Robust Multi-body Motion Tracking Using Commute Time Clustering

Huaijun Qiu and Edwin R. Hancock

Department of Computer Science, University of York
York, YO10 5DD, UK

Abstract. The presence of noise renders the classical *factorization method* almost impractical for real-world multi-body motion tracking problems. The main problem stems from the effect of noise on the shape interaction matrix, which looses its block-diagonal structure and as a result the assignment of elements to objects becomes difficult. The aim in this paper is to overcome this problem using graph-spectral embedding and the k-means algorithm. To this end we develop a representation based on the commute time between nodes on a graph. The commute time (i.e. the expected time taken for a random walk to travel between two nodes and return) can be computed from the Laplacian spectrum using the discrete Green's function, and is an important property of the random walk on a graph. The commute time is a more robust measure of the proximity of data than the raw proximity matrix. Our embedding procedure preserves commute time, and is closely akin to kernel PCA, the Laplacian eigenmap and the diffusion map. We illustrate the results both on the synthetic image sequences and real world video sequences, and compare our results with several alternative methods.

1 Introduction

Multi-body motion tracking is a challenging problem which arises in shape from motion, video coding, the analysis of movement and surveillance. One of the classical techniques is the *factorization method* of Costeira and Kanade [4]. The basic idea underpinning this method is to use singular value decomposition (SVD) to factorize the feature trajectory matrix into a motion matrix and a shape matrix. The shape interaction matrix is found by taking outer product of the right eigen-vector matrix, and can be used to identify the independently moving objects present. Gear [7] has developed a related method based on the reduced row echelon form of the matrix, and object separation is achieved using probabilistic analysis on a bipartite graph. Both methods work well in the ideal case when there is no noise (i.e. feature-point jitter) and outliers are not present. However, real-world image sequences are usually contaminated by the two types of noise. There have been several attempts to overcome this problem. For instance, Ichimura [9] has improved the *factorization method* by using a discriminant criterion to threshold-out the noise and outliers.

Rather than working with a matrix derived from the data, some researchers place the emphasis on the original data. Kanatani [10, 19, 18] developed a subspace separation method by incorporating dimension correction and model selection. Wu et al [21] argue that the subspaces associated with the different objects are not only distinct, but also orthogonal. They hence employ an orthogonal subspace decomposition method to

A. Leonardis, H. Bischof, and A. Pinz (Eds.): ECCV 2006, Part I, LNCS 3951, pp. 160–173, 2006.

separate objects. This idea is further extended by Fang et al who use independent subspaces [6] and multiple subspace inference analysis [5]. In addition to attempting to improve the behaviour of the factorization method under noise, there has been a considerable effort at overcoming problems such as degeneracy, uncertainty and missing data [8, 22].

The factorisation method is clearly closely akin to graph-spectral methods used in clustering, since it uses the eigenvector methