Factorization-based Segmentation of Motions

Terrance E. Boult and Lisa Gottesfeld Brown
Vision and Robotics Laboratory
Dept. of Computer Science, Columbia University, NYC NY 10027
boult@cs.columbia.edu brown@division.cs.columbia.edu

Abstract

This paper addresses the problem of motion segmentation using the Singular Value Decomposition of a feature track matrix. It is shown that, under general assumptions, the number of numerically nonzero singular values can be used to determine the number of motions. Furthermore, motions can be separated using the right singular vectors associated with the nonzero singular values. A relationship is derived between a good segmentation, the number of nonzero singular values in the input and the sum of the number of nonzero singular values in the segments. The approach is demonstrated on real and synthetic examples. The paper ends with a critical analysis of the approach.

1 Introduction and Previous Work

The use of "motion" information has been around for many years, and considerable research progress has been made. Little of this work on motion, however, addresses the issue of segmenting the motions in a scene into different components but rather assumes, often implicitly, that that the images have been previously segmented into their constituent motions. Previous work on motion segmentation includes:

- intensity based image segmentation using known depth [Peleg and Rom, 1990] [Yamamoto, 1990],
- looking for "boundaries" in "optic-flow fields" (i.e. assuming locally uniform motion, but not knowledge of depth), [Shizawa and Mase, 1990] [Adiv, 1985], [Murray and Buxton, 1987],
- clustering in some type of *a priori* defined parametric motion space [Fennema and Thompson, 1979], [Dickmanns, 1989],
- techniques using Markov Random Fields to find both a segmentation and description of a motion sequence [Heita and Bouthemy, 1990], [Subrahmonia *et al.*, 1990],
- and a technique which looks for 2 motion components in an image sequence [Bergen *et al.*, 1990b], [Bergen *et al.*, 1990a].

Much of this work has made restrictive assumptions on the scene/motion to allow segmentation or assumes considerable *a priori* information (e.g. a depth map).

The approach we introduce in this paper, segments the motions in a scene into their different rigid body motions without knowledge of camera or object motion, shape, or depth. We assume an input of tracked features and, except for the smoothness necessary for that tracking, the approach to segmentation and motion/shape recovery does not require smoothness assumptions on either the objects or the motion. To obtain these desirable features we use the elegant "factorization approach" to shape/motion recovery. The idea of using SVD factorization of motion tracks was recently introduced (see [Tomasi and Kanade, 1990a],[Tomasi and Kanade, 1990b]) and is discussed in more detail elsewhere in these proceedings.

The paper is organized as follows: we first discuss the SVD in a little more detail. We then present a short review of Tomasi and Kanade's ground breaking work on SVD factorization and motion recovery. This material in hand, we describe our approach in Section 3. Section 4 presents some experimental testing of the approach, followed by a critical analysis in Section 5. We end with some conclusions and a discussion of future work.

2 Background

In this section we present background material on the singular value decomposition and on the motion factorization technique.

2.1 Some properties of the SVD

Assume, without loss of generality, I is an $m \times p$ matrix, $m \geq p$. By computing the Singular Value Decomposition (SVD) of the input I we can obtain 3 matrices L, Σ, R such that $I = L\Sigma R^{\mathsf{T}}$. Furthermore, $L = [l_1, l_2, \ldots l_p]$ is an $p \times p$ orthogonal matrix, $R = [r_1, r_2, \ldots r_p]$ is an $n \times p$ orthogonal matrix, and Σ is a diagonal $p \times p$ matrix $diag(\sigma_1, \sigma_2, \ldots \sigma_p)$ with $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_p \geq 0$. The value σ_i is referred to as the ith singular value, and the column vectors l_i, r_i are, respectively, the ith left singular vector and the ith

right singular vector.

We now state some properties of the SVD which will be useful in the remainder of the paper. There is a strong relationship between SVD and eigenvalues/eigenvectors. In particular, the singular values are the eigenvalues of $I^\mathsf{T} I$ and the right singular vectors are its eigenvectors. Since $I^\mathsf{T} I$ is a symmetric matrix, the singular values are also the squares of the eigenvalues of I. To help in visualizing how the SVD extracts much of the structure inherent in a matrix, it is also helpful to know that the singular values of the matrix I are precisely the lengths of the semi-axes of the hyperellipsoid E defined by $E = \{y | y = Ix, \|x\|_2 = 1\}$. Furthermore, the location of the axes of this ellipsoid are given by the columns of L. Thus the SVD is strongly related to the principal component analysis of the data in I.

The relationship between SVD and principal components explains why we are able to perform motion segmentation using clustering of the right singular vectors. Principal component analysis is generally performed to reduce the dimensionality of a data set with many interrelated variables to a much smaller set, the principal components, while retaining as much of the original variation as possible. Each principal component is composed of a linear function α which operates on the vector x of random variables and maximizes the variance (and is uncorrelated with previously found components). The kth principal component is given by $\alpha_k^T x$ where α_k is an eigenvector of the covariance matrix of x corresponding to its kth eigenvalue. Since the right singular vectors of the SVD are the eigenvectors of $I^\mathsf{T} I$, they are precisely the principal components. In our case, each random variable is the location of a feature point given at different times. Random variables are related in a definite way according to whether they are part of the same motion. This is ultimately expressed in the principal components.

If the SVD of I is such that $\sigma_1 \geq \ldots \geq \sigma_k > \sigma_{k+1} = \ldots \sigma_p = 0$, then we know, $Rank(I) = k$ and $I = \sum_{i=1}^{k} \sigma_i l_i \otimes r_i^\mathsf{T} = \hat{L}_k \hat{\Sigma}_k \hat{R}_k^\mathsf{T}$. When the matrix I is noisy, the issue of determining "numerical" rank, call it $\mathcal{R}(I)$, is more difficult and will be discussed in section 3.4.

Finally, we comment on the computational complexity of the SVD algorithm. For our needs, the straightforward approach to the SVD costs around $7mp^2 + 4p^3 + O(p^2 + mp)$ FLOPS for a matrix with m rows and p columns. Code for computing the SVD can be found in any good numerical package (LINPACK, EISPACK, NAG, IMSL). We use a locally modified version of the code from Numerical Recipes in C [Press *et al.*, 1988].

2.2 Background on the shape/motion factorization technique.

Our discussion of the factorization technique has been broken into four terse subsections. We will treat both the 2D case (motion restricted to a plane) and the 3D case. The rationale for this is that the 2D case is significantly easier to visualize and present while the methods are very similar in practice.

All of the conceptual content of this section follows from the work of Tomasi and Kanade. For the 2D problem we follow [Tomasi and Kanade, 1990c] and for the 3D problem we follow [Tomasi and Kanade, 1991]. This section is intended to introduce notation for our development, and does not thoroughly explore the factorization-based approach.

The following assumptions are needed to make the approach feasible:

- the imaging system is orthographic,
- there are at least 3 frames
- there exists a feature tracker which can, given the image sequence, solve the frame to frame correspondence problem and track points over extended periods of time.
- each object in motion yields ≥ 3 points which are tracked over the entire image sequence,
- a motion must have a nonzero rotational component.

Again, these are minimal assumptions. In general performance will be better with more frames and more points taken over a wide change in rotational angles.

2.2.1 Input representation: track matrix of shapes in motion

The input to the factorization procedure is a matrix I, representing tracks, i.e. image positions of feature points over time. We assume there are p feature points over f frames, and that in frame i point j is at pixel location $(u_{i,j}, v_{i,j})$ in the image plane. These can be interpreted as a pair of matrices U and V giving the horizontal and vertical component of the point's location respectively. Let U' and V' represent the same position matrices where each row has been shifted so as to have mean 0 (i.e. independently subtract the centroid of each frame from every element of that row). It is not necessary that every frame of the motion sequence be in the input matrix. In particular, while dense sampling may be necessary for tracking, the input matrix can be made from significantly fewer frames.

In the 2D case, our input matrix is $f \times p$ and is simply U. For the 3D case we assume an input matrix W which is $(2f) \times p$ with $W = \begin{smallmatrix} U' \\ V' \end{smallmatrix}$. If the dimensionality of the input does not matter, we will use I to represent the input track matrix of size $m \times p$.

2.2.2 Representing Shape

We represent a point in the world coordinate system as ρ_i. For the projection of this point in the 2D case, we adopt a homogeneous representation $(x_i, z_i, 1)$, so we can represent translation. In 3D, where translation is removed by subtracting the centroid of each frame, we use the representation (x_i, y_i, z_i). We can then collect p of these point into a shape matrix, say S of size $3 \times p$, by considering each point as a column in the matrix.

2.2.3 Representing Motion

In 2D let α_i be the angle between the X axis and the camera in frame i. The projection of point ρ_j into the image is given by $u_{i,j} = [\cos(\alpha_i),\ \sin(\alpha_i),\ t_i] \cdot [x_j, z_j, 1]^\mathsf{T}$, where t_i is the projection, onto the frame f_i, of the translation vector (as measured from the first frame). We can collect this into a motion matrix, M_{2d}.

In the 3D representation, we assume a fixed world coordinate system. In this coordinate system let $\vec{r}_i$ be a unit vector (represented by its endpoint) which is aligned with the image rows in frame f_i, and let $\vec{c}_i$ be the unit vector aligned with the image columns in frame f_i. Given this, we see that $U'_{i,j} = \vec{r}_i \cdot p_j$ and $V'_{i,j} = \vec{c}_i \cdot p_j$. Thus we can build a $(2f) \times 3$ motion matrix taking $\vec{r}_i$, $i = 1..f$ as the first f rows and $\vec{c}_i$, $i = 1..f$ as rows $(f+1) \ldots 2f$.

With this notation we have $U = M_{2d}S$ and $W = M_{3d}S$, or letting the dimension be implicit $I = MS$.

2.2.4 Factoring the input matrix

Let us consider the SVD of I, and let $\hat{\Sigma}$ be the submatrix of Σ containing the numerically nonzero singular values. Let $\hat{L}$ ($\hat{R}$) be the associated rows (columns) of L (R respectively). Then, as is always true for the SVD, $I = \hat{L}\hat{\Sigma}\hat{R}^\mathsf{T}$. We drop the $\hat{}$ for simplicity and henceforth the exact interpretation of L Σ or R will be given by the context.

Note that the dimensionality of the L and R matrices are exactly the same as the dimensionality of the shape and motion matrix because if the motion/shape is not degenerate, then $\mathcal{R}(I) = 3$. If we let $M' = L \cdot \Sigma^{\frac{1}{2}}$ and $S' = \Sigma^{\frac{1}{2}} \cdot R^\mathsf{T}$, then we even have the same form $I = M' \cdot S'$. In fact, these matrices are, up to an affine transformation, exactly the shape and motion matrices, i.e. there exists a nonsingular A such that $M = M' \cdot A, S = A^{-1}S'$. The exact procedure for obtaining A depends on the dimension as detailed in [Tomasi and Kanade, 1990c] and [Tomasi and Kanade, 1991]. Intuitively, A is obtained by requiring the resulting motion matrix to have the proper orthogonality properties.

3 Description of factorization-based segmentation of multiple motions

In this section we first establish a relationship between the number of singular values and the number of motions. We present a simple example of the approach on real data. We end this section with a more technical discussion of the determination of rank and our cluster analysis process.

