(\mathtt{W}_i), \text{rank}(\mathtt{W}_{i'})\}, \end{aligned} \tag{7}$$

where the latter equation rules out the case of fully dependent motions. The problem is then to find an algorithm that deals both with the case of partially dependent and fully independent motions, which we do in the following sections.

## 2.1. Projecting onto a 5-dimensional subspace

### 2.1.1 SVD and complete data

Let us first consider the case in which all the feature points are visible in all frames, so that all the entries of the data matrix $\mathtt{W} \in \mathbb{R}^{2F \times P}$ are known. The first step of our algorithm is to project the point trajectories (columns of $\mathtt{W}$) from $\mathbb{R}^{2F}$ to $\mathbb{R}^5$. In choosing a projection, it makes sense to lose as little information as possible by projecting into a dominant eigensubspace, which we can do simply by computing the SVD of $\mathtt{W} = \mathtt{U}\mathtt{D}\mathtt{V}^\top$, and then defining a new data matrix $\hat{\mathtt{W}}$ to consist of the first 5 rows of $\mathtt{V}^\top$. At a first sight, it may seem counter-intuitive to perform this projection. For instance, if we have $F = 4$ frames of $n = 2$ independent four-dimensional motions, then we can readily apply the CK algorithm, because we are in a nondegenerate situation. However, if we first project onto $\mathbb{R}^5$, the motions subspaces become partially dependent because $\text{rank}(\hat{\mathtt{W}}) = 5 < 4+4 = 8$. What is the reason for projecting then? The reason is that the clustering of data lying on subspaces is preserved by a generic linear projection. For instance, if one is given data lying on two lines in $\mathbb{R}^3$ passing through the origin, then one can first project the two lines onto a plane in general position[2] and then cluster the data inside that plane. More generally the principle is [18]:

**Theorem 2.1.** *If a set of vectors $\{\boldsymbol{w}_i\}$ all lie in a linear subspace of dimension $k$ in $\mathbb{R}^N$, and if $\pi$ represents a linear projection into a subspace $S$ of dimension $K$, then the points $\{\pi(\boldsymbol{w}_i)\}$ lie in a linear subspace of $S$ of dimension no greater than $k$. Furthermore, if $K > k$, then there is an open and dense set of projections such that the dimension of the projected subspaces is still $k$.*

The same principle applies to the motion segmentation problem. Since we know that the maximum dimension of each motion subspace is four, then projecting onto a generic five-dimensional subspace preserves the clustering of the motion subspaces. Loosely speaking, in order for two different motions to be distinguishable from each other, it is enough for them to be different along one dimension, i.e. we do not really need to have the subspaces be different in all four dimensions. It is this key observation the one that enable us to treat *all* partially dependent motions as well as *all* independent motions in the same framework: clustering subspaces of dimension two, three or four living in $\mathbb{R}^5$.

Another advantage of projecting the data onto a 5-dimensional space is that, except for the projection itself, the complexity of the motion segmentation algorithm we are about to present becomes independent on the number of frames. Indeed, our algorithm requires a minimum of only three frames.[1] Furthermore, it enables us to handle the case of missing data, which could not be done with the CK algorithm, as we explain in the next section.

[2] A plane perpendicular to any of the lines or perpendicular to the plane containing the lines would fail.

### 2.1.2 PowerFactorization and incomplete data

As an alternative to using the SVD to do the projection, we can use the technique of PowerFactorization [5], which in some cases may be more rapid. In addition, it allows us deal with the case in which some entries of the data matrix $\mathtt{W} \in \mathbb{R}^{2F \times P}$ are missing, a fairly common occurrence in feature tracking due to occlusions or points disappearing from the field of view. In this case, we need a way to project such point trajectories into $\mathbb{R}^5$ in the same way as with complete trajectories. Clearly this cannot be done using SVD, as detailed in the previous section.

We use a method adapted for incomplete data, based on an analysis of the "power method" for computation of eigenvalues of a matrix. The method known as PowerFactorization gives a rapid method for approximating low rank matrices. The PowerFactorization algorithm is discussed in some detail in [5].

**Complete data case.** We begin by describing PowerFactorization in the complete data case. Let $\mathtt{W}$ be a matrix of dimension $N \times P$ that we want to approximate by some matrix of rank $r$. We start with a random matrix $\mathtt{A}_0$ of dimension $N \times r$, and then carry out the following steps for $k = 1, 2, \ldots$ until convergence of the product $\mathtt{A}_k{\mathtt{B}_k}^\top$.

1. Let $\mathtt{B}_k = \mathtt{W}^\top \mathtt{A}_{k-1}$.
2. Orthonormalize the columns of $\mathtt{B}_k$ by (for instance) the Gram-Schmidt algorithm. This sometimes called QR algorithm, since it is equivalent to replacing $\mathtt{B}_k$ by a matrix $\mathtt{B}'_k$ such that $\mathtt{B}_k = \mathtt{B}'_k \mathtt{N}_k$, where $\mathtt{B}'_k$ has orthonormal columns, and $\mathtt{N}_k$ is upper-triangular.
3. Let $\mathtt{A}_k = \mathtt{W}\mathtt{B}'_k$.

It was indicated in [5] that the sequence $\mathtt{A}_k{\mathtt{B}_k}^\top$ converges rapidly to the rank-$r$ matrix closest to $\mathtt{W}$ in Frobenius norm, provided that $\mathtt{W}$ is close to having rank $r$. Specifically, the rate of convergence is proportional to $(s_{r+1}/s_r)^k$, where $s_i$ is the $i^{th}$ largest singular value of $\mathtt{W}$. Observe that this algorithm is very simple, requiring only matrix multiplications and Gram-Schmidt normalization.

In the case of motion segmentation, the subject of this paper, we wish to replace $\mathtt{W}$ by a matrix obtained by projecting its columns onto a 5-dimensional subspace. If $\mathtt{W} = \mathtt{A}\mathtt{B}^\top$ is the nearest rank-5 factorization to $\mathtt{W}$, then $\mathtt{B}^\top$ is the matrix that we require.

**Missing data case.** In the case where some of the entries of $\mathtt{W}$ are not known, we can not carry out SVD or matrix multiplication either. However, the goal remains the same – to find matrices $\mathtt{A}$ and $\mathtt{B}$ such that $\mathtt{A}\mathtt{B}^\top$ is as close to $\mathtt{W}$ as possible. The measure of closeness is

$$\sum_{(i,j)\in\mathcal{I}} (\mathtt{W}_{ij} - (\mathtt{A}\mathtt{B}^\top)_{ij})^2 \tag{8}$$

where $\mathcal{I}$ is the set of pairs $(i, j)$ for which $\mathtt{W}_{ij}$ is known.

In the case of missing data, PowerFactorization takes a slightly different form. Starting as before with a random matrix $\mathtt{A}_0$, we alternate the following steps until convergence of $\mathtt{A}_k{\mathtt{B}_k}^\top$.

1. Given $\mathtt{A}_{k-1}$, find the $n \times r$ matrix $\mathtt{B}_k$ that minimizes $\sum_{(i,j)\in\mathcal{I}} |\mathtt{W}_{ij} - (\mathtt{A}_{k-1}{\mathtt{B}_k}^\top)_{ij}|^2$.
2. Normalize $\mathtt{B}_k$.
3. Given $\mathtt{B}_k$, find the matrix $\mathtt{A}_k$ that minimizes $\sum_{(i,j)\in\mathcal{I}} |\mathtt{W}_{ij} - (\mathtt{A}_k{\mathtt{B}_k}^\top)_{ij}|^2$.

In this algorithm, the computation of each $\mathtt{B}_k$ and $\mathtt{A}_k$ proceeds just one column at a time, and consists of finding the least-squares solution to a set of linear equations.

