Motion Segmentation using the Hadamard Product and Spectral Clustering

Jae-Hak Kim Lourdes Agapito

Department of Computer Science
Queen Mary, University of London
London, E1 4NS, United Kingdom

Abstract

In this paper we present a motion segmentation algorithm for image sequences based on the Hadamard (or Schur) product of shape interaction matrices computed over a range of dimensions of the ambient space and using a spectral clustering algorithm. Most motion segmentation algorithms proposed to date are based on the use of a shape interaction matrix, obtained via factorization, since it encodes the essential information to segment independently moving rigid objects. However, so far, most studies have been limited to using a single shape interaction matrix to cluster the motions of different objects. In this paper, we propose to combine the shape interaction matrices computed for different subspace dimensions using the Hadamard product. The benefit of this approach is that the affinity of trajectories belonging to the same object is stressed while the affinity between trajectories belonging to different objects is diminished. Once the final shape interaction matrix is computed, we use a spectral clustering algorithm to segment the different motions. Experiments on the Hopkins155 data set for both independent and articulated motions show that our new algorithm provides a lower miss-classification error rate, outperforming other state of the art algorithms.

1. Introduction

Motion segmentation for image sequences has been studied for decades in computer vision. The factorization algorithm introduced by Tomasi and Kanade [11] recovers the motion and 3D structure of an object viewed by an orthographic camera by taking the singular value decomposition (SVD) of a measurement matrix of n 2D image positions across f images. Since then, the factorization algorithm has been widely used by researchers as an approach to tackle the motion segmentation problem.

Costeira and Kanade [1] proposed decomposing a measurement matrix and then extracting the motion invariant information which is encapsulated in a matrix, called *"shape interaction matrix"*. Their method works well for independently moving objects, however, most real image sequences tend to have dependent motions so their method fails if the motions are dependent. For dealing with dependent motions, Zelnik-Manor and Irani [18] suggested to use the k dominant eigenvectors of the matrix that separates the dependent motions. They also decompose a measurement matrix but instead of using all eigenvectors, they use only k (the number of motions) eigenvectors for constructing a shape interaction matrix. Although their algorithm can solve the problem for dependent motions, it does not perform well for articulated motions. Yan and Pollefeys [15] proposed an algorithm which can manage articulated motions by local subspace sampling. Instead of taking the dot product of two vectors to build the shape interaction matrix, they used the principal angles computed by sampling the two different subspaces. Their algorithm works particularly well for the case of articulated motions.

Vidal and Hartley [13] introduced a GPCA algorithm

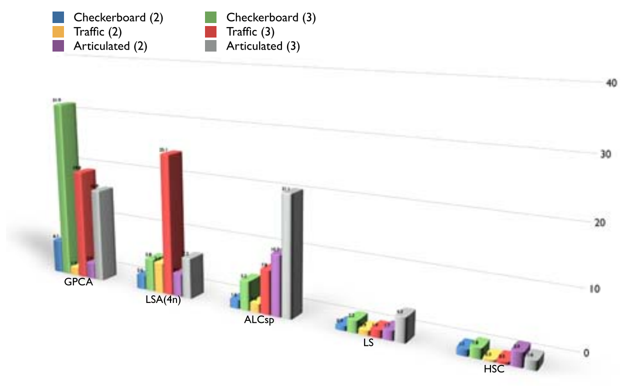

Figure 1. ***Comparision of miss-classification rate.*** *Our HSC method outperforms other methods – GPCA, LSA4n ALCsp and LS [13, 15, 6, 4].*

with PowerFactorzation to deal with missing data in motion segmentation. Schindler *et al.* [7, 8] suggested an algorithm to solve the multi-body structure and motion problem by using Monte-Carlo sampling in two views and n views. Rao *et al.* [6] worked on an algorithm for motion segmentation based on a subspace separation approach by using a lossy compression technique from information theory. More recently, Silva and Costeira [2] suggested a subspace separation algorithm when the data has outliers.

Lauer and Schnönrr [4] have recently proposed an algorithm to search for the best dimension of ambient space by analysing the gap between eigenvalues which eventually finds the correct number of clusters for the spectral clustering algorithm. However, their algorithm assumes that the number of motions is known and uses it for searching the best dimension. Therefore, without knowing the number of motions, it is not applicable to practical applications.

The motion segmentation algorithm proposed in this paper is most closely related to Lauer and Schnönrr's work [4]. We also find an upper bound for the dimension of the ambient space but instead of searching for the best dimension within the bounds, we mix a range of possible dimensions of the ambient space ($r = 2, \dots, n$) by combining the shape interaction matrices computed for different values of r using the Hadamard product. Effectively the affinity matrix built using the Hadamard product increases the separability of the different motion subspaces.

We provide convincing results on the Hopkins155 dataset that show the superiority of our motion clustering algorithm with respect to the other state of the art algorithms discussed above.

2. Factorization for Motion Segmentation

In this section we revisit Tomasi and Kanade's factorization algorithm [11] to compute the 3D structure and motion of a scene observed by an orthographic camera given 2D image tracks. We also define the concept of *shape interaction matrix* introduced by Costeira and Kanade in [1] to solve the problem of motion segmentation for multiple independently moving objects. We investigate the properties of the shape interaction matrix in detail in the cases when the objects move independently and when there are dependencies in the motions.