3.1 Representing multiple motions and the related number of singular values

When there is a single motion we consider the same representation as in [Tomasi and Kanade, 1990c] and [Tomasi and Kanade, 1991], (see Section 2.2) which results in a matrix formulation of $I = MS$, where I is $m \times p$, M is $m \times 3$, S is $3 \times p$ and $m = 2f$ or f depending on the dimension.

We now consider the case where there are two motions. Let M' and M'' be motion matrices of size $m \times 3$. Let the associated shape points be S', S'' consisting of p' and p'' points respectively. Assume these points are associated (in some order) with tracks $t_i, i = 1, \ldots, (p' + p'')$.. Then we can represent the track matrix $I = \mathcal{M}\mathcal{S}$, where $\mathcal{M} = [M'|M'']$ with $|$ being matrix concatenation. Each column $\mathcal{S}_j$ is of the form $[S'_{j,1}, S'_{j,2}, S'_{j,3}, 0, 0, 0]^\mathsf{T}$ if track t_j is associated with point S'_j and of the form $[0, 0, 0, S''_{j,1}, S''_{j,2}, S''_{j,3}]^\mathsf{T}$ if track t_j is associated with point S''_j.

That such a decomposition accounts for the track matrix is easily shown. Generalizing this to N motions is straightforward, resulting in a motion matrix $\mathcal{M}$ being $m \times 3N$ and the shape matrix being $3N \times \sum_{k=1}^{k=N} p_k$.

3.1.1 Multiple motions and factorization: Definitions and observations

In some ways the motion matrix M is, in reality, not a motion at all. It is not an optic flow; it is not a parametric equation defining the path of the object; it is not even, necessarily, the path taken by any point in the scene. Instead, each row in the motion matrix can be interpreted as the transformation which takes a world point, in a fixed frame of reference, to its projection in the associated image frame. A motion is then a subspace of f-dimensional space defined by the span of the columns of M. The path (i.e. $x(t), y(t), z(t)$) taken by any single object point is represented as a single point in this f-dimensional space, and hence we call this *path space*. A rigid body motion requires all points being considered to follow paths which are within the span of the columns of the associated motion matrix M. Since we cannot directly measure M, we will generally consider the information in the columns of I, to define the observable motion. We will use motion to mean either real motion (M) or observable motion (I) where the context should disambiguate the interpretation. We

now explore the ramifications of this representation.

In what follows, let $I_i, i = 1..N$ be track matrices of different motions with $\mathcal{R}(I_i) > 0$. Let M_i be the associated motion matrices and S_i the shapes. Assume no noise so that $\mathcal{R}(I) = Rank(I)$.

Definition 1 *A track matrix I corresponds to a motion if and only if (hereafter* **iff** *) its columns span a nonempty subset of path space*

Observation 1-1 *A track matrix I corresponds to a motion* **iff** $\mathcal{R}(I) > 0$.

Definition 2 *Assume I_1 is associated with a single motion (M_1 and S_1 assumed to be $f \times 3$ and $3 \times p$ respectively.) I_1 is called nondegenerate* **iff** $\mathcal{R}(I_1) = 3$. *That is, a nondegenerate motion spans a 3 dimensional subspace of path space. (Similarly define nondegenerate M_1 and S_1)*

Observation 2-1 *For N motions $\mathcal{R}(M) \leq 3N$, $\mathcal{R}(S) \leq 3N$ and hence $\mathcal{R}(I) \leq 3N$.*

Observation 2-2 *A nondegenerate track matrix is the product of a nondegenerate motion and nondegenerate shape. Nondegenerate motions require at least 3 frames. Nondegnerate shapes requires at least 3 points.*

Definition 3 *The track matrix I_2 contains a different motion from that of I_1* **iff** *the span of the columns of I_2 is not contained within the span of the columns of I_1. That is I_2 has a motion different from I_1* **iff** *there are points in path space which could be associated with I_2 which could never be generated by the motion underlying I_1.*

Observation 3-1 *It follows that I_2 contains a motion different from that of I_1* **iff** $\mathcal{R}(I_1|I_2) > \mathcal{R}(I_1)$.

Definition 4 *Two motions I_1 and I_2 are said to be different motions* **iff** *they each contain a motion different from the other. (Note that if the span of I_1 is a proper subset of the span of I_2 then I_2 contains a motion different from I_1, but not the other way around).*

Observation 4-1 *It follows that I_2 and I_1 are different motions* **iff** $\mathcal{R}(I_1|I_2) > \max(\mathcal{R}(I_1), \mathcal{R}(I_2))$.

Observation 4-2 *Note that such definitions, in terms of subspaces, may not totally capture the intuitive notion of "different motions". If there are two well separated clusters within a single subspace, humans might interpret them as different motions. This definition is saying that two motions are the same if there exists a rigid body interpretation that makes them the same. It does not preclude using additional information to further label submotions with different labels. (In fact, the algorithm to be presented can often "segment" these well separated clusters, however it knows that they are, according to the above definitions, 2 clusters from the same motion.)*

Observation 4-3 *Adding a single track from a "different" motion to I must increase the rank of I by 1.*

Observation 4-4 *If track matrix I is decomposed into 2 parts I_1 and I_2, then $\mathcal{R}(I) \leq \mathcal{R}(I_1) + \mathcal{R}(I_2)$.*

Definition 5 *I_1 and I_2 are called linearly independent motions* **iff** $\mathcal{R}(I_1|I_2) = \mathcal{R}(I_1) + \mathcal{R}(I_2)$.