It was pointed out in [5] that if $\mathtt{W}$ has no missing entries, then this algorithm gives precisely the same sequence of products as the version of the algorithm involving matrix multiplications. Consequently, it is provably rapidly convergent to the optimal solution. In the case of moderate amounts of missing data, we can not strongly assert this, though the theoretical result for complete data gives us a strong expectation that this will be so; this expectation is borne out by practical experience. PowerFactorization fails to converge to the global minimum only in rare cases, such as with strongly banded data (that is a long image sequence with only short point-tracks).

Essentially what this algorithm is doing is simply alternating between computing $\mathtt{A}_k$ and $\mathtt{B}_k$ using least-squares. Similar (not identical) alternation algorithms have been proposed in [12, 3]. As mentioned [5] gives theoretical justification for this algorithm.

## 2.2. Fitting motion subspaces using GPCA

We have reduced the motion segmentation problem to finding a set of linear subspaces in $\mathbb{R}^5$, each of dimension at most 4, which contain the data points (or come close to them). The points in question are the columns of the projected data matrix $\hat{\mathtt{W}}$. We solve this problem by fitting and differentiating polynomials using GPCA [18, 17].

### 2.2.1 Representing motion subspaces with polynomials

It serves our purposes to fit a slightly more general model to the points, fitting them by an *algebraic variety*. In essence, notice that the $n$ motion subspaces can be represented as the zero-set of $m$ polynomials $\{p_{n\ell}(\boldsymbol{w})\}_{\ell=1}^m$ of degree $n$ in 5 variables, i.e. $\boldsymbol{w} \in \mathbb{R}^5$. That is, a set of linear subspaces forms an algebraic variety in $\mathbb{R}^5$. Our task is to find the algebraic variety that best fits the set of points in $\mathbb{R}^5$. The fact that the set of linear subspaces forms an algebraic variety (in simple terms, can be expressed as the zero set of a collection of polynomials) is simple enough. If all the subspaces are hyperplanes (having dimension 4) in $\mathbb{R}^5$, then a single polynomial of degree $n$ suffices. This is because a single plane is represented by a single linear polynomial, just as a plane in $\mathbb{R}^3$ is represented by a linear polynomial equation $ax+by+cz = 0$. Similarly, a set of $n$ hyperplanes is represented by the product of $n$ linear polynomials, one for each plane, e.g. $(ax + by + cz)(dx + ey + fz) = 0$.

A linear subspace of codimension greater than one (that is, dimension less than 4 in $\mathbb{R}^5$) is not represented by a single equation. Instead, it may be expressed as the intersection of a set of hyperplanes. Equivalently, the points on the subspace simultaneously satisfy the equations of each of the intersecting hyperplanes – Thus, a subspace of codimension $c$ is the common zero-set of $c$ distinct linear polynomials. Finally, points on a collection of $n$ linear subspaces will satisfy any polynomial formed as the product of linear polynomials, where each linear polynomial represents a hyperplane plane through one of the subspaces. Thus, the collection of $n$ subspaces forms the zero set of a collection of degree $n$ polynomials, as claimed.

Although by the above argument each of the polynomials factors into linear factors, we will not use this condition, since it is not preserved in the presence of slight perturbations (noise) applied to the points. The polynomials that we find will not exactly factor, hence their zero set (the algebraic variety) will not precisely consist of a set of linear subspaces. We will deal with this difficulty later.

Although in the case of degenerate motions, the data points (namely the columns of $\widehat{\mathtt{W}}$) will generally lie in a subspaces of $\mathbb{R}^5$ of dimension less than 4, we nevertheless obtain good results by fitting to dimension 4 subspaces. Consequently, for the present we limit the discussion to fitting the points to a codimension-1 variety. This means that we need only find a single polynomial that fits the points.