2.1. Factorization Algorithm

The case of a single object. Assuming an affine camera viewing a single 3D object with n points, the projection of those world points onto f frames can be described as

$$\mathtt{W} = \begin{bmatrix} \mathbf{x}_{11} & \mathbf{x}_{12} & \dots & \mathbf{x}_{1n} \\ \mathbf{x}_{21} & \mathbf{x}_{22} & \dots & \mathbf{x}_{2n} \\ \vdots & \vdots & & \vdots \\ \mathbf{x}_{f1} & \mathbf{x}_{f2} & \dots & \mathbf{x}_{fn} \end{bmatrix} = \begin{bmatrix} \mathtt{M}_1 \\ \mathtt{M}_2 \\ \vdots \\ \mathtt{M}_f \end{bmatrix} \begin{bmatrix} \mathbf{s}_1 & \mathbf{s}_2 & \dots & \mathbf{s}_n \end{bmatrix},$$

where $\mathtt{x}_{ij} = (u_{ij}\ v_{ij})^\top$ encodes the 2D image coordinates of point j in the i-th image, $\mathtt{M}_i = [\mathtt{R}_i \mid \mathbf{t}_i]$ is the 2×4 affine projection matrix for frame i which encodes the 2×3 rotation matrix $\mathtt{R}_i$ and the 2×1 translation vector $\mathbf{t}_i$, and $\mathbf{s}_j = (X_j\ Y_j\ Z_j\ 1)^\top$ are the 4-vectors that encode the 3D coordinates of the n points.

Therefore, the $2f \times n$ measurement matrix $\mathtt{W}$ can be factorized into the product of two low-rank matrices as $\mathtt{W} = \mathtt{M}_{2f\times r}\ \mathtt{S}_{r\times n}$, where $\mathtt{M}$ and $\mathtt{S}$ correspond to the motion and shape subspaces respectively. As a result, the rank of $\mathtt{W}$ is constrained to be $\text{rank}\{\mathtt{W}\} \leq r$ where $r = 4$ in the case of an affine camera viewing a single object. This rank constraint forms the basis of Tomasi and Kanade's factorization method for the estimation of 3D structure and motion.

The classic approach in factorization is to exploit the rank constraint to factorize the measurement matrix into a motion matrix $\mathtt{M}$ and a shape matrix $\mathtt{S}$ by truncating the SVD of $\mathtt{W}$ to the rank r specific to the problem as $\mathtt{W} = \mathtt{MS} = \mathtt{UD}^{\frac{1}{2}}\mathtt{D}^{\frac{1}{2}}\mathtt{V}^\top$, where $\mathtt{M} = \mathtt{UD}^{\frac{1}{2}}$ is the motion matrix, $\mathtt{S} = \mathtt{D}^{\frac{1}{2}}\mathtt{V}^\top$ is the shape matrix. However, this factorization is not unique since any invertible $r \times r$ matrix $\mathtt{T}$ can be inserted, leading to the alternative factorization: $\mathtt{W} = (\hat{\mathtt{M}}\mathtt{T})(\mathtt{T}^{-1}\hat{\mathtt{S}})$. Therefore, we now have a factorization of the measurement matrix as follows:

$$\mathtt{W} = \hat{\mathtt{M}}\hat{\mathtt{S}} = \left(\mathtt{UD}^{\frac{1}{2}}\mathtt{T}\right)\left(\mathtt{T}^{-1}\mathtt{D}^{\frac{1}{2}}\mathtt{V}^\top\right).$$

In the structure from motion scenario, the problem is then to find the transformation matrix $\mathtt{T}$ that removes the affine ambiguity, upgrading the reconstruction to metric and constraining the motion matrices to have the appropriate structure.

Multiple objects with independent motions. Consider now k independently moving objects in the scene and assume that the image coordinates have been ordered according to the different objects. In this case, the measurement matrix may be described as $\mathtt{W} = [\ \mathtt{W}_1 \mid \mathtt{W}_2 \mid \dots \mid \mathtt{W}_k\ ]$, where $\mathtt{W}_k$ contains the image coordinates of the k-th object. Assuming there are n_i points on each object, the total number of points is $n = \sum_{i=1}^{k} n_i$. In the case of independently moving objects we may decompose the measurement matrix $\mathtt{W} = \mathtt{MS}$ as follows:

$$\mathtt{M} = \begin{bmatrix} \mathtt{M}_{11} & \mathtt{M}_{21} & \dots & \mathtt{M}_{k1} \\ \mathtt{M}_{12} & \mathtt{M}_{22} & \dots & \mathtt{M}_{k2} \\ \vdots & \vdots & \ddots & \vdots \\ \mathtt{M}_{1f} & \mathtt{M}_{2f} & \dots & \mathtt{M}_{kf} \end{bmatrix}$$

$$\mathtt{S} = \begin{bmatrix} \mathtt{S}_1 & & & \\ & \mathtt{S}_2 & & \\ & & \ddots & \\ & & & \mathtt{S}_k \end{bmatrix},$$

where $\mathtt{M}_{ij}$ is the 2×4 matrix that projects the coordinates of the j-th object onto the i-th frame, and $\mathtt{S}_j$ is the $4 \times n_j$ shape matrix for j-th object. Therefore, the $2f \times n$ measurement matrix $\mathtt{W}$ may be decomposed into the product of two matrices as $\mathtt{W} = \mathtt{M}_{2f\times 4k}\mathtt{S}_{4k\times m}$. For this reason, in the multi-body factorization case the rank of $\mathtt{W}$ is at most $4k$, so it can be decomposed via SVD as follows:

$$\mathtt{W} = [\, \mathtt{W}_1 \mid \mathtt{W}_2 \mid \dots \mid \mathtt{W}_k \,] = \mathtt{MS}$$

