Video Motion Segmentation Using New Adaptive Manifold Denoising Model

Dijun Luo, Heng Huang
The University of Texas at Arlington
500 UTA Blvd., Arlington, TX 76019
dijun.luo@gmail.com, heng@uta.edu

Abstract

Video motion segmentation techniques automatically segment and track objects and regions from videos or image sequences as a primary processing step for many computer vision applications. We propose a novel motion segmentation approach for both rigid and non-rigid objects using adaptive manifold denoising. We first introduce an adaptive kernel space in which two feature trajectories are mapped into the same point if they belong to the same rigid object. After that, we employ an embedded manifold denoising approach with the adaptive kernel to segment the motion of rigid and non-rigid objects. The major observation is that the non-rigid objects often lie on a smooth manifold with deviations which can be removed by manifold denoising. We also show that performing manifold denoising on the kernel space is equivalent to doing so on its range space, which theoretically justifies the embedded manifold denoising on the adaptive kernel space. Experimental results indicate that our algorithm, named Adaptive Manifold Denoising (AMD), is suitable for both rigid and non-rigid motion segmentation. Our algorithm works well in many cases where several state-of-the-art algorithms fail.

1. Introduction

Video motion segmentation is a topic in computer vision and image understanding which aims to measure and predict the motions of moving objects over time in image sequences or videos and is a crucial computer vision processing step for many real world applications. At the early stages of studying this problem, researchers focused on 2-D motion segmentation techniques, *i.e.*, they aimed to separate each frame of a video sequence into different regions of a coherent 2-D motion (optical flow). Later on, researchers incorporated scenes which contain different moving objects, and tried to identify each object and its motions as a coherent entity. In such cases, the segmentation task has to be performed with the assumption of motions in 3-D space, not only in 2-D. This has motivated several works on 3-D motion segmentation during the last decade, which can be roughly separated into two categories: affine methods and perspective methods [11]. In this paper, we focus on affine methods, which have shown better performance than perspective methods.

For affine methods, previous research assumed that the key points on the same object lie on a strict affine subspace, which is not true in real applications especially when the non-rigid objects appear. Even worse, similar objects often share common regions or subspaces on the noisy manifold, which makes it difficult for traditional approaches to partition the objects. Figure 1 illustrates a typical example in which traditional approaches fail to separate motions of different objects, but our new manifold denoising method succeeds. In this sequence, two objects (represented by pink and blue key points from each object) are slowly moving and share similar patterns (see **a**). A kernel PCA (principal component analysis) view shows that pink and blue point groups are entangled together on a connected smooth manifold with deviations. Since the two pink and blue groups are not linearly separable, it is difficult for traditional approaches to separate them (see **b**). However, after the manifold denoising, they can be easily separated (see **c**). Details of the experimental setting will be provided in Section 4.

Upon these observations we propose a novel motion segmentation method which assumes the trajectories of object points lie on a smooth manifold and performs adaptive manifold denoising to obtain the segmentation structures. Our model does not require the rigidity constraints on the objects and thus can be applied in more applications where the observed scenes are complicated.

The major difficulty of the manifold denoising is that it requires an explicit linear space, which is not available in our kernel space. Thus, the manifold denoising cannot be directly used. To address this problem, we propose the embedded manifold denoising method which performs a kernel PCA to remove the null space. We rigorously prove that performing manifold denoising on the kernel space is equivalent to doing so on its range space.

We validate our method, named Adaptive Manifold De-

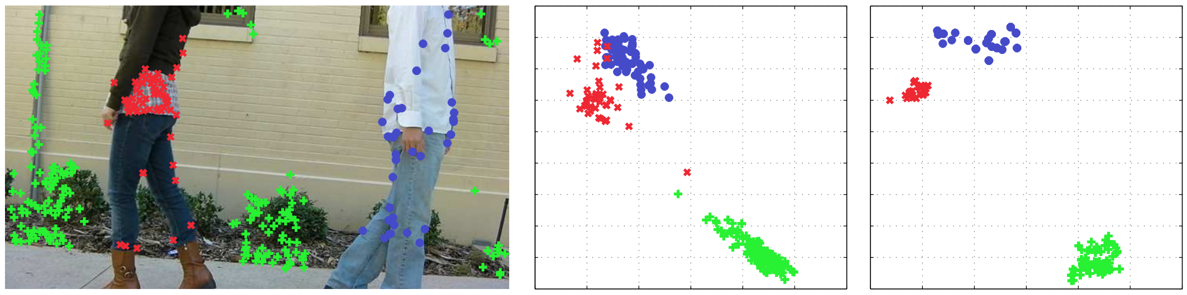

Figure 1. Two moving objects closely lie on smooth manifold. **(a)**: The ground truth segmentations of the key points at the first frame. **(b)**: A kernel PCA view (x-axis and y-axis are the first and second principal components of matrix $\mathbf{W}$ in Equation (25), details will be given in Section 3.4.) of the original manifold. **(c)**: A kernel PCA view of the manifold using manifold denoising.

noising (AMD), on both rigid and non-rigid motions. Experimental results indicate that the proposed algorithm has good performance in both rigid and non-rigid motion segmentation. Our algorithm works well in many cases where several state-of-the-art algorithms fail.

2. Multi-Body Video Motion Segmentation

Recently increasing interests focused on extending motion estimation methods to scenes having multiple moving objects. This is empirically useful in many real world applications. Motion segmentation is to partition the points of objects according to their different motions before applying standard motion estimation techniques to each group.

Here we give a short introduction of the geometry of the 3-D motion segmentation problem from multiple affine views and show that it is equivalent to clustering multiple low-dimensional linear subspaces of a high-dimensional space. This problem can be stated as follows. Given point trajectories corresponding to C objects undergoing C different rigid-body motions relative to the camera, cluster the trajectories according to the C motions.