### 2.2.2 Fitting points to polynomials

In an abstract context, we are given a set of points $\boldsymbol{w}_i$ in $\mathbb{R}^5$ and we wish to find a homogeneous polynomial of degree $n$ (in 5 variables) that fits the points. Recall that $n$ is the expected number of independent motions. To consider a more familiar problem, suppose we want to fit a homogeneous polynomial of degree $n$ in three variables to a set of points. For simplicity, let $n = 2$. The general homogeneous polynomial of degree 2 in three variables is $ax^2 + by^2 + cz^2 + dxy + exz + fyz$. This is in fact the equation of a conic in the projective plane $\mathcal{P}^2$. Now, suppose that a point $(x_i, y_i, z_i)$ lies on this curve. We substitute into this polynomial and equate to zero to obtain a linear equation in the six unknowns $a, \ldots, f$. It does not matter that the polynomial is non-linear in the variables $x$, $y$ and $z$; it is linear in the coefficients that we need to determine. Given sufficiently many points (in this case 5) on the curve we may solve a set of linear equations to determine the coefficients $a, \ldots, f$. With more than 5 equations we find a least-squares fit to the points. This is a common procedure of algebraic fitting of a curve to a conic.

The process extends naturally to fitting any number of points in any space $\mathbb{R}^K$ to a curve of degree $n$. The pro-

cess consists of generating all the possible $M_n$ monomials of degree $n$ in the $K$ variables, i.e. $w_1^{n_1} w_2^{n_2} \cdots w_K^{n_K}$, which we can stack into a vector $\widetilde{\boldsymbol{w}} \in \mathbb{R}^{M_n}$. A general homogeneous polynomial of degree $n$ is a combination of these monomials with certain coefficients, i.e. $\sum c_{n_1,\ldots,n_K} w_1^{n_1} w_2^{n_2} \cdots w_K^{n_K} = \boldsymbol{c}^\top \widetilde{\boldsymbol{w}}$. Each point required to satisfy the polynomial will lead to a linear equation in these coefficients $\boldsymbol{c}$, and sufficiently many points will allow us to determine $\boldsymbol{c}$ as the least-squares solution of

$$\widetilde{\mathtt{W}}^\top \boldsymbol{c} = 0, \tag{9}$$

where the columns of $\widetilde{\mathtt{W}} = [\widetilde{\boldsymbol{w}_1} \cdots \widetilde{\boldsymbol{w}_P}] \in \mathbb{R}^{M_n \times P}$ contain all the monomials of degree $n$ generated by the columns of $\hat{\mathtt{W}}$.

### 2.2.3 Motion segmentation by polynomial differentiation

At this stage, we have a homogeneous degree-$n$ polynomial $p$ in 5 variables fitting a set of points $\boldsymbol{w}_i$. Ideally, the polynomial $p$ factors into $n$ linear factors, each one corresponding to a hyperplane in $\mathbb{R}^5$. The present task is to partition the points $\boldsymbol{w}_i$ according to which of these hyperplanes they lie on (or near). With inexact measurements, the polynomial $p$ will not exactly factor into linear factors. However, the variety it defines (its zero-set) will *approximate* a set of codimension-one subspaces, or hyperplanes.

We consider the derivative of $p$. This is a 5-vector

$$Dp = (\partial p/\partial x_1, \ldots, \partial p/\partial x_5)$$

where $p$ is a polynomial in the variables $x_i$. The key observation is that this derivative, when evaluated at a point $\boldsymbol{w}$ lying on the variety defined by $p$, yields the normal vector to the variety at that point.

Considering still the ideal case, if two points $\boldsymbol{w}_i$ and $\boldsymbol{w}_j$ lie on the same hyperplane, then $Dp(\boldsymbol{w}_i)$ and $Dp(\boldsymbol{w}_j)$ will be vectors in the same direction, whereas if they lie on different hyperplanes, these direction vectors will be different. In fact, since $p$ is a homogeneous polynomial, the hyperplanes in question will be linear codimension-1 subspaces passing through the origin. Therefore, two such hyperplanes are identical if and only if they have the same unit normal vector. Therefore two data points $\boldsymbol{w}_i$ and $\boldsymbol{w}_j$ will lie on the same hyperplane if and only if the angle between $Dp(\boldsymbol{w}_i)$ and $Dp(\boldsymbol{w}_j)$ is zero (or $180°$).