$$\mathtt{M} = [\, \mathtt{U}_1 \mid \mathtt{U}_2 \mid \dots \mid \mathtt{U}_k \,] \begin{bmatrix} \mathtt{D}_1^{\frac{1}{2}} & & & \\ & \mathtt{D}_2^{\frac{1}{2}} & & \\ & & \ddots & \\ & & & \mathtt{D}_k^{\frac{1}{2}} \end{bmatrix}$$

$$\mathtt{S} = \begin{bmatrix} \mathtt{D}_1^{\frac{1}{2}} & & & \\ & \mathtt{D}_2^{\frac{1}{2}} & & \\ & & \ddots & \\ & & & \mathtt{D}_k^{\frac{1}{2}} \end{bmatrix} \begin{bmatrix} \mathtt{V}_1^\top & & & \\ & \mathtt{V}_2^\top & & \\ & & \ddots & \\ & & & \mathtt{V}_k^\top \end{bmatrix}$$

where $\mathtt{W}_k = \mathtt{U}_k\mathtt{D}_k\mathtt{V}_k^\top$.

Once more, this decomposition is not unique since a non-singular matrix block diagonal matrix $\mathtt{T}$ of the form $\mathtt{T} = \text{blkdiag}(\mathtt{T}_1, \mathtt{T}_2, \dots, \mathtt{T}_k)$ would give the alternative factorization:

$$\hat{\mathtt{M}} = \mathtt{MT}$$
$$\hat{\mathtt{S}} = \mathtt{T}^{-1}\mathtt{S}\,.$$

The remaining problem would now be to find the transformation $\mathtt{T}$ that imposes the metric constraints to upgrade the reconstruction to metric and imposes the block structure to the motion and shape matrices expressed above. However, the difficulty with this is that so far we have assumed that the image tracks have been ordered according to the different objects so the metric constraints are only applicable once the segmentation is known. To achieve this necessary motion segmentation, Costeira and Kanade [1] defined the *shape interaction matrix*, a mathematical structure that preserves the original subspace structure. In this framework, motion segmentation amounts to the problem of linear subspace separation.

3. Shape Interaction Matrix

The main difficulty in using a factorization approach to segment the image trajectories arising from the different objects is that the shape matrix $\hat{\mathtt{S}}$ is not uniquely determined by SVD. Moreover, although the matrix $\mathtt{V}^\top$ is uniquely computed, it looses its block diagonal structure since in general it is a linear combination of the different motion subspaces. Therefore, the matrix $\mathtt{V}$ cannot be used alone to determine separate motions for different objects.

The case of independent motions. In the case of independent and non-degenerate motions, a motion segmentation algorithm was proposed by Costeira and Kanade in [1] using the *shape interaction matrix* which they defined as:

$$\mathtt{Q} = \mathtt{VV}^\top\,,$$

where $\mathtt{V}^\top = \text{blkdiag}(\mathtt{V}_1^\top, \mathtt{V}_2^\top, \dots, \mathtt{V}_k^\top)$. Costeira and Kanade proved that the in the case when the entries of the measurement matrix are ordered according to each object, the shape interaction matrix is block diagonal with each block corresponding to each independently moving object. More importantly, in the case when the segmentation of the tracks is unknown the values of the entries of the shape interaction matrix do not vary, only their ordering is affected. For k independent motions, the shape entries of the shape interaction matrix $\mathtt{Q}$ satisfy the following property:

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

Another important property of $\mathtt{Q}$ is that it is invariant to the motion. Therefore the shape interaction matrix encodes all the information to carry out motion segmentation of independently moving objects and Costeira and Kanade's segmentation algorithm reduces to finding the permutations that will impose a block diagonal structure to the shape interaction matrix.

The case of dependent motions. In the case of multiple objects with fully or partially dependent motions, the shape interaction matrix $\mathtt{Q}$ is not a block diagonal matrix as shown by Zelnik-Manor and Irani [18] who proved that $\mathtt{Q}$ has block diagonal structure if and only if the matrix $\mathtt{V}$ is block diagonal. By definition, the columns of $\mathtt{V}$ are the eigenvectors of the matrix $\mathtt{W}^\top\mathtt{W}$ which has the form:

$$\mathtt{W}^\top\mathtt{W} = \begin{bmatrix} \mathtt{W}_1 & \mathtt{W}_2 & \dots & \mathtt{W}_k \end{bmatrix}^\top \begin{bmatrix} \mathtt{W}_1 & \mathtt{W}_2 & \dots & \mathtt{W}_k \end{bmatrix}$$
$$= \begin{bmatrix} \mathtt{S}_1^\top\mathtt{M}_1^\top\mathtt{M}_1\mathtt{S}_1 & \mathtt{S}_1^\top\mathtt{M}_1^\top\mathtt{M}_2\mathtt{S}_2 & \dots & \mathtt{S}_1^\top\mathtt{M}_1^\top\mathtt{M}_k\mathtt{S}_k \\ \mathtt{S}_2^\top\mathtt{M}_2^\top\mathtt{M}_1\mathtt{S}_1 & \mathtt{S}_2^\top\mathtt{M}_2^\top\mathtt{M}_2\mathtt{S}_2 & \dots & \mathtt{S}_2^\top\mathtt{M}_2^\top\mathtt{M}_k\mathtt{S}_k \\ \vdots & \vdots & \ddots & \vdots \\ \mathtt{S}_k^\top\mathtt{M}_k^\top\mathtt{M}_1\mathtt{S}_1 & \mathtt{S}_k^\top\mathtt{M}_k^\top\mathtt{M}_2\mathtt{S}_2 & \dots & \mathtt{S}_k^\top\mathtt{M}_k^\top\mathtt{M}_k\mathtt{S}_k \end{bmatrix}$$

The off-diagonal blocks of $\mathtt{W}^\top\mathtt{W}$ are only zero if the motions $\mathtt{M}_i, \ldots, \mathtt{M}_j$ are independent. Therefore, $\mathtt{V}$ and $\mathtt{Q}$ will only have a block diagonal structure if and only if the motions are independent.