Moreover, most real applications deal with both rigid and non-rigid objects. For example, a robot collects information from a rigid environment, meanwhile, it also needs to identify non-rigid objects such as human beings. Typically, previous approaches did not intrinsically handle rigid and non-rigid objects simultaneously. However, our new proposed algorithm is able to handle both cases well. For the convenience of discussion, we start from the introduction of rigid-body motion.

2.1. Motion Subspace of A Rigid-Body Motion

The camera transformation is usually assumed as *affine*, thus it is well known that $\mathbf{x}_{fp} \in \mathbb{R}^2$, which is the inhomogeneous coordinate of the image of point $\mathbf{X}_p \in P^3$ in frame f, satisfies the projection equation

$$\mathbf{x}_{fp} = \mathbf{A}_f \mathbf{X}_p, \tag{1}$$

where $\mathbf{A}_f \in \mathbb{R}^{2\times 4}$ is the affine camera projection matrix for frame f, which depends on the relative position between the object and the camera at frame f.

Let $\mathbf{B}_1 \in \mathbb{R}^{2F\times P}$ be the matrix whose P columns are the image point trajectories $\{\mathbf{x}_{fp}\}_{p=1}^{P}$. It follows Eq. (1) that $\mathbf{B}_1$ can be decomposed into a motion matrix $\mathbf{M}_1 \in \mathbb{R}^{2F\times 4}$ and a structure matrix $\mathbf{S}_1 \in \mathbb{R}^{P\times 4}$ as:

$$\mathbf{B}_1 = \mathbf{M}_1 \mathbf{S}_1^T. \tag{2}$$

This is an ideal case. In real world applications, however, this decomposition can not be applied exactly. In this paper, we assume that the noise is added on a smooth manifold with heat kernel diffusion process, which will be discussed in the following section.

2.2. Motion Subspace of Multi-Object Rigid-Body Motions

Let's assume that P trajectories $\{\mathbf{x}_{fp}\}_{p=1}^{P}$ are derived from C objects with respect to C rigid-body motions relative to a moving camera with affine transformation. The 3-D motion segmentation problem is the task of partitioning these P trajectories into the C moving objects. Suppose the trajectories associated with each object are located in a d_i-dimensional linear subspace of $\mathbb{R}^{2F}$, the 3-D motion segmentation problem is equivalent to clustering a set of points into C subspaces of $\mathbb{R}^{2F}$ of unknown dimensions $d_i \in \{2,3,4\}$ for $i = 1, \cdots, C$. Notice that the data matrix can be written as

$$\mathbf{B} = [\mathbf{B}_1, \mathbf{B}_2, \cdots, \mathbf{B}_C]\,\Gamma \in \mathbb{R}^{2F\times P}, \tag{3}$$

where $\mathbf{B}_i \in \mathbb{R}^{2F\times P}$ are the P_i trajectories associated with the ith moving object, $P = \sum_{i=1}^{C} P_i$, and $\Gamma^T \in \mathbb{R}^{P\times P}$ is an unknown permutation matrix.

The purpose of the motion segmentation is to identify the permutation matrix Γ such that:

$$\mathbf{B} = \mathbf{M}\mathbf{S}^T\Gamma, \tag{4}$$

where $\mathbf{S}$ is a block diagonal matrix. There are several approaches to derive such decomposition, *e.g.* [6, 8]. In this paper we focus on direct clustering approaches by assuming that the $\mathbf{B}$ is generated by a smooth low-dimension manifold with heat diffusion process, *i.e.* the subspaces of the image of $\mathbf{B}$ can be considered as points on the Grassmannian of $\mathbb{R}^{2F}$.

3. Adaptive Manifold Denoising

Adaptive manifold denoising is designed to remove the noise introduced by the sensors, the key point matching, and the non-rigid effect. Our method is motivated from previous research of manifold denoising [3]. However, the previous denoising method cannot be applied directly to solve the motion segmentation task. In this section, we provide new techniques to solve motion segmentation via manifold denoising.

3.1. Manifold Denoising

In manifold denoising model [3], the data points are assumed to lie on an unknown p-dimensional manifold M with noise, where the dimension p is the number of independent parameters in the data. More specifically, we assume that the data are mapped from a smooth manifold $i : \mathbf{M} \to \mathbb{R}^d$ to the feature space $\mathbb{R}^d$, where d is the original dimension of the data, and that the data are perturbed by noise with the following form,

$$\mathbf{X} = i(\mathbf{Z}) + \epsilon, \tag{5}$$

where $\mathbf{Z}$ is the coordinate of the manifold $\mathbf{M}$. For convenience, we further assume that the noise is Gaussian, *i.e.* $\epsilon \sim \mathcal{N}(0, \sigma)$. The distribution of $\mathbf{X}$ is

$$P_{\mathbf{X}}(\mathbf{x}) = \left(2\pi\sigma^2\right)^{-\frac{d}{2}} \int_{\mathbf{M}} e^{-\frac{\|\mathbf{x}-i(\mathbf{z})\|^2}{2\sigma^2}} p(\mathbf{z}) dV(\mathbf{z}), \tag{6}$$

where $p(\mathbf{z})$ is the distribution of the $\mathbf{Z}$ on the manifold. Note that the Gaussian measure is equivalent to the heat kernel of diffusion process on $\mathbb{R}^d$.