Observation 5-1 *A track matrix from N linearly independent motions will have $\mathcal{R}(I) = 3N$.*

Observation 5-2 *If a track matrix I, composed of N linearly independent nondegenerate motions (containing at least 4 points each), is decomposed into 2 parts I_1 and I_2, then $\mathcal{R}(I) = \mathcal{R}(I_1) + \mathcal{R}(I_2)$* **iff** *for every set of shape points $\{S\}_i$ associated with a single motion $\{M\}_i$ in I, the tracks from these points/motions are contained entirely in either I_1 or I_2. In simpler terms, under general motion assumptions the segmentation of I is good (but not necessarily complete) if and only if the number of nonzero singular values before segmentation is the same as the sum of the nonzero singular values of the segments.*

To prove this last observation, let $\{M_i S_i\}_j$ be the tracks from motion i that appear in segment j, with the understanding that if no points of motion i appear in segment j, then $\{M_i S_i\}_j$ is empty. Also let segment $j = 0$ refer to the original (unsegmented data). By definition we have, modulo some column permutations, $I_j = \{M_1 S_1\}_j | \ldots | \{M_N S_N\}_j$, which implies $\mathcal{R}(I_j) = \sum_{i=1}^{i=N} \mathcal{R}(\{M_i S_i\}_j)$. Thus we can restate our observation as

$$\sum_{i=1}^{i=N} \mathcal{R}(\{M_i S_i\}_0) \leq \sum_{i=1}^{i=N} \mathcal{R}(\{M_i S_i\}_1) + \sum_{i=1}^{i=N} \mathcal{R}(\{M_i S_i\}_2) \quad (1)$$

with equality holding if and only if the segmentation is good. That a good segmentation implies equality follows directly from the above equation and the definition of linear independence. To show that equality implies a good segmentation we argue as follows. First, note that since the number of nonzero singular values is always nonnegative, the splitting of any motion cannot reduce the right hand sum. Since each motion contains at least 4 points either all of its points are in one segment, in which case it contributes a value of n to the right hand side, or its points are split across the partition and it contributes a minimum of $n + 1$ to the right hand sum.

Noting that each motion's contribution to the left hand side is exactly n, the observation follows.

One of the reasons for proving the observation as we did is that it provides insight to what may happen if the assumptions of the observation are violated. If there are motions with only 3 points, then a segmentation may fail in an undetectable manner. More of a potential problem is that if there are motions which are linearly dependent, then there may exist incorrect segmentations into two groups which will *not* increase the total number of singular values. Luckily, the observation often holds even if there are linearly dependent motions. If some of the motions are dependent, the equality will hold if and only if all the points in the linearly dependent subset of motions are grouped into a single partition.

The importance of this final observation should not be overlooked. It provides a theoretical basis for checking the segmentation. If we start with $3N$ singular values and eventually get N subsets with 3 singular values each, we know that we have a correct segmentation!

3.2 Overview of approach

The method can be summarized as the following steps:

1. Find tracks (we do not address this issue in this paper) and compute track matrix I
2. Compute SVD yielding L, Σ, R
3. Determine $\mathcal{R}(I)$ and prune columns and rows of L and R^T respectively.
4. Generate clusters, using R, to get potential segmentations.
5. For a potential segmentation, remove the 'segmented' tracks from original image yielding I_i, and compute the SVD of each separately. If $\mathcal{R}(I) = \sum_i \mathcal{R}(I_i)$, then we know the partition is good, otherwise we try the next clustering at this level. If the last clustering fails, backtrack to the call to this level.
6. Given a good partition, if a cluster has ≤ 3 singular values we compute shape and motion, else treat this as a new input matrix for the next level of segmentation and recursively goto step 2.

The first step, which we assume is handled by some other process, determines the tracks of points in the scene. We also expect an error estimate for the track information. It is not really important that the tracks are dense in either time or space, although the quality of the motion estimate and shape estimates depend on those densities respectively.

3.3 A simple 2D Example using Real Data

We now present a simple example which we will follow through as we describe the algorithm in more detail.

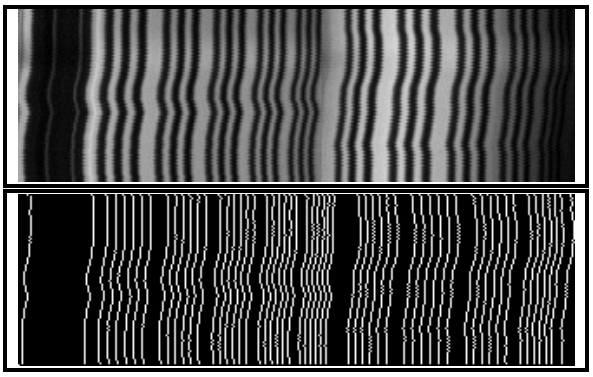

Figure 1: Top shows original epi-polar image (graylevel camera scanlines). The bottom figure shows the tracks found for 57 points over 101 frames.

The motion was obtained by moving a camera in a plane using a precision Datel rotation stage while keeping a high contrast scene in view. While the camera rotated, 2 objects in the scene were independently moved. One object (on the right) was rotated about its axis, and the other was translated with a small amount of local rotation. We obtained our epi-image by grabbing one 512 pixel scanline per frame time for 100 frames. To make the segmentation task more difficult we moved the epi's of the two "objects" closer to produce the epi-image and its associated edges shown in Figure 1. The high contrast objects allowed for easy "tracking" of features using a simple Sobel edge detector and edge linking which locally fits a quadratic to the Sobel response and tracks the maximum with subpixel precision. Tracks which were not continued from the first line to the last line were not included in the input matrix. Figure 2 shows the segmented tracks determined by the algorithm.