Consequently, motion segmentation algorithms that depend on the diagonal structure of the shape interaction matrix, such as Costeira and Kanade's, will fail in the presence of dependencies between the motions of the different objects. This can occur frequently in real sequences. The specific case of articulated motion was treated by Yan and Pollefeys [15].

3.1. Dimension of the Shape Interaction Matrix

In the case of dependent motions, Zelnik-Manor and Irani [18] noticed that when the dependence between the motions is not full there will be at least some vectors in $\mathtt{V}$ that will capture the independence between the motions. Therefore, they proposed to build the shape interaction matrix using only the most dominant vectors in $\mathtt{V}$. In particular, they take r vectors where r was equal to the number of objects. However, the number of vectors from $\mathtt{V}$ chosen to build the shape interaction matrix $\mathtt{Q}$ can vary from $r = 2, \ldots, n$ (where n is the number of points in total).

The values of the entries of the shape interaction matrix will depend on the dimension of the subspace constructed using the rows of the matrix $\mathtt{V}$. Let $\mathtt{V}(r)$ be a sub-matrix of $\mathtt{V}$ constructed by taking the first r columns of $\mathtt{V}$. We may define the shape interaction matrices $\mathtt{Q}(r)$ for different dimensions r as follows:

$$\mathtt{Q}(r) = \mathtt{V}(r)\mathtt{V}(r)^\top \ , \tag{1}$$

for the range of dimensions $r = 2, \ldots, n$.

The shape interaction matrix $\mathtt{Q}(r)$ of variable dimension is a square matrix of size n and the values of its entries are in the range of 0 to 1 because it is a matrix constructed taking the dot product of column vectors of $\mathtt{V}(r)$ which is a unitary matrix.

3.2. Shape Interaction Matrix for Segmentation

Costeira and Kanade used the shape interaction matrix to determine independent motions for multiple objects in [1]. However, the method does not work well for articulated objects. Motion segmentation for articulated objects is studied by Yan and Pollefeys in [15, 16, 17]. They take the matrix $\mathtt{V}$ from the SVD of $\mathtt{W}$ and normalize $\mathtt{V}$, then an affinity matrix is constructed by principal angles of locally sampled subspaces to cluster articulated motions.

A spectral clustering algorithm for motion segmentation is proposed by Lauer and Schnörr in [4]. Their method searches for the dimension of ambient space, which is best suited for spectral clustering.

4. Motion Segmentation by Spectral Clustering

4.1. Spectral Clustering

In recent years, the spectral clustering algorithm has become attractive to the computer vision and machine learning communities because of its powerful performance and easy implementation [14, 5, 10, 9]. Spectral clustering is analogous to a min-cut problem, which splits a graph into two subgraphs to minimize the amount of capacity (or weight) of an edge between two partitioned subgraphs.

The spectral clustering algorithm uses an affinity matrix of the data to perform the partitioning. The affinity matrix, which encodes the similarity between different vectors, is generally defined as $\mathtt{A}_{ij} = \exp(-|\mathbf{x}_i - \mathbf{x}_j|/2\sigma)$, where $\mathbf{x}_i$ are the data vectors. In the case of motion segmentation the data vectors are the rows $\mathbf{v}_i$ of the matrix $\mathtt{V}$ given by the SVD of the measurement matrix.

The partitioning is then achieved by taking the largest k eigenvectors of the graph laplacian of the affinity matrix. Because the definition of the graph laplacian is not consistent between authors, here we use the following definition: $\mathtt{L} = \mathtt{D}^{-\frac{1}{2}}\mathtt{A}\mathtt{D}^{-\frac{1}{2}}$, where $\mathtt{D}$ is a diagonal matrix with $\mathtt{D}_{ii} = \sum_{j=1}^{n} \mathtt{A}_{ij}$. The k largest eigenvectors of $\mathtt{L}$ are computed and used for clustering using any clustering algorithm, for instance, a k-means algorithm.

4.2. Affinity Matrix for Motion Segmentation

Spectral clustering has been used in the context of motion subspace segmentation by different authors [15, 4]. One of the main differences between these algorithms is the way in which the affinity matrix is specifically defined. Yan and Pollefeys [15] define the affinity matrix as $\mathtt{A} = e^{-\sum \sin^2(\theta)}$ where θ is principal angles of two subspaces. Lauer and Schnörr [4] define the affinity matrix as $\mathtt{A} = \cos^{2\alpha}(\theta)$ where θ is the angle between two vectors in subspaces and α is a scalar.

4.3. Dimension of Ambient Space

Another important issue is the choice of the dimension of the ambient space. For k independent motions, Vidal and Hartley suggested to use $r = k + 1$ for the dimension of ambient space in [13]. Lauer and Schnörr proposed to search for the optimal dimension of ambient space within the bounds $r = [k+1, \ldots, 4k+1]$ in [4] by choosing the dimension that maximizes the separation between subspaces.

Motivated by Lauer and Schnörr's work [4], in this paper, we find an upper bound for the dimension of the ambient space but instead of searching for the best dimension within the bounds, we mix a range of possible dimensions of the ambient space ($r = 2, \ldots, n$) by combining the shape interaction matrices computed for different values of r using the Hadamard product.