The key idea of manifold denoising is to reverse the diffusion process which is the optimal estimation of the intrinsic manifold $\mathbf{M}$. According to [3], the diffusion process is defined by

$$\partial_t \mathbf{X}^T = -\gamma \mathbf{L} \mathbf{X}^T, \tag{7}$$

where $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n)$ is the $d \times n$ data matrix and $\gamma > 0$ is the diffusion constant. And $\mathbf{L}$ is the $n \times n$ discrete *Laplacian* matrix:

$$\mathbf{L} = \mathbf{D} - \mathbf{W}, \tag{8}$$

where $\mathbf{W}$ includes similarity measurements between data points sampled from the manifold $\mathbf{M}$ and $\mathbf{D} = \mathbf{diag}(d_1, d_2, \cdots, d_n), d_i = \sum_{j=1}^n W_{ij}$. Due to the diffusion model here, we always use the Gaussian similarity:

$$W_{ij} = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right), \tag{9}$$

where σ is a model parameter. To solve the differential equation (7), an implicit Euler-scheme is employed,

$$\left(\mathbf{X}^{t+1}\right)^T - \left(\mathbf{X}^t\right)^T = -\delta t \gamma \mathbf{L} \left(\mathbf{X}^{t+1}\right)^T, \tag{10}$$

which leads to

$$\mathbf{X}^{t+1} = \mathbf{X}^t (\mathbf{I} + \delta t \gamma \mathbf{L})^{-1}. \tag{11}$$

Notice that $(\mathbf{I}+\delta t \gamma \mathbf{L})^{-1}$ is symmetric here. In order to handle a kernel matrix as input, we solve the following problem:

$$\phi(\mathbf{X})^{t+1} = \phi(\mathbf{X})^t (\mathbf{I} + \delta t \gamma \mathbf{L})^{-1}, \tag{12}$$

where $\phi(\mathbf{X}) = [\phi(\mathbf{x}_1), \phi(\mathbf{x}_2), \cdots, \phi(\mathbf{x}_n)]$ and ϕ is the kernel map from $\mathbf{x}$ to Hilbert space H with inner product $\langle \cdot, \cdot \rangle_H$, which can be evaluated for elements x, y by means of a kernel function $K(x, y) := \langle \phi(x), \phi(y) \rangle_H$.

3.2. Embedded Manifold Denoising

The major difficulty in solving Eq. (12) is that we typically do not know the explicit form of $\phi(\mathbf{x})$, hence solving this equation is intractable in motion segmentation task. To solve this problem, we propose the embedded manifold denoising method to use a proper embedded data $\mathbf{U} = \mathbf{P}\phi(\mathbf{X})$ and perform manifold on $\mathbf{U}$,

$$\mathbf{U}^{t+1} = \mathbf{U}^t (\mathbf{I} + \delta t \gamma \mathbf{L})^{-1}. \tag{13}$$

We will prove that updating using Eq. (12) and Eq. (13) have theoretically equivalent results. To be more specific, we prove the following theorem:

Theorem 1 *If $\forall i, \phi(\mathbf{x}_i)$ can be written in the following form*

$$\phi(\mathbf{x}_i) = \mathbf{Q} \begin{bmatrix} \mathbf{u}_i \\ 0 \end{bmatrix}, \tag{14}$$

where $\mathbf{Q} = \mathbf{Q}^{\parallel} \oplus \mathbf{Q}^{\perp}$ is the full orthonormal basis, $\mathbf{Q}^T \mathbf{Q} = \mathbf{I}$, $\mathbf{Q}^{\parallel}$ is the range space and $\mathbf{Q}^{\perp}$ is the null space, and $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_n]$, then the following equation holds,

$$\phi(\mathbf{X}^t) = \mathbf{Q}^{\parallel} \mathbf{U}^t, \tag{15}$$

where $\phi(\mathbf{X}^t)$ and $\mathbf{U}^t$ are the t-th iteration results using Eq. (12) and Eq. (13), respectively.

Proof: We first show that computing graph Laplacian matrix $\mathbf{L}$ using $\phi(\mathbf{X})$ and $\mathbf{U}$ is equivalent. Because

$$\begin{aligned}
&\|\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\|^2 \\
=&\|\phi(\mathbf{x}_i)\|^2 + \|\phi(\mathbf{x}_j)\|^2 - 2\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j) \\
=&[\mathbf{u}_i^t, 0]\mathbf{Q}^T \mathbf{Q} \begin{bmatrix} \mathbf{u}_i \\ 0 \end{bmatrix} + [\mathbf{u}_j^t, 0]\mathbf{Q}^T \mathbf{Q} \begin{bmatrix} \mathbf{u}_j \\ 0 \end{bmatrix} - 2[\mathbf{u}_i^t, 0]\mathbf{Q}^T \mathbf{Q} \begin{bmatrix} \mathbf{u}_j \\ 0 \end{bmatrix} \\
=&\|\mathbf{u}_i\|^2 + \|\mathbf{u}_j\|^2 - 2\mathbf{u}_i^T \mathbf{u}_j \\
=&\|\mathbf{u}_i - \mathbf{u}_j\|^2,
\end{aligned}$$

for any i, j, similarity $\mathbf{W}_{ij}$ computed by Eq. (9) is identical to use $\phi(\mathbf{X})$ and $\mathbf{U}$, so is Laplacian matrix $\mathbf{L}$. Thus

$$\phi(\mathbf{X}^1) = \phi(\mathbf{X})^0 (\mathbf{I}+\delta t \gamma \mathbf{L})^{-1} = \mathbf{Q}^{\parallel} \mathbf{U}^0 (\mathbf{I}+\delta t \gamma \mathbf{L})^{-1} = \mathbf{Q}^{\parallel} \mathbf{U}^1. \tag{16}$$