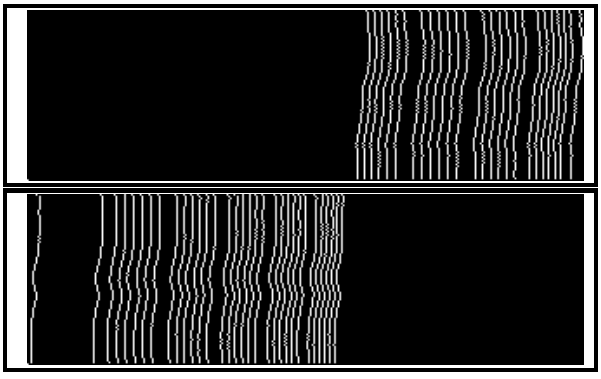

Figure 2: Segmented tracks from detected motions

3.4 Determining rank

A central part of our segmentation algorithm requires determining the rank or number of nonzero singular values for the idealized input which lead to the measured input matrix I. If, as is usually the case, our desired input matrix, say D, is perturbed by noise (er-

ror), say E, then we cannot expect the singular value decomposition of $I = D + E$ to yield the exact number of singular values of D; i.e. we need to do more than determine the number of nonzero singular values. In determining an approximation to the rank of D we will use some knowledge of E.

It can be shown that the difference between the singular values of the ideal input, $\sigma_k(D)$ and the singular values of the measured input, $\sigma_k(I)$ satisfy the following properties:

$$|\sigma_k(I) - \sigma_k(D)| \leq \epsilon\sigma_1(I), \quad \forall k \leq p, \tag{2}$$

$$|\sigma_k(I) - \sigma_k(D)| \leq \sigma_1(E) < \|E\|_F \quad \forall k \leq p, \tag{3}$$

and

$$\sum_{k=1}^{p}(\sigma_k(I) - \sigma_k(D))^2 \leq \|E\|_F^2 = \sum E_{i,j}^2 \tag{4}$$

where $\|\cdot\|_2$ and $\|\cdot\|_F$ are the second and Frobenius matrix norms respectively, and ϵ is the machine precision for computation. See [Golub and van Loan, 1983, Sect 6.5 and Cor. 8.3.2, 8.3.5] for proofs and more detail.

We now develop bounds on $\mathcal{R}(i)$. Let be k^* be the smallest integer such that $\forall k \geq k^*$, $\sigma_k(D) = 0$. Notice for $k \geq k^*$ we have $\sigma_k(I) - \sigma_k(D) = \sigma_k(I)$. Combining this observation with equations 2– 4 we have a lower bound

$$\mathcal{R}(I) \geq \text{ max } k \text{ s.t. } \begin{cases} \sigma_k(I) \geq \epsilon\sigma_1(I), \\ \sigma_k(I) \geq \|E\|_F, \text{or} \\ \sqrt{\sum_{j=k}^{j=p}\sigma_j^2(I)} > \|E\|_F \end{cases} \tag{5}$$

These lower bounds on $\mathcal{R}(I)$ require a conservative estimate (over estimate) of $\|E\|_F$. Note this relies on relatively weak knowledge of the noise E, and makes no assumption about the shape of the error distribution. In the examples in this paper we assume $\|E\|_F \leq \delta_1$, where δ_1 is set to be greater than or equal to the expected point RMS error. For real datasets, the expected RMS is computed by computing the SVD of a known nondegenerate single motion and subtracting the reconstruction (using first 3 singular values) from the original track matrix. For the example images estimated RMS was .04.

In our upper bound we assume that there is a computable predicate $Np(E)$, which when applied to a matrix E returns true only if E "must" be considered pure noise. As we will see later, because we are interested in tracks of motion, it is quite reasonable to assume that any matrix must be noise if $|E_{i,j}| < \delta_2, \quad \forall i, j$. For the real examples we have set a very conservative estimate of $\delta_2 = .0016$. (Our upper bounds would be smaller if we were less conservative). Another reasonable noise predicate can be obtained by assuming that for all observable I we will have $\|E\|_F \geq \delta_2$.

Assuming that the predicate $Np(E)$ is conservative (i.e. it never returns TRUE if the matrix E could be valid data), we can get an upper bound on $\mathcal{R}(I)$:

$$\mathcal{R}(I) \leq \min\ k \text{ s.t. } Np\left(I - \sum_{i=1}^{i=k}\sigma_i l_i \otimes r_i^\top\right) \tag{6}$$

In words, the upper bound is the smallest k such that the difference between I and the kth reconstruction of I must be noise.

It should be noted that the sanctity of these "bounds" depends on the conservative measures of the noise models, while the distance between them (and hence the usefulness in approximating $\mathcal{R}(I)$) depend on the error model being as sharp as possible, i.e. not too conservative. In most cases the upper and lower bounds differ by more than 1 and hence only restrict $\mathcal{R}(I)$, rather than determining it. Because, as we shall see later, it is convenient for our algorithm to have a single number for $\mathcal{R}(I)$, in these cases we make a final "heuristic" determination of it using a knee finding technique on the logarithm of singular values between the upper and lower bound. That is, we chose as $\mathcal{R}(I)$ the i that has maximal curvature on the curve $(i, \log(\sigma_i))$ for i in the range determined by equations 5– 6. This last heuristic step has correctly determined $\mathcal{R}(I)$ in most of our test cases. It is important to note that without the bounds to determine the search window, the knee finding would be much more difficult since in the region of valid singular values there may be other "local" knees, especially in the case where there are multiple motions of significantly different magnitudes.

For our 2D example, the 8 largest singular values were 13820.38, 51.91 16.24 3.51, 1.96, 1.26, 1.14, and .96. Just looking at the above numbers it would be hard to determine if there was one or two motions without knowledge of the expected noise and the bounds on $\mathcal{R}$. The algorithm for determining $\mathcal{R}$ determines the bounds 4 and 34 and the knee finder determines that there are 5 nonzero singular values. After segmentation the algorithm found one cluster had 3 nonzero singular values, the other 2 nonzero singular values.