5. Our New Motion Segmentation Algorithm

To create the mixture of shape interaction matrices, computed for different values of r we define a new affinity matrix (or similarity matrix) $\mathtt{A}(r)$ computed from the shape interaction matrix as follows:

$$\mathtt{A}(r) = \exp(\mathtt{Q}(r)) \ , \tag{2}$$

where r is the chosen subspace dimension, $\mathtt{Q}(r)$ is the shape interaction matrix for that dimension, as defined in (1) and $\exp(\mathtt{P})$ computes the exponential of a square matrix $\mathtt{P}$.

Hadamard product of affinity matrices. Given two matrices $\mathtt{X}$ and $\mathtt{Y}$ in $\mathbb{R}^{m\times n}$, the Hadamard product (or Schur product) is the element-wise multiplication of the two matrices $\mathtt{X}$ and $\mathtt{Y}$ defined as: $(\mathtt{X}\cdot\mathtt{Y})_{ij} = \mathtt{X}_{ij}\mathtt{Y}_{ij}$. We have chosen to mix affinity matrices computed for different dimensions r of the vector space using the Hadamard product. The new affinity matrix is defined as follows:

$$\mathtt{H}_{ij} = \prod_{r=2}^{D} \mathtt{A}(r)_{ij} \ , \tag{3}$$

where D is the upper bound of the range of subspace dimensions considered ($D \le n$ where n is the total number of points). Note that each affinity matrix is normalized before the Hadamard product is applied (for instance, by the largest singular value of the matrix). The computation of the upper bound for the subspace dimension D will be considered in one of the following sections. Without loss of generality, consider the values of $\mathtt{H}_{ij}$ in (3) without the normalization scale. Combining (1) and (2), equation (3) may be rewritten as follows:

$$\begin{aligned}\mathtt{H}_{ij} &= \exp\left(\sum_{r=2}^{D} \mathtt{V}(r)_i \mathtt{V}(r)_j^{\top}\right)\\ &= \exp\left(\sum_{r=2}^{D} \cos\theta(r)_{ij}\right) \quad \le \quad e^{D-1} \ ,\end{aligned}$$

where $\mathtt{V}(r)_i$ is the i-th column vector of the matrix $\mathtt{V}(r)$ and $\theta(r)_{ij}$ is the angle between two vectors i and j in the subspace of dimension r. In the case of independent motions, if the vectors of $\mathtt{V}(r)$ for points i and j belong to different subspaces subspace at any dimension r, the entry of $\mathtt{H}_{ij}$ will be close to zero because the angle between the two vectors for that dimension is close to zero. However, if the vectors at i and j are from the same subspace over the range of dimensions $r = 2, \dots, D$, then $\mathtt{H}_{ij}$ will be close to e^{D-1}. Accordingly, the off-diagonal blocks of the matrix $\mathtt{H}$ will become close to zero and the main-diagonal blocks will become close to e^{D-1}.

An example. In this section we use a simple example with data generated synthetically to describe the benefits of using the Hadamard product of affinity matrices to segment different motions using spectral clustering. The synthetic data was generated assuming three different objects moving independently viewed by an affine camera which is static in the world. The 3D points on the three objects were moved independently by introducing different rotations and translations, then projected onto images by the affine camera. To verify the independence of the motions, the block diagonal structure of $\mathtt{W}^{\top}\mathtt{W}$ and $\mathtt{V}\mathtt{V}^{\top}$ is examined. Figure 2 shows the affinity matrices and the Hadamard product of affinity matrices calculated for subspace dimensions $r = 2, \dots, 9$ in the case when the objects are moving independently. Note that the affinity matrices $\mathtt{A}(r)$ have zero (gray) values on the off-diagonal blocks and non-zero (white) values on the main-diagonal blocks when $r = 8$. However, the Hadamard product of affinity matrices $\mathtt{H}$ has zero (black) values in the off-diagonal blocks and non-zero (white) values in the main-diagonal blocks when $r >= 5$. Note that the off-diagonal blocks of $\mathtt{H}$ are closer to zero (darker) than the off-diagonal blocks of $\mathtt{A}(r)$. Therefore, there is a better chance to separate independent motions using the matrix $\mathtt{H}$.

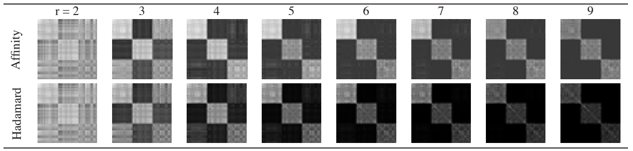

Figure 2. $\mathtt{A}(r)$ ***and*** $\mathtt{H}$ ***for independent motions at*** $r = 2, \dots, 9$***.*** *(Top row) Affinity matrices for* $r = 2, \dots, 9$*. (Bottom row) The Hadamard product of affinity matrices for* $r = 2, \dots, 9$*.*

For dependent motions, the three objects move dependently because they share the same rotations. In Figure 3, the form of the affinity matrices and the Hadamard product of affinity matrices for dimensions $r = 2, \dots, 9$ are shown in the case of dependent motions. The off-diagonal blocks of the matrix $\mathtt{A}(r)$ are not zero which reflects similarity within points in different objects. This is not a good property for motion segmentation. However, the Hadamard product of affinity matrices $\mathtt{H}$ has zeros in the off-diagonal blocks. Hence, the matrix $\mathtt{H}$ has a better affinity information than $\mathtt{A}(r)$ for motion segmentation.