For the same reason, $\phi(\mathbf{X}^t) = \mathbf{Q}^{\parallel}\mathbf{U}^t, t = 2, 3, \cdots$. In order to obtain $\mathbf{U}$, one can compute the SVD (Singular Value Decomposition) of $\phi(\mathbf{X})$,

$$\phi(\mathbf{X}) = [\mathbf{V}_1, \mathbf{V}_2] \begin{bmatrix} \mathbf{S} \\ 0 \end{bmatrix} \tilde{\mathbf{U}}^T. \tag{17}$$

By using $\mathbf{Q}^{\parallel} = \mathbf{V}_1$, $\mathbf{Q}^{\perp} = \mathbf{V}_2$ and $\mathbf{U} = \tilde{\mathbf{U}}\mathbf{S}$, we fulfill the requirement in Eq. (14). □

Theorem 1 suggests that performing manifold denoising on the kernel space and on its PCA subspace are equivalent. In our implementation, given a kernel matrix K, we perform the eigenvector decomposition:

$$\mathbf{K} = \tilde{\mathbf{U}}\Sigma\tilde{\mathbf{U}}^T, \tag{18}$$

and let $\mathbf{U} = \tilde{\mathbf{U}}\Sigma^{1/2}$. One can easily show that Eqs. (17) and (18) are equivalent with $\Sigma = \mathbf{S}^2$. Notice that in the updating of Eq. (13), we do not need to know $\mathbf{Q}$.

We show a typical case of employing manifold denoising on a motion segmentation task of three objects in Fig. 1. **(b)** shows the PCA view on the original manifold, and **(c)** shows the PCA view of the data after performing the embedded manifold denoising. The manifold captures the intrinsic structure of the motion subspace and removes the perturbation noise. Clearly it is easier to separate the moving objects after the manifold denoising is applied.

3.3. Adaptive Kernel for Motion Segmentation

The embedded manifold denoising proposed above requires a kernel matrix as input. Here we show how to obtain an adaptive kernel matrix with local enhancement by utilizing the local structure of motion subspace. Let $\{(\mathbf{c}_{i1}, \mathbf{c}_{i2})\}, i = 1, 2, \cdots, m$ be the set of m pairwise points which are identified to be in the same object. We expect to embed the points with a Hilbert map ϕ such that points from the same object are mapped into the same point by projecting the feature space to the null space of the constraint vectors.

Let $\mathbf{A}$ be the $m \times d_\phi$ dimensional constraint matrix:

$$\mathbf{A} = \begin{pmatrix} (\phi(\mathbf{c}_{11}) - \phi(\mathbf{c}_{12}))^T \\ \vdots \\ (\phi(\mathbf{c}_{m1}) - \phi(\mathbf{c}_{m2}))^T \end{pmatrix}. \tag{19}$$

Obviously if

$$\mathbf{P} = \mathbf{I} - \mathbf{A}^T \left(\mathbf{A}\mathbf{A}^T\right)^+ \mathbf{A}, \tag{20}$$

then

$$\mathbf{P}\mathbf{A}^T = \left(\mathbf{I} - \mathbf{A}^T \left(\mathbf{A}\mathbf{A}^T\right)^+ \mathbf{A}\right)\mathbf{A}^T = \mathbf{A}^T - \mathbf{A}^T = 0, \tag{21}$$

indicating that

$$\mathbf{P}\phi(\mathbf{c}_{i1}) = \mathbf{P}\phi(\mathbf{c}_{i2}), i = 1, 2, \cdots, m, \tag{22}$$

i.e., under the projection $\mathbf{P}$, the points from the same object are projected into the same point. Similar technique was used in [12] with interesting results.

We use the following projection as a new mapping

$$\hat{\phi}(\mathbf{x}) = \mathbf{P}\phi(\mathbf{x}). \tag{23}$$

By applying the projection, the projected kernel function is given by

$$\begin{aligned} \hat{\mathbf{K}}(\mathbf{x}, \mathbf{y}) &= \hat{\phi}(\mathbf{x})^T\hat{\phi}(\mathbf{y}) = \phi(\mathbf{x})^T\mathbf{P}^T\mathbf{P}\phi(\mathbf{y}) \qquad (24) \\ &= \phi(\mathbf{x})^T\mathbf{P}\phi(\mathbf{y}) \\ &= \phi(\mathbf{x})^T(\mathbf{I} - \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^+\mathbf{A}\phi(\mathbf{y}) \\ &= \mathbf{K}(\mathbf{x}, \mathbf{y}) - \mathbf{K}(\phi(\mathbf{x}), \mathbf{A})^T(\mathbf{A}\mathbf{A}^T)^+\mathbf{K}(\phi(\mathbf{y}), \mathbf{A}). \end{aligned}$$

We notice that

$$\begin{aligned} \mathbf{P}\mathbf{P}^T &= \left(\mathbf{I} - \mathbf{A}^T \left(\mathbf{A}\mathbf{A}^T\right)^+ \mathbf{A}\right)\left(\mathbf{I} - \mathbf{A}^T \left(\mathbf{A}\mathbf{A}^T\right)^+ \mathbf{A}\right)^T \\ &= \mathbf{I} - \mathbf{A}^T \left(\mathbf{A}\mathbf{A}^T\right)^+ \mathbf{A} = \mathbf{P}. \end{aligned}$$

The modified kernel $\hat{\mathbf{K}}$ has the following property:

Theorem 2 *If* $\mathbf{K}$ *is semi-positive definite, then* $\hat{\mathbf{K}}$ *in Eq. (24) is semi-positive definite.*

Proof $\forall \mathbf{x}, \mathbf{x}^T\hat{\mathbf{K}}\mathbf{x} = \mathbf{x}^T\phi(\mathbf{X})^T\mathbf{P}^T\mathbf{P}\phi(\mathbf{X})\mathbf{x} = \sum_i \mathbf{z}_i^2 \geq 0$, where $\mathbf{z}_i = (\mathbf{P}\phi(\mathbf{X})\mathbf{x})_i$ and $\phi(\mathbf{X}) = [\phi(\mathbf{x}_1), \phi(\mathbf{x}_2), \cdots, \phi(\mathbf{x}_n)]$. By definition, $\hat{\mathbf{K}}$ is semi-positive definite. □

The process of constructing Locally Enhanced Kernel is summarized in Algorithm 1.