3.5 Discussion of cluster analysis

First, we point out that the right vector associated with the largest singular value, call it, r_1, need not be the vector which gives the best segmentation. If the two motion parameters are sufficiently mixed together in a particular dimension, but separated in another, the row associated with a smaller singular value may actually give a cleaner segmentation. For a large number of motions it is often the case that a row r_i will allow

one to easily segment out a single motion, while lumping the remaining $N-1$ motions into a single cluster. Therefore, it is useful to consider numerous r_i's.

This is really a general clustering problem, closely related to clustering used in factor analysis. However, because we have the ability to check a partition of the data, we take a slightly more conservative approach. Our goal is not to find all clusters at once, but rather to proceed by breaking the data up into 2 partitions and checking the break. If it is good, then we recursively solve each of the sub-problems. Thus there are three parts to our clustering algorithm: initialization of clusters, refinement of clusters, splitting data and checking clusters.

We perform the clustering as follows. First we use the first right vector and use this to break the track matrix into two new input matrices. The SVD of each of the proposed segments is computed and we check the number of singular values as suggested by Observation 5-2. If this check is satisfied we continue segmentation on the new input matrices. If it fails, we then try clustering on the second right vector. If this fails we try the sum of the first two vectors weighted by their singular values. Obviously we could add other choices, but these have been sufficient for now.

For the 1D clustering problem initialization is straightforward. We know we are looking for 2 clusters, and we use the cluster with maximal separation for initialization. In particular, we compute the nearest neighbor distance for each point and use the largest distance as the breakpoints between the clusters. For the 2D clustering, we initially cluster on the maximal separation of the first right vector.

Given the initial clusters, we refine them attempting to minimize the RMS distance of each point from the cluster center. An algorithm for this is detailed in [Duda and Hart, 1973, Ch. 6]. Applying this iterative algorithm results in 2 clusters and a measure of compactness (root mean square distance to cluster center) for each cluster. In most of our examples, the number of iterations =1, i.e. the initial clustering is already minimal. In the very noisy cases however, the initial clustering is a poor approximation and the iterative improvement offered by this approach can significantly improve the quality of segmentation. The squared error nature can, at times, cause a few outliers to be included in the wrong cluster. The issues of normalization in clustering arise here (e.g. do we use $\sigma_i r_i$, $\sqrt{\sigma_i} r_i$, or simply use r_i), see [Duda and Hart, 1973, pp. 216 ad pp. 224]. Currently we do not normalize.

Figure 3 shows the first two components of R used in clustering on the 2D example. The points are shown with labels so you can see which points correspond to which motions. Note that the first right vector (associated with the first singular value) correctly classifies the motions resulting in the aforementioned segmentation.

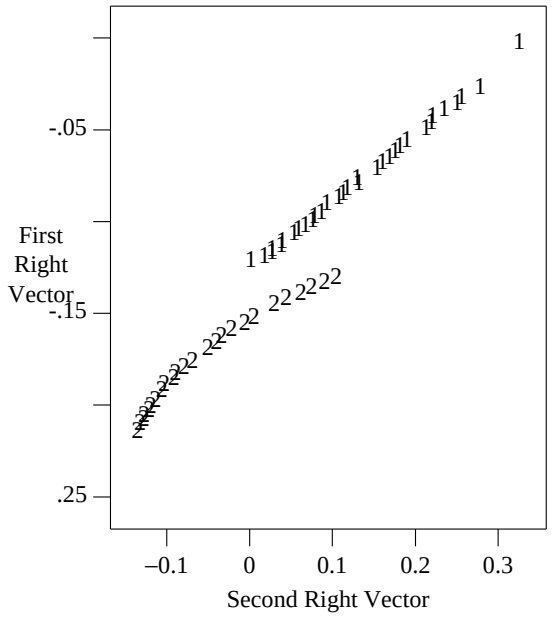

Figure 3: Here we see the clustering for each track shown in Figure 1 using its entry in the first right singular vector as the Y coordinate, and its entry in the second right vector as the X coordinate. Each point is labeled with the motion which generated it.

4 Initial Experimentation

Our initial experiments have been very promising. Figure 4 illustrates our method applied to a dataset with tracks from four different motions. The epi-image was constructed in a similar fashion to our previous 2D example with real data except that additionally, we combined epi-images acquired at different time periods in order to have an example with more motions. The tracks from each motion are contiguous and can be distinguished using the bottom image in this figure. Here the tracks from two different motions can be clearly discerned; the tracks not in this image, make up the other two motions. In Figure 5, ‘ we show the results of clustering using the 2nd right right vector. Our algorithm segmented the points marked with “x” from those with the “o”. The latter correspond to the motion on the right in the bottom image of Figure 4. However, it is clear from this plot, that it is possible to segment all four motion components just using the first two right vectors of the original SVD, without further recursion.

We have also experimented with synthetic data using both additive and multiplicative white noise, $N(0, sd)$, displacing each track point. We have been able to segment two motions with complete accuracy for noise levels up to 50% of the point value. An example of this

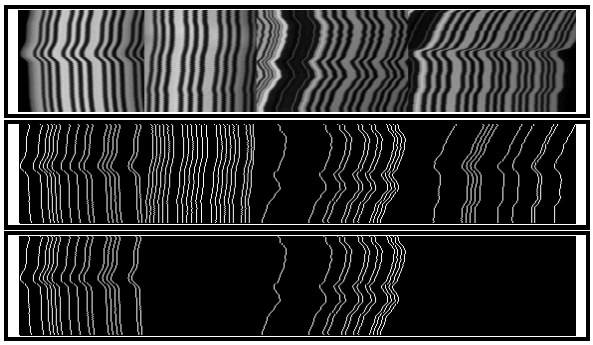

Figure 4: Top shows dataset with 4 motions. The middle shows the tracks found for 59 points over 100 frames. The bottom shows the tracks of two clearly distinguishable motions.