This suggests a procedure for partitioning the points $\boldsymbol{w}_i$. When $p$ does not factor into linear factors, the normal vectors $Dp(\boldsymbol{w}_i)$ and $Dp(\boldsymbol{w}_j)$ will not be exactly the same. Nevertheless, we may define a similarity measure for two such points, as follows.

$$S_{ij} = \cos^2(\theta_{ij})$$

where $\theta_{ij}$ is the angle between the two vectors $Dp(\boldsymbol{w}_i)$ and $Dp(\boldsymbol{w}_j)$. Then $S_{ij} \approx 1$ if points $\boldsymbol{w}_i$ and $\boldsymbol{w}_j$ are from the same motion (and so belong to the same hyperplane), whereas $S_{ij} < 1$ if they belong to different motions.

Given the so-defined similarity matrix $S \in \mathbb{R}^{P \times P}$, one can apply any spectral clustering technique to obtain the segmentation of the feature points, e.g. [21]. Once the features have been clustered, one can estimate the motion and structure of each moving object using the standard factorization approach for affine cameras, e.g. [14]. We therefore have the following algorithm for motion estimation and segmentation from multiple affine views.

**Algorithm 1 (Multiframe motion segmentation algorithm)** *Given a matrix $\mathtt{W} \in \mathbb{R}^{2F \times P}$ containing $P$ feature points in $F$ frames (possibly with missing data)*

1. ***Projection**: Project the data onto a five-dimensional space using either SVD (complete data) or power factorization (incomplete data) to obtain a (complete) data matrix $\hat{\mathtt{W}} = [\boldsymbol{w}_1, \ldots, \boldsymbol{w}_P] \in \mathbb{R}^{5 \times P}$.*
2. ***Multibody motion estimation via polynomial fitting**: Compute the left null-vector $\mathbf{c}$ of the embedded data matrix $\widetilde{\mathtt{W}} = [\widetilde{\boldsymbol{w}_1}, \ldots, \widetilde{\boldsymbol{w}_P}] \in \mathbb{R}^{M_n \times P}$ to obtain a polynomial $p$ representing the $n$ motion subspaces.*
3. ***Feature clustering via polynomial differentiation**: Cluster the feature points by applying spectral clustering to the similarity matrix $S_{ij} = \cos^2(\theta_{ij})$, where $\theta_{ij}$ is the angle between the vectors $Dp(\boldsymbol{w}_i)$ and $Dp(\boldsymbol{w}_j)$ for $i, j = 1, \ldots, P$.*
4. ***Motion Estimation**: apply the standard factorization approach for affine cameras to each one of the $n$ group of features to obtain motion and structure parameters.*

### 2.2.4 Degenerate and dependent motions

Since our method is particularly intended to handle degenerate and dependent motions, in which the different linear subspaces interesect non-trivially, or have smaller dimension than 4, we need to understand why the proposed method works in this case.

In the case of dependent (but full-dimension) motions, the hyperplane subspaces will have dimension 4 in $\mathbb{R}^5$, as usual. Two such subspaces will have distinct normals, even if they intersect in some non-zero subspace. Since the normals are different, our method will work effectively.

In the case of degenerate motions, some of the subspaces may have smaller dimension than the expected dimension 4. Such a subspace may be defined as the intersection of more than one hyperplane. By choosing a single polynomial $p$ to fit all subspaces, we effectively choose a single one of these hyperplanes containing the low-dimensional subspace.[3] As long as this hyperplane does not correspond

[3] It was shown in [17] that even though for degenerate motions the polynomial $p$ may not be factorizable, its derivative at a point in one of the

with the hyperplane defining one of the other motions, there will be no problem with using such a hyperplane to segment the degenerate motion. Generically this favorable condition will apply. In fact a degenerate motion can always be separated from a non-degenerate one, unless the low-dimensional subspace lies inside the hyperplane corresponding to the non-degenerate motion.