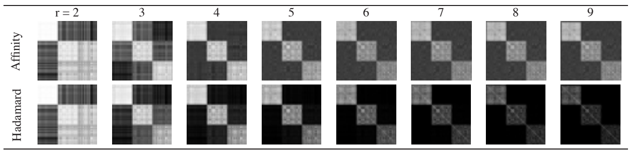

Figure 3. $\mathtt{A}(r)$ ***and*** $\mathtt{H}$ ***for dependent motions at*** $r = 2, \dots, 9$***.*** *(Top row) Affinity matrices for* $r = 2, \dots, 9$*. (Bottom row) The Hadamard product of affinity matrices for* $r = 2, \dots, 9$*.*

Advantages of the Hadamard product of $\mathtt{A}(r)$. The main advantage of computing the Hadamard product of affinity matrices calculated for different dimensions is to integrate information about the object for different subspace dimensions. Given a dimension r, we have a shape interaction matrix $\mathtt{Q}(r)$ where the range of dimensions r can be between 2 and n. If the dimension r is 2, then the shape interaction matrix $\mathtt{Q}(2)$ is a projection onto a two-dimensional subspace. Therefore, it has information to cluster data projected onto the two-dimensional space. For $r = 2$, for instance, the affinity matrix $\mathtt{A}(2)$ has a large number of high values (near to 1) in total while, for $r = n$, the affinity matrix $\mathtt{A}(n)$ has a large number of low values (near to 0) in total. Therefore, it seems reasonable to mix the different types of information rather than selecting a single dimensionality for clustering.

Selecting the value for D. It is crucial to select a reasonable upper bound for the highest subspace dimension D used in the Hadamard product. A good mixture of affinity matrices which provides good separation between spaces should have high values (close to 1, meaning high affinity) for points belonging to the same motions and low values for points belonging to different motions. To choose the subspace dimension D at which to stop adding new matrices to the Hadamard product we examine the histogram of $\mathtt{H}$ and find two centroids from bins of the histogram. If the two centroids are far apart, then the matrix $\mathtt{H}$ has distinct high and low values, meaning there is a good separation between the subspaces. In addition, we compute the number of hits close to each of the two centroids. If the difference between these two values is high then the matrix $\mathtt{H}$ has a good shape for clustering. We choose the value for D according to these two measures.

Figure 4 (top row) compares the values of the *distance between the centroids* and the *difference of hit counts* for the histograms computed using the affinity matrix $\mathtt{A}(r)$ in one case and the Hadamard product $\mathtt{H}(r)$ in the other, for different values of the dimension r. We analyse the histograms obtained for the examples of independent (left) and dependent (right) motion illustrated in Figure 2 and Figure 3 respectively. It is clear that the distance between the histogram centroids increases as new dimensions are added to the Hadamard product (blue circles) until a maximum is reached for $r = 7$. The bottom row of Figure 4 shows that this coincides with the point where the lowest misclassification rate is reached, which explains that the distance between the centroids of the histogram is related to the separability of the subspaces. However, when the affinity matrix is used, the distance between the centroids (green squares) is smaller and does not increase with the dimensionality meaning that the separability of the subspaces is smaller than in the Hadamard product case. The difference of the hit counts for the histograms calculated using the Hadamard product $\mathtt{H}$ (red crosses) also show higher values than those computed with the Affinity matrices (purple triangles). Therefore these graphs illustrate how the choice of the best value for D is related to choosing good threshold values for the distance between the centroids t_1 and the difference of hit counts t_2 of the histogram. For instance, in the above experiments we found that the values $t_1 = 0.5$ and $t_2 = 0.8$ provided good results.

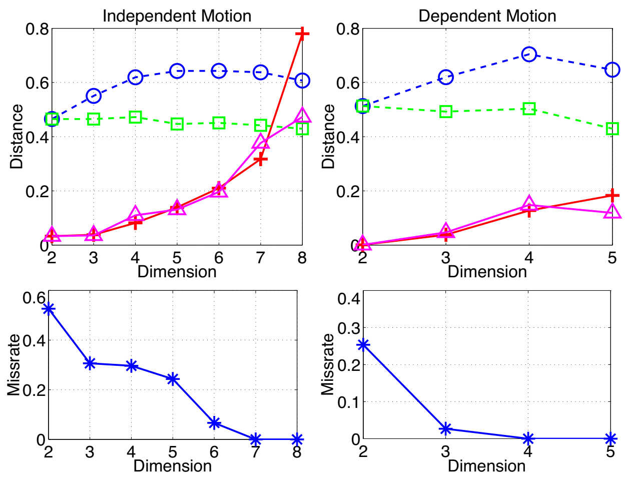

Figure 4. ***Selecting the value of D.*** *(Top row) Information on the histograms calculated with the Affinity matrices and the Hadamard product for different dimension values for the independent motion (left) and the dependent motion (right) examples shown in Figure 2. We show the distance between the two centroids of the histograms calculated using the Hadamard product (blue circles) and the Affinity matrices (green squares). The difference of hit counts is shown for the Hadamard product (red crosses) and the Affinity matrices (purple triangles) histograms. (Bottom row) Misclassification rate using the Hadamard product for the (left) independent motion and (right) the dependent motion examples.*

This simple algorithm can be implemented by using a k-means clustering algorithm with two clusters. The k-means algorithm will give two centroids between 0 to 1 if the matrix $\mathtt{A}$ is scaled down to maximum 1. Therefore, we find two centroids using the k-means algorithm, examine the number of hits around the centroids, using the thresholds defined above. It is advisable to use a fast k-means algorithm by Elkan [3] to reduce the computation time.