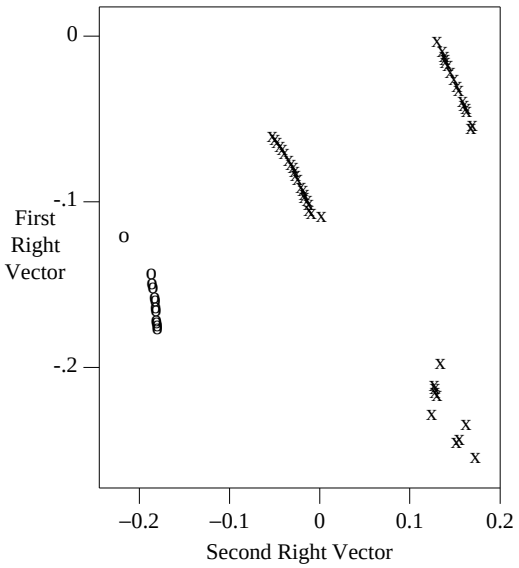

Figure 5: Here we see the clustering of the first 2 right singular vectors for a realistic case with four motions. The first segmentation splits the tracks labeled with an "x" from those with a "o" - the latter correspond to the motion given by the tracks shown on the right in the bottom of Figure 4.

is shown in Figure 6 which used 100 frames of 2 synthetic 2D motions for random shapes with 30 points each. This figure shows the tracks of both motions. The bottom of Figure 6 contains these tracks with multiplicative white noise of $sd = .5$. If it were possible to obtain the input tracks in such a noisy environment, our algorithm would be able to segment them exactly. (We note that the the right singular values clearly cluster so that no tracks are mislabeled although the number of significant singular values becomes very difficult to determine and hence we cannot be sure that the segmentation is good.)

The results of segmenting a difficult 3D synthetic example with three motions can be seen in Figure 7. (Points are labeled with their corresponding motion.) The figure shows a plot of the first two right singular vectors for each of 3 motions (10 random (overlapping) shape points per motion, 50 frames of motion). The first pass of the algorithm split the motion into two groups, one containing motion 1 and the point labeled 3', and the second the remaining points. Initially it found 9 singular values, and after segmentation there were 4 and 6 singular values. The group with 6 singular values (motions 2 and 3) were further decomposed into their correct motion components.

This example also shows why it might be useful to disregard the cluster labels for points on the fringes of the cluster and then add them into the cluster after recomputing the SVD. One way to do this would be to split the tracks conservatively into two groups I_1 and I_2 (more than $\mathcal{R}(I)$ points each), and compute the SVD of each. Then for any track t_i that was on the fringe, compute $w = \hat{L}_k^\top t_i$. This should have $w_i \approx 0, \forall i > \mathcal{R}(I_k)$ if t_i is in the same subspace. While it sounds promising, this technique has yet to be implemented and tested.

5 Critical analysis

While the technique presented in this paper does, in our opinion, a very good job at segmentation of multiple motions, there are still a few difficulties with the approach. Most of the difficulties are actually problems with the underlying factorization technique, and may, we hope, be overcome with additional effort.

We present a check-list of the advantages (+) and disadvantages (−) of this approach. Those aspects which are both pros and cons will be marked with ±.

+ The segmentation approach using the R component of the SVD appears to be extremely powerful.
+ The factorization technique simultaneously provides shape and motion.
+ Shape is represented relative to object centroid and hence is stable with respect to short-baselines in motion.
+ A technique for bounding and approximating the numerical rank $\mathcal{R}(I)$ has been developed
+ The segmentation approach comes with a theoretically derived check on the quality of the segmentation (assuming linearly independent motions and that $\mathcal{R}(I)$ is correct)).
± The segmentation method is quite robust w.r.t. positional noise.
± Cost is $O(mn^2 + n^3)$ (with a reasonable constant). If the number of points and frames are not too large

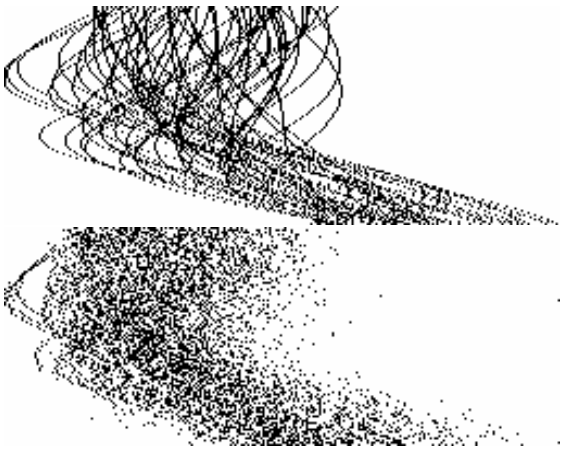

Figure 6: Tracks of 2 motions for a synthetic 2D case. The top figure shows the tracks with no noise and the bottom for $\sigma = .5$.

this is quite reasonable. For example for 200×100 3D data set (hence a 400×100 input matrix) the approach takes $\approx$ 64 CPU seconds on a 12Meg-SparcStation1.

– Factorization approach assumes orthographic projection.

– The segmentation method can have difficulties when there are nearly dependent motions.

6 Conclusions and Future Work

This paper generalized the factorization approach for simultaneous recovery of motion and shape to handle multiple motions. It presented a technique, using the information available from factorization, for the segmentation of multiple motions. The method includes a way to bound the numerical rank of the input and proves how to use this to check, in general, that the segmentation is valid. The segmentation technique was demonstrated on numerous real and synthetic examples.