**Modelling low-dimension subspaces** We refer the reader to [17] for a possible extension of the above techniques to model low-dimension subspaces explicitly. To do this, instead of fitting the data to a single polynomial, we need to compute a set of independent polynomials $p_k$ that fit the points. Derivatives of these polynomials represent a set of vectors generating a normal subspace, rather than a single normal. The similarity measure defined previously may be generalized to measure the largest principal angle between the normal subspaces corresponding to different points. Details are left to a more complete report.

# 3. Experiments on real images

In this section, we test our 3-D motion segmentation algorithm on various motion sequences, some of them shown in Figure 1. The sequences were chosen to display the performance of our algorithm in the following situations:

1. *Sequences with missing data*: we choose some sequences with more than 10 frames for which data points are not present in all the frames. In such sequences, typically, about 30% of the data was missing.
2. *Sequences with degenerate and nondegenerate motions*: we choose sequences with full motions (Can-Book), linear motions (Castle), planar motions (Castle), screw motions (Puma) and combinations of linear and full motions (3-Cars), so as to display the performance of the algorithm both in degenerate and nondegenerate motion configurations.
3. *Sequences with perspective effects*: most of the chosen sequences present noticeable perspective effects, due to large depth variations, or forward motions. Our algorithm does not model such effects, because it is based on the affine projection model.
4. *Sequences with transparent motions*: except for the 3-Cars and Can-Book sequences, which truly contain multiple motions, all the other sequences are actually static. We made them dynamic by generating a second set of correspondences with the $x$ and $y$ coordinates interchanged. In this fashion, the compound correspondences have no spatial separation, hence motion is the only cue for clustering.
5. *Sequences with minimum number of frames*: we choose some sequences where only three views are available, e.g. Can-Book, Wilshire and Tea-Tins. Three is the minimum number of views needed so that our algorithm can project the data onto $\mathbb{R}^5$. Notice that this number does not depend on the number independent motions. Also, remember that the CK algorithm needs at least four frames for two independent motions, and that the minimum number of frames is $2n$ for $n$ independent motions.

Table 1 shows the segmentation results for each one of the sequences. Notice that the sequences have rather different number of feature points and number of frames. However, in all the cases the algorithm gives a misclassification error of less than 5.5%. Furthermore, notice that the algorithm achieved perfect classification for some of the sequences. It is important to emphasize here that, given the matrix of projected data, $\hat{\mathtt{W}} \in \mathbb{R}^{5\times P}$, we applied our algebraic motion segmentation algorithm directly. That is, we did not apply any pre or post processing to the data, nor did we remove outliers or improve the segmentation by iterative refinement. However, one may further improve the estimates of our algorithm by using it to initialize any iterative technique that alternates between motion estimation and data clustering, such as K-subspace [7] or EM for mixtures of PCA's [13].

Figure 2 shows the shape interaction (similarity) matrix used by the CK algorithm for the Can-Book sequence. We built the interaction matrix by using different values for the rank of the $\mathtt{W}$ matrix. It can be noticed that, because of degenerate motions and partial dependencies among the motion subspaces, the interaction matrices of the CK algorithm do not give much information about the clustering of the correspondences. In contrast, our algorithm computes a similarity matrix from the angles between the motions subspaces, after extracting a basis for them in a purely algebraic fashion. Exploiting the algebraic structure of the mixture model, enabled us to define a much better measure of similarity among motion subspaces.