Algorithm 1 LEK($\mathbf{W}, \mathbf{c}$)

```
    Input: Feature trajectories data $\mathbf{X}$, constraint set $\mathbf{c}$.
    Output: Local Enhanced Kernel $\hat{\mathbf{K}}$.
    $\mathbf{K} = \mathbf{W}$.
    $\mathbf{S}_{ij} = \mathbf{K}_{\mathbf{c}_{i1}\mathbf{c}_{i2}} - 2\mathbf{K}_{\mathbf{c}_{i1}\mathbf{c}_{j2}} + \mathbf{K}_{\mathbf{c}_{j1}\mathbf{c}_{j2}}$, for each $i, j$ in $\mathbf{c}$.
    $\mathbf{K}_1 = [\mathbf{K}_{:,\mathbf{c}_{11}}, \mathbf{K}_{:,\mathbf{c}_{21}}, \mathbf{K}_{:,\mathbf{c}_{m1}}]$,
    $\mathbf{K}_2 = [\mathbf{K}_{:,\mathbf{c}_{12}}, \mathbf{K}_{:,\mathbf{c}_{22}}, \mathbf{K}_{:,\mathbf{c}_{m2}}]$.
    $\hat{\mathbf{K}} = \mathbf{K} - (\mathbf{K}_1 - \mathbf{K}_2)\mathbf{S}^+(\mathbf{K}_1 - \mathbf{K}_2)^T$
    Output: $\hat{\mathbf{K}}$.
```

3.4. Local Subspace Kernel

We employ subspace similarity in Local Subspace Analysis (LSA) as a kernel to determine $\phi(\mathbf{x})$. The LSA algorithm proposed by Yan and Pollefeys in [14] is based on a linear projection and spectral clustering. LSA fits a subspace locally around each projected point. The main steps of the local algorithm are summarized as follows:

Projection: Project the trajectories onto a subspace of dimension $D = rank(\mathbf{B})$ using the SVD of $\mathbf{B}$. The value of D is determined by model selection techniques. The resulting points in $\mathbb{R}^D$ are then projected onto the hypersphere $\mathbf{S}^{D-1}$ by setting their norm to 1.

Local subspace estimation: For each point i, its k nearest neighbors are computed using the angles between the vectors or their Euclidean distance as a metric. After that, a local subspace $\mathbf{B}_i$ is fitted to the point and its neighbors. The dimension d_i of the subspace $\mathbf{B}_i$ depends on the kind of motion (*e.g.*, general motion, purely translational, *etc.*) and the position of the 3-D points (*e.g.* general position, all on the same plane, *etc.*). The dimension d_i is also determined by model selection techniques.

Similarity graph construction and spectral clustering: Compute a similarity matrix between pairwise points $i, j = 1, 2, \cdots, P$ as

$$W_{ij} = e^{-\sum_{m=1}^{d_{ij}} \sin^2(\theta_m)}, \tag{25}$$

where the θ_m are the principal angles between two subspaces $\mathbf{B}_i$ and $\mathbf{B}_j$, and $d_{ij} = \min(\mathbf{dim}(\mathbf{B}_i), \mathbf{dim}(\mathbf{B}_j))$. Spectral clustering [9, 7] is then applied on the similarity matrix $\mathbf{W}$. With the construction of $\mathbf{W}$, we have the following property:

Theorem 3 $\mathbf{W}$ *defined in Eq. (25) is semi-positive definite.*

Combining this property and Theorem 2, we know that the final locally enhanced matrix is a valid kernel matrix. One example of the kernel PCA view of the kernel matrix is visualized in Figure 1 (b). Please notice that this construction is $O(n^2)$, which is similar to other graph-based approaches. For each pair of data points (i, j), we need to compute the principle angles of two subspaces, which requires a SVD decomposition in a typical implementation. This is the most computationally expensive part of our algorithm, as well as LSA algorithms.

3.5. AMD Algorithm

Our AMD method is summarized in Algorithm 2. In step 2, we pick the top αC ranked W_{ij} as the must-link pairs. α is set to 3 in all our experiments. In step 4.2, $\mathbf{e}$ is a column vector with all elements 1. T is the number of iterations needed by the manifold denoising which is determined by the strategy used in [3].

Algorithm 2 **AMD**$(\mathbf{X}, C, \alpha)$

```
Input: Trajectories data $\mathbf{X}$, number of clusters $C$.
Step 1: Compute $\mathbf{K} = \mathbf{W}$ with $\mathbf{X}$ using Eq. (25).
Step 2: Sort the off-diagonal elements in $\mathbf{W}$, pick the top $\alpha C$ pairs as pair set $\mathbf{c}$.
Step 3: $\hat{\mathbf{K}} = \mathbf{LEK}(\mathbf{W}, \mathbf{c})$,. Perform eigenvector decomposition on $\hat{\mathbf{K}}$: $\hat{\mathbf{K}} = \tilde{\mathbf{U}}\Sigma\tilde{\mathbf{U}}^T$, set $\mathbf{U} = \tilde{\mathbf{U}}\Sigma^{1/2}$,
Step 4:
for $t = 1 : T$ do
    Step 4.1: Compute $\tilde{\mathbf{W}}$ using Eq. (9) on $\mathbf{U}$, set $\mathbf{L} = \mathbf{D} - \tilde{\mathbf{W}}$, where $\hat{\mathbf{D}} = \mathbf{diag}(\tilde{\mathbf{W}}\mathbf{e})$
    Step 4.2: $\mathbf{U} \leftarrow \mathbf{U}(\mathbf{I} + \delta t\mathbf{L})^{-1}$.
end for
Step 5: Cluster data points into $C$ partitions on $\mathbf{U}$ using spectral clustering: $\pi_1, \pi_2, \cdots, \pi_C$.
Output: $\Pi = [\pi_1, \pi_2, \cdots, \pi_C]$.
```