There is considerable future work to be done in the area of factorization-based motion analysis. For example, consider the many technical reports on which Tomasi and Kanade appear to be working [Tomasi and Kanade, 1991]. Much of that future work applies here as well. There are a few things on which we expect to continue working as they have particular impact on our segmentation approach. These areas include a more thorough error analysis of the segmentation including a better analysis of different approaches to clustering (including removing the fringe points), comparison with previous motion segmentation techniques, and extending the factorization approach to handle partial tracks and tracks distorted by perspective.

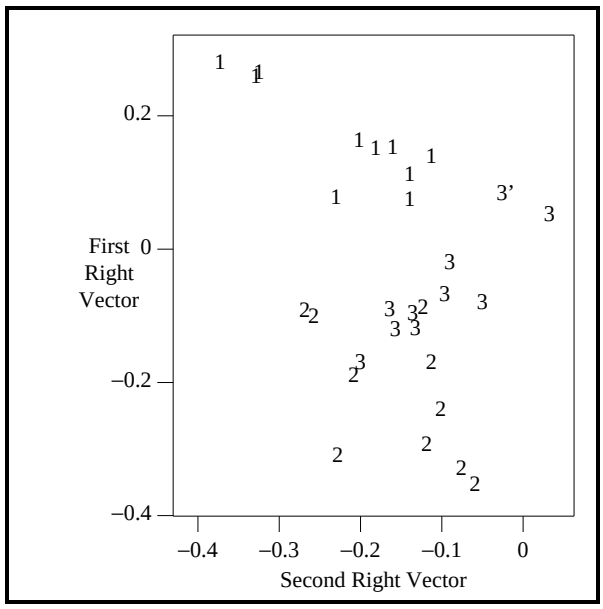

Figure 7: Here we see the clustering of the first 2 right singular values for 3 synthetic motions. Point 3' is mistakenly clustered in with the data from motion 1.

References

[Adiv, 1985] G. Adiv. Determining three-dimensional motion and structure from optical flow generated by several moving objects. *IEEE PAMI*, 7(4):384–401, July 1985.

[Bergen *et al.*, 1990a] J. Bergen, P.J. Burt, R. Hingorani, and S. Peleg. Computing two motions from three frames. In *ICCV*, pages 27–32. IEEE, Dec 1990.

[Bergen *et al.*, 1990b] J. Bergen, P.J. Burt, R. Hingorani, and S. Peleg. Transparent-motion analysis. In *Springer Lecture Notes in Computer Science: Proceedings of ECCV*, volume 427, pages 566–569. Springer-Verlag, April 1990.

[Dickmanns, 1989] E.D. Dickmanns. Subject-object discrimination in 4d dynamic scene interpretations for machine vision. In *Proc. IEEE Workshop on Visual Motion*, pages 289–305. IEEE, March 1989.

[Duda and Hart, 1973] R. O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis*. Wiley, NYC, NY, 1973.

[Fennema and Thompson, 1979] C. Fennema and W.B. Thompson. Velocity determination in scenes containing several moving objects. *CVGIP*, 9:301–316, 1979.

[Golub and van Loan, 1983] G.H. Golub and C.F. van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, MD, 1983.

[Heita and Bouthemy, 1990] F. Heita and P. Bouthemy. Motion based segmentation. In *Proc. Int. Conf on Pat. Rec.*, pages 378–383. IEEE, June 1990.

[Murray and Buxton, 1987] D.W. Murray and B.F. Buxton. Scene segmentation from visual motion using global optimization. *IEEE PAMI*, 9(2):220–228, Mar 1987.

[Peleg and Rom, 1990] S. Peleg and H. Rom. Motion based segmentation. In *Proc. Int. Conf on Pat. Rec.*, pages 109–113. IEEE, June 1990.

[Press *et al.*, 1988] W. Press, B. Flannery, S. Teukolsky, and W. Vetterling. *Numerical Recipes in C.* Cambridge Univ. Press, Cambridge, UK, 1988.

[Shizawa and Mase, 1990] M. Shizawa and K. Mase. Simultaneous multiple optical flow estimation. In *Proc. Int. Conf on Pat. Rec.*, pages 274–278. IEEE, June 1990.

[Subrahmonia *et al.*, 1990] J. Subrahmonia, Y.P. Hung, and D.B. Cooper. Motion based segmentation. In *Proc. Int. Conf on Pat. Rec.*, pages 390–398. IEEE, June 1990.

[Tomasi and Kanade, 1990a] C. Tomasi and T. Kanade. Shape and motion from image streams: a factorization method. Technical report, Carnegie Mellon Computer Science, Pitt. PA, Sep 1990. TR Num. CMU-CS-90-166.

[Tomasi and Kanade, 1990b] C. Tomasi and T. Kanade. Shape and motion without depth. In *Proceedings of the DARPA Image Understanding Workshop*, pages 258–271. DARPA, Morgan Kaufman Pub. San Mateo CA, Sep 1990. ISBN 1-55860-140-6.

[Tomasi and Kanade, 1990c] C. Tomasi and T. Kanade. Shape and motion without depth. In *ICCV*, pages 91–95. IEEE, Dec. 1990.

[Tomasi and Kanade, 1991] C. Tomasi and T. Kanade. Shape and motion from imgage streams: a factorization method. 2. point features in 3d motion. Technical report, Carnegie Mellon Computer Science, Pitt. PA, January 1991. TR Num. CMU-CS-91-105.

[Yamamoto, 1990] M. Yamamoto. A segmentation method based on motion from image sequence and depth. In *Proc. Int. Conf on Pat. Rec.*, pages 230–233. IEEE, June 1990.