5.1. Final Segmentation Algorithm

Our motion segmentation algorithm using the Hadamard and spectral clustering (HSC method) can be summarized as follows:

HSC Algorithm Given a matrix $\mathtt{W} \in \mathbb{R}^{2f\times n}$ for n points in f frames

1. ***Projection:*** *Take the SVD of* $\mathtt{W}$ *and obtain* $\mathtt{V} \in \mathbb{R}^{n\times n}$.
2. ***Construct affinity matrices*** $\mathtt{A}(r)$ ***for different dimensions and multiply using the Hadamard product until the upper bound dimension*** D ***is reached:*** *For the range* $r = [2, \dots, n]$*, consider the* r*-dimension subspace of* $\mathtt{V}$ *by taking* r *columns of* $\mathtt{V}$*. Build a shape interaction matrix* $\mathtt{Q} = \mathtt{V}\mathtt{V}^\top$*. Obtain an affinity matrix* $\mathtt{A}$ *and compute the Hadamard product as* $\mathtt{H}_{ij} = \mathtt{H}_{ij}\mathtt{A}_{ij}/|\mathtt{A}|$ *where* $|\mathtt{A}|$ *is the norm of the matrix* $\mathtt{A}$*.*
3. ***Determine the dimension*** D***:*** *Find two centroids* $\mathbf{c}_1$ *and* $\mathbf{c}_2$ *from* $\mathtt{H}$ *using a k-means algorithm. Repeat step (2) until the distance of the two centroids is* $|\mathbf{c}_1 - \mathbf{c}_2| \leq t_1$ *and the difference between the hit counts is* $|h_1 - h_2| \leq t_2$*, where* t_1 *and* t_2 *are threshold values, and* $h_i = histc(\mathtt{A}, \mathbf{c}_i)$ *is the number of the histogram counts in* $\mathtt{A}$ *near the centroid* $\mathbf{c}_i$*. Here, the dimension* D *is the value* r *at the current iteration.*
4. ***Clustering via spectral clustering algorithm:*** *Do the spectral clustering using the final affinity matrix* $\mathtt{H}$*.*

6. Experiments

In all our experiments we assume that all the feature points are visible over all frames (no missing data).

Synthetic image sequence. Three objects in 3D are randomly placed in front of cameras and the points on each object are projected onto an image by an affine camera as shown in Figure 5. Over $f = 50$ frames, two of the objects are moving and the third one is static so we consider it to be background. There are two independent motions on the two objects but the affine camera is also moving. Therefore, there are three independent motions in total. Gaussian noise with variance 2 pixels is added to image coordinates. Figure 5-(bottom-right), shows that our algorithm correctly segments the motions of the three objects.

Real image sequences. We tested our algorithm on the Hopkins155 dataset [12] which contains real image sequences. The sequences in the Hopkins155 dataset contain images of three different types: checkerboard, traffic and articulated image sequences. In Figure 6, three types of sample images and motion segmentation on the images are shown and the correct segmentation is superimposed on the images using a distinct colour.

For comparison of our algorithm (HSC) with other state of the art algorithms such as ALCsp [6], GPCA [13], LSA4n [15], and LS [4], we show the result in table 1 and table 2. Our HSC method significantly outperforms ALCsp, GPCA, LSA4n and performs slightly better than the most recent LS.

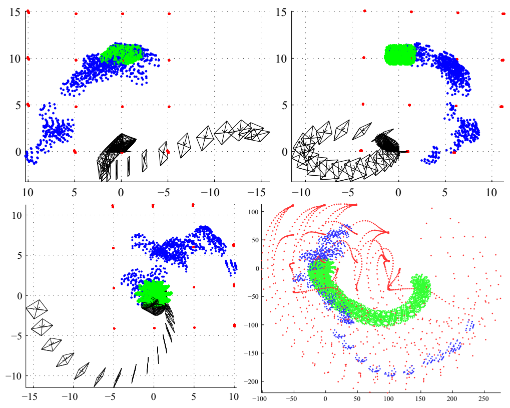
Figure 5. ***Synthetic data.*** *(Top row and the bottom-left) 3D Positions of three objects (red, green and blue) and cameras (black) over 50 frames. Two objects of blue and green are independent motions and the red object is static as a background. (Front, side and top view in the order) (Bottom right) Projected image trajectories for the three objects (red, green and blue) and its motion classification result by our technique.*

Checker(2)	ALCsp	GPCA	LSA4n	LS	HSC
Average	1.55	6.09	2.57	**0.85**	1.12
Median	0.29	1.03	0.27	0.00	0.00
Traffic(2)	ALCsp	GPCA	LSA4n	LS	HSC
Average	1.59	1.41	5.43	0.90	**0.33**
Median	1.17	0.00	1.48	0.00	0.00
Artic(2)	ALCsp	GPCA	LSA4n	LS	HSC
Average	10.70	2.88	4.10	**1.71**	2.44
Median	0.95	0.00	1.22	0.00	0.00

Table 1. ***Miss-classification rates for two motions.*** *Average miss-classification error rate for Hopkins155 sequences in two motions. Reported results from [6, 4].*

Checker(3)	ALCsp	GPCA	LSA4n	LS	HSC
Average	5.20	31.95	5.80	2.15	**1.77**
Median	0.67	32.93	1.77	0.47	0.55
Traffic(3)	ALCsp	GPCA	LSA4n	LS	HSC
Average	7.75	19.83	25.07	1.35	**0.49**
Median	0.49	19.55	23.09	0.19	0.00
Artic(3)	ALCsp	GPCA	LSA4n	LS	HSC
Average	21.08	16.85	7.25	4.26	**1.60**
Median	21.08	16.85	7.25	4.26	1.60

Table 2. ***Miss-classification rates for three motions.*** *Average miss-classification error rate for Hopkins155 sequences in three motions. Reported results from [6, 13].*

We also test our algorithm on other real image sequences for instance Schindler's data [7]. Figure 6 shows that our algorithm achieves the correct motion segmentation. In these experiments, two threshold values $t_1 = 0.29$ and $t_2 = 0.87$ are used over all real image sequences.