4. Experimental Results

We evaluate our algorithm by two data sets. The first one is *Hopkins 155* [4] which is a standard benchmark for motion segmentation. This data set contains both rigid and non-rigid moving objects, but the resolution is low, and there are mismatched keypoints in the sequences. In order to demonstrate the capability of the manifold denoising, we also capture 3 video sequences, in which there are at least two non-rigid moving objects. Figure 1 shows one of these sequences. We capture video in high definition (1280×720) and carefully select the trajectory key points such that the major perturbation stems from the non-rigid movement, rather than from mismatched keypoints or other noise. These sequences are thus suitable to test the manifold denoising effects.

4.1. Compared Methods

GPCA. Generalized Principal Component Analysis (GPCA) is an algebraic method for clustering data lying in multiple subspaces proposed by Vidal *et al.* [13]. For this method, we directly use the code provided by [4].

LSA. LSA[14] is described in Section 3.4. For this method, we use two settings: with 5-nearest neighbors and $4C$-nearest neighbors.

Random Sample Consensus (RANSAC). RANSAC is a statistical method for fitting a model to a cloud of points corrupted with outliers in a statistically robust way [2, 5]. The implementation is provided by [4].

Projective Factorization (PF). PF iteratively does the following: (1) obtain the depths (2) estimate the subspace separation according to the depths [6]. For this method, they need to combine PF with a specific subspace separation approach, we use both PF+GPCA and PF+LSA.

Multi-Stage Learning (MSL). The MSL algorithm is a statistical approach proposed in [10]. This algorithm is based on Costeira's and Kanade's factorization method (CK) [1, 5] and Kanatani's subspace separation method (SS) [5]. For this method, we directly use the code provided by the authors[1] with default settings.

[1]http://www.suri.it.okayama-u.ac.jp/e-program-separate.html

4.2. Data Descriptions

Hopkins 155 contains 155 motion sequences, including 120 2-motion and 35 3-motion sequences. This collection includes three subsets of data: *CheckerBoard, Traffic,* and *Articulated*. It should be noted that in *CheckBoard* subset, all objects are rigid. In *Traffic* subset, the objects are hybrid: some are rigid (cars, roads) and some are non-rigid (pedestrians on roads). In *Articulated* subset most of the objects are non-rigid.

The second data set (marked as *Nonrigid* in the results table) include 3-motion sequences which are designed to test the manifold denoising method. These video sequences are taken with high resolution and mismatched key points are manually removed. The purpose of discarding the mismatched points is to remove all other perturbations except the non-rigid motion effect. We are interested in the behavior of our manifold denoising algorithm under such smooth condition, compared to other subspace segmentation methods.

4.3. Evaluation of Results

We compare all the previously mentioned methods by three standard metrics: clustering accuracy, Normalized Mutual Information (NMI), and purity. The results are shown in Table 1. AMD achieves 98.42% in overall clustering accuracy. We also separately summarize the clustering accuracy of 2-motion and 3-motion (including *Nonrigid*) in Table 2. For 2-motion and 3-motion sequences, the overall clustering accuracies of our method are 98.94% and 97.92%, respectively. In all cases, our method outperforms other state-of-the-art motion segmentation methods.

We present the results on non-rigid motion segmentation (data sets *Traffic* 3-motion and *Nonrigid* sequences) in Figure 2. In the figure, one point with $a\%$ accumulative percentage (x-axis) and $b\%$ (y-axis) accuracy/NMI/purity means there are at least $a\%$ data points which have at least $b\%$ chance to be correctly segmented. One can observe that our method has much better performance in all the metrics on these non-rigid sequences.

The segmentation for GPCA, RANSAC, LSA5, LSA4n, and our method for three sequences are shown in Figure 3. Notice that only the grouping information is shown in this figure, *i.e.* key points marked by the same symbol (also color) in each image are grouped into the same class (object) by corresponding method. It can be observed that our algorithm segments the motion significantly better than the other approaches.

4.4. Highlighted Observations and Discussion

Results show that for all the measurements (accuracy, NMI, and purity), our method outperforms all other methods, as highlighted using bold fonts in Table 1.

For non-rigid motion sequences (*Traffic* and *Nonrigid*), our approach is much better than other methods, especially for *Nonrigid*. The improvement is the result of applying the manifold denoising process. As long as the objects are smooth (in the sense of manifold), manifold denoising works well. The average accuracies of LSA5 and LSA4n are 74.25% and 74.30%, which are worse than our method (97.09%). In *Nonrigid* category, our method achieves a perfect score *w.r.t* the two other performance indices while other approaches are much lower.

5. Conclusion

We proposed a novel video motion segmentation framework using the new embedded manifold denoising and local enhancement kernel function. Because of the existence of noise and outliers, the motion segmentation problem is a very difficult problem in real world applications and the traditional approaches often do not perform very well. By considering the smoothness of the manifold in which trajectories lie, our method is capable of capturing the hidden structure of the moving objects, including both rigid and non-rigid objects. Our algorithm works well in cases where many other start-of-the-art approaches fail, especially in the cases with hybrid rigid and non-rigid objects.