# 4. Conclusions

This paper has presented a new algorithm for 3-D motion segmentation from multiple affine views, which deals with both complete and incomplete data, and applies to independent and partially dependent (degenerate) motions. The algorithm uses power factorization to project the data onto a five dimensional space, and GPCA to cluster the projected subspaces. Our experiments demonstrated that our algorithm works also for projective sequences, even though it is not designed for that case. Overall it gives a misclassification error of less than $5.5\%$ with up to $30\%$ missing data. Open research avenues include extending the algorithm to projective reconstruction of multiple rigid-body motions from multiple perspective views.

motion subspaces still gives a normal vector to that subspace. It is the hyperplane associated with this normal vector the one we are referring to here, which exists for any $p$, factorizable or not.

Table 1: Misclassification error for different sequences.

| Sequence | Points | Frames | Motions | Error |
|---|---|---|---|---|
| Boat | 686 | 11 | 2 | 2.19% |
| Can-Book | 170 | 3 | 2 | 1.18% |
| Wilshire | 200 | 3 | 2 | 5.50% |
| Tea-Tins | 84 | 3 | 2 | 1.19% |
| NEC | 82 | 8 | 2 | 0.00% |
| 3-Cars | 173 | 15 | 3 | 4.62% |
| Puma | 64 | 16 | 2 | 0.00% |
| Castle | 56 | 11 | 2 | 0.00% |

# References


[1] T.E. Boult and L.G. Brown. Factorization-based segmentation of motions. In *Proc. of the IEEE Workshop on Motion Understanding*, pages 179–186, 1991.

[2] J. Costeira and T. Kanade. Multi-body factorization methods for motion analysis. In *CVPR*, pages 1071–1076, 1995.

[3] F. De la Torre and M. J. Black. Robust principal component analysis for computer vision. In *CVPR*, pages 362–369, 2001.

[4] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge, 2000.

[5] R. Hartley and F. Schaffalitzky. PowerFactorization: an approach to affine reconstruction with missing and uncertain data. In *Australia-Japan Advanced Workshop on Computer Vision*, 2003.

[6] R. Hartley and R. Vidal. The multibody trifocal tensor: Motion segmentation from three perspective views. In *CVPR*, 2004.

[7] J. Ho, M.-H. Yang, J. Lim, K.-C. Lee, and D. Kriegman. Clustering apperances of objects under varying illumination conditions. In *CVPR*, volume 1, pages 11-18, 2003.

[8] K. Kanatani. Motion segmentation by subspace separation and model selection. In *CVPR*, volume 2, pages 586–591, 2001.

[9] K. Kanatani. Evaluation and selection of models for motion segmentation. In *Asian Conference on Computer Vision*, pages 7–12, 2002.

[10] K. Kanatani and C. Matsunaga. Estimating the number of independent motions for multibody motion segmentation. In *ECCV*, pages 25–31, 2002.

[11] K. Kanatani and Y. Sugaya. Multi-stage optimization for multi-body motion segmentation. In *Australia-Japan Advanced Workshop on Computer Vision*, pages 335–349, 2003.

[12] H. Y. Shum, K. Ikeuchi, and R. Reddy. Principal component analysis with missing data and its application to polyhedral object modeling. *IEEE Transactions on PAMI*, 17(9):854–867, 1995.

[13] M. Tipping and C. Bishop. Mixtures of probabilistic principal component analyzers. *Neural Computation*, 11(2), 1999.

[14] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography. *IJCV*, 9(2):137–154, 1992.

[15] P. H. S. Torr. Geometric motion segmentation and model selection. *Phil. Trans. Royal Society of London A*, 356(1740):1321–1340, 1998.

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

[17] R. Vidal, Y. Ma, and J. Piazzi. A new GPCA algorithm for clustering subspaces by fitting, differentiating and dividing polynomials. In *CVPR*, 2004.

[18] R. Vidal, Y. Ma, and S. Sastry. Generalized principal component analysis (GPCA). In *CVPR*, volume 1, pages 621–628, 2003.

[19] R. Vidal, Y. Ma, S. Soatto, and S. Sastry. Two-view multibody structure from motion. *International Journal of Computer Vision*, 2004.

[20] R. Vidal and S. Sastry. Optimal segmentation of dynamic scenes from two perspective views. In *CVPR*, volume 1, pages 281-286, 2003.