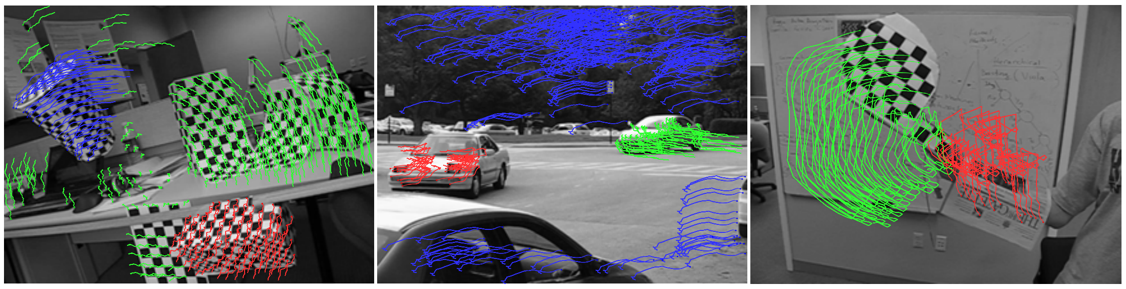
Figure 6. ***Hopkins155 sample and motion segmentation.*** *(Checkerboard) "1R2RCR" with three motions, (Traffic) "cars5" with three motions and (Articulated) "arm" with two motions.*

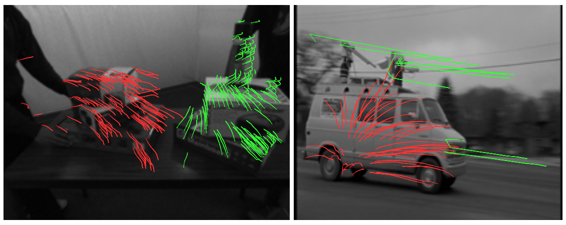
Figure 7. ***Image sequences of "boxes" and "deliveryvan".*** *Motion segmentation result using our algorithm on Schindler's data [7].*

7. Conclusions

We have proposed a motion segmentation algorithm that uses the Hadamard product of affinity matrices computed for a range of different subspace dimensions. Using the Hadamard product of affinity matrices gives a better shape of the affinity matrix for spectral clustering than using a single affinity matrix. In experiments, we have shown that our new algorithm gives better results than other state of the art algorithms for real image sequences. However, we have also noticed our method can fail when the images have articulated objects because our algorithm does not address the problem of articulated motion directly. Improving the performance of the algorithm in the case of articulated motion is a study for future work.

8. Acknowledgements

We would like to thank Fabien Lauer for the discussion and experimental results on his algorithm [4]. This work was funded by the European Research Council under ERC Starting Grant agreement 204871-HUMANIS.

References

[1] J. Costeira and T. Kanade. A multibody factorization method for independently moving objects. *International Journal of Computer Vision*, 29(3):159–179, 1998.

[2] N. da Silva and J. Costeira. Subspace segmentation with outliers: A grassmannian approach to the maximum consensus subspace. *IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008*, pages 1–6, 2008.

[3] C. Elkan. Using the triangle inequality to accelerate k-means. *In Proceedings of the Twentieth International Conference on Machine Learning*, pages 147–153, Jan 2003.

[4] F. Lauer and C. Schnörr. Spectral clustering of linear subspaces for motion segmentation. *The 12th IEEE International Conference on Computer Vision*, 2009.

[5] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. *Advances in Neural Information Processing Systems (NIPS)*, Jun 2001.

[6] S. Rao, R. Tron, R. Vidal, and Y. Ma. Motion segmentation via robust subspace separation in the presence of outlying, incomplete, or corrupted trajectories. *IEEE Conference on Computer Vision and Pattern Recognition*, Jan 2008.

[7] K. Schindler, U. James, and H. Wang. Perspective n-view multibody structure-and-motion through model selection. *Lecture Notes in Computer Science*, Jan 2006.

[8] K. Schindler and D. Suter. Two-view multibody structure-and-motion with outliers through model selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(6):983–995, 2006.

[9] J. Shi and J. Malik. Motion segmentation and tracking using normalized cuts. *Computer Vision, 1998. Sixth International Conference on*, pages 1154–1160, 1998.

[10] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Jan 2000.

[11] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization approach. *International Journal of Computer Vision*, 9(2):137–154, November 1992.

[12] R. Tron and R. Vidal. A benchmark for the comparison of 3-d motion segmentation algorithms. *IEEE Conference on Computer Vision and Pattern Recognition*, 4(5), 2007.

[13] R. Vidal and R. Hartley. Motion segmentation with missing data using powerfactorization and gpca. *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, Jan 2004.

[14] U. von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

[15] J. Yan and M. Pollefeys. A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate. *Lecture Notes in Computer Science*, 3954:94, 2006.

[16] J. Yan and M. Pollefeys. Recovering articulated non-rigid shapes, motions and kinematic chains from video. *Lecture Notes in Computer Science*, 4069:90, 2006.

[17] J. Yan and M. Pollefeys. A factorization-based approach for articulated nonrigid shape, motion and kinematic chain recovery from video. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(5):865–877, 2008.

[18] L. Zelnik-Manor and M. Irani. Degeneracies, dependencies and their implications in multi-body and multi-sequence factorizations. *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, 2:II– 287–93 vol.2, 2003.