Acknowledgement

This work was partially supported by US NSF IIS-1117965, IIS-1302675, IIS- 1344152.

References

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

[2] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Comm. A.C.M.*, 24(6):381–395, June 1981.

[3] M. Hein and M. Maier. Manifold denoising. In *NIPS*, pages 561–568. MIT Press, 2006.

[4] Johns-Hopkins. `http://www.vision.jhu.edu/`.

[5] K. Kanatani. Motion segmentation by subspace separation and model selection. In *ICCV*, pages II: 586–591, 2001.

[6] T. Li, V. Kallem, D. Singaraju, and R. Vidal. Projective factorization of multiple rigid-body motions. In *CVPR*, pages 1–6, 2007.

[7] D. Luo, H. Huang, C. Ding, and F. Nie. On the eigenvectors of p-laplacian. *Machine Learning*, 81(1):37–51, 2010.

[8] D. Luo, F. Nie, C. Ding, and H. Huang. Multi-subspace representation and discovery. pages 405–420, 2011.

[9] A. Y. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*. MIT Press, 2002.

[10] Y. Sugaya and K. Kanatani. Geometric structure of degeneracy for multi-body motion segmentation. In *Statistical Methods in Video Processing*, pages 13–25, 2004.

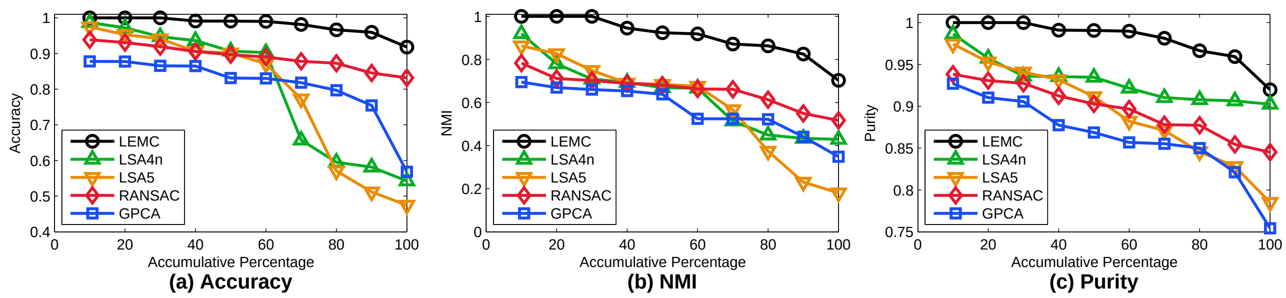

Figure 2. Detailed comparison of GPCA, RANSAC, LSA with 5-nearest neighbors (LSA5), LSA with 4n-nearest neighbors (LSA4n), and our method (AMD) on *Traffic* 3-motion sequences.

Table 1. Accuracy comparisons of GPCA, RANSAC, LSA with 5-nearest neighbors (LSA5), LSA with 4n-nearest neighbors (LSA4n), and our method (AMD). Avg is the average measurements and Med is the median.

	GPCA		RANSAC		LSA5		LSA4n		AMD	
Accuracy	Avg	Med	Avg	Med	Avg	Med	Avg	Med	Avg	Med
Check2	93.87	98.37	94.07	98.01	91.31	94.82	96.95	99.74	98.82	100.00
Check3	69.07	67.54	76.26	80.43	69.42	67.63	94.81	98.89	98.92	99.59
Traffic2	98.37	100.00	95.98	100.00	97.70	99.02	95.09	98.63	97.18	99.42
Traffic3	78.24	81.82	88.38	87.80	74.25	77.24	74.30	65.73	97.09	98.13
Articul2	95.51	99.74	96.53	98.82	96.04	99.47	98.24	99.19	98.50	99.52
Articul3	83.68	79.77	80.53	76.30	73.91	78.03	96.56	98.84	96.39	99.33
Nonrigid	85.32	85.58	87.76	89.68	87.29	87.10	91.79	92.08	**100.00**	**100.00**
ALL	89.85	97.35	91.23	97.25	88.32	94.71	95.33	99.27	**98.42**	**99.79**
NMI										
Check2	76.45	87.47	76.77	85.99	67.32	68.62	89.25	97.32	95.25	100.00
Check3	44.81	43.79	52.86	53.97	43.80	43.74	88.15	94.61	95.94	97.30
Traffic2	89.65	100.00	83.12	100.00	84.02	88.25	79.88	87.22	86.33	93.31
Traffic3	52.29	52.46	64.86	66.32	54.14	56.67	60.27	51.41	86.44	87.16
Articul2	81.35	94.95	83.34	91.36	81.69	96.18	87.23	92.32	89.38	94.39
Articul3	61.96	62.16	62.34	59.63	48.53	63.63	87.81	95.42	88.35	96.75
Nonrigid	63.31	64.96	72.94	74.83	71.36	73.02	69.38	70.71	**100.00**	**100.00**
ALL	72.66	82.67	73.94	80.49	67.22	69.46	85.56	93.36	**92.74**	**97.99**
Purity										
Check2	94.24	98.37	94.88	98.01	91.98	94.82	97.74	99.74	98.89	100.00
Check3	75.28	75.72	80.35	81.52	74.69	74.59	95.53	98.89	98.92	99.59
Traffic2	98.49	100.00	98.13	100.00	98.36	99.02	97.79	98.63	97.18	99.42
Traffic3	86.07	85.68	88.96	87.80	88.70	88.21	92.85	92.15	97.11	98.13
Articul2	95.51	99.74	96.53	98.82	96.10	99.47	98.24	99.19	98.50	99.52
Articul3	83.68	79.77	82.66	76.30	73.91	78.03	96.56	98.84	96.39	99.33
Nonrigid	89.32	89.38	88.03	89.68	87.66	87.10	95.77	94.27	**100.00**	**100.00**
ALL	91.46	97.35	92.70	97.25	90.23	94.71	97.18	99.29	**98.45**	**99.79**

Table 2. Clustering accuracy comparison of GPCA, RANSAC, LSA5, LSA4n, MSL, PF+GPCA, PF+LSA and AMD on the whole dataset.