[21] Y. Weiss. Segmentation using eigenvectors: a unifying view. In *CVPR*, pages 975–982, 1999.

[22] Y. Wu, Z. Zhang, T.S. Huang and J.Y. Lin. Multibody grouping via orthogonal subspace decomposition. In *CVPR*, volume 2, pages 252–257, 2001.

[23] L. Zelnik-Manor and M. Irani. Degeneracies, dependencies and their implications in multi-body and multi-sequence factorization. In *CVPR* , volume 2, pages 287-293, 2003.


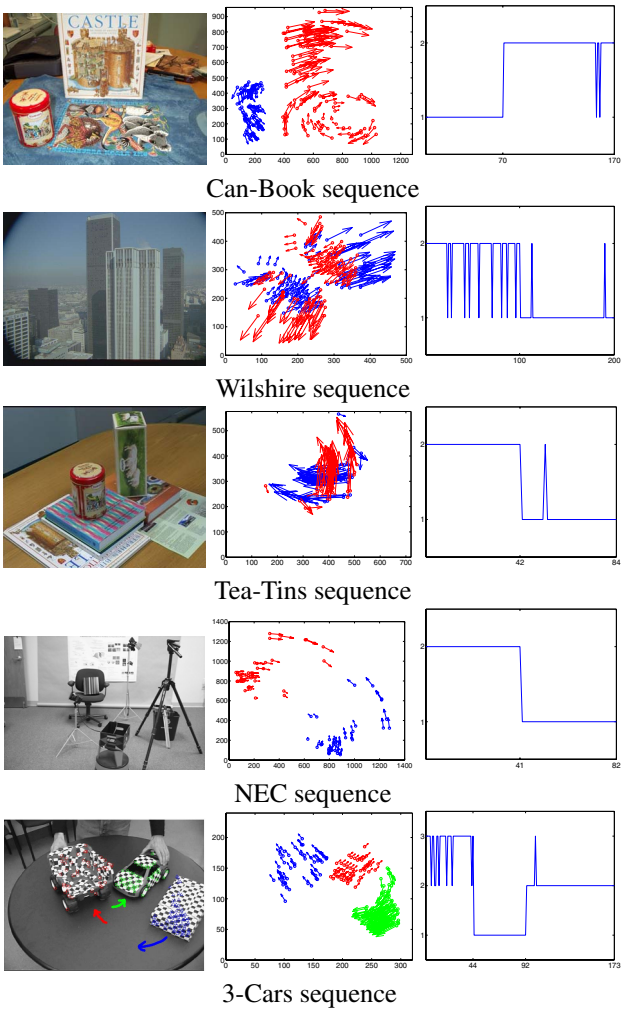


Figure 1: Motion segmentation results. Left: first frame of each sequence. Center: displacement of the correspondences between two views. Right: clustering of the correspondences given by our algorithm.

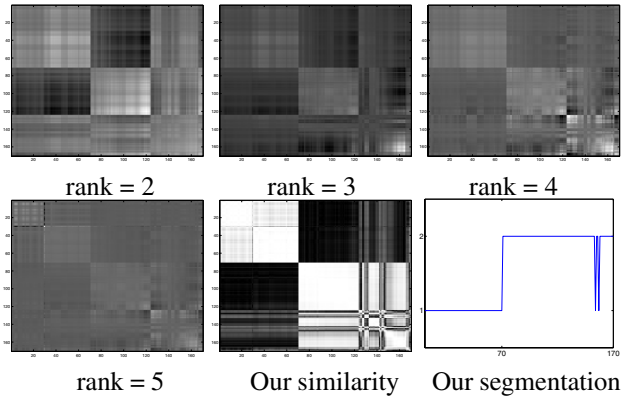


Figure 2: Similarity/Interaction matrices from the Costeira and Kanade algorithm for different rank approximations and from our algorithm for the Can-Book sequence.