		GPCA	RANSAC	LSA5	LSA4n	MSL	PF+GPCA	PF+LSA	AMD
2-motion	Average	89.94	90.97	88.36	96.13	95.86	96.70	96.77	**98.49**
	Median	97.89	97.48	94.82	99.42	100	99.47	99.47	99.80
3-motion	Average	89.93	92.78	88.22	92.08	90.27	81.32	93.77	**97.92**
	Median	92.79	96.15	96.37	98.68	97.67	84.14	98.38	99.10

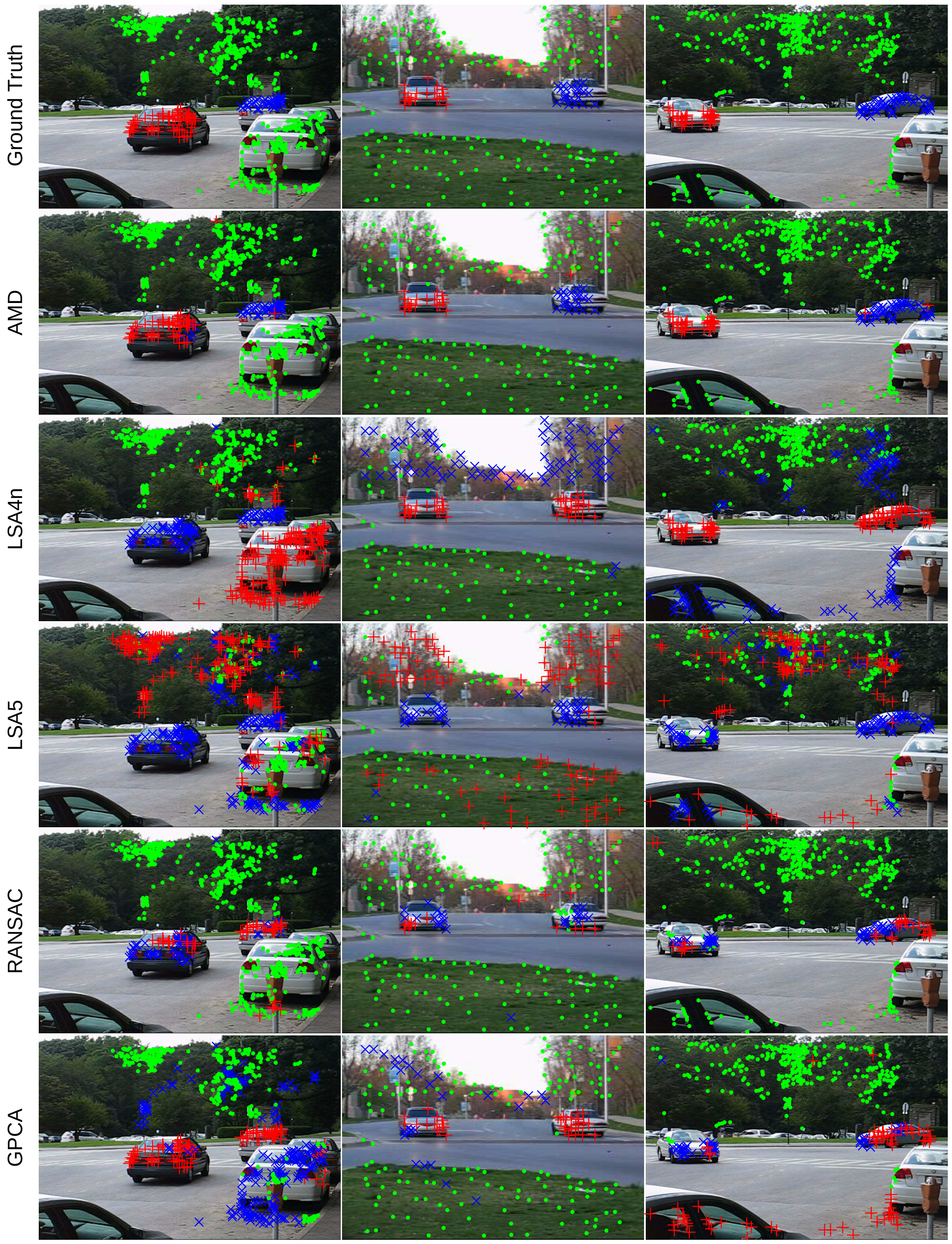

Figure 3. Key points segmentation results for GPCA, RANSAC, LSA5, LSA4n, and our method (AMD). The ground truth is also shown. Key points marked by the same symbol (color) in each image are grouped into the same class (object) by corresponding method.

[11] R. Tron and R. Vidal. A benchmark for the comparison of 3-D motion segmentation algorithms. In *CVPR*, 2007.

[12] O. Tuzel, F. Porikli, and P. Meer. Kernel methods forweakly supervised mean shift clustering. In *ICCV*, 2009.

[13] R. Vidal, Y. Ma, and S. Sastry. Generalized principal component analysis (GPCA). *IEEE Trans. Pattern Analysis and Machine Intelligence*, 27(12):1945–1959, Dec. 2005.

[14] J. Y. Yan and M. Pollefeys. A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate. In *ECCV*, 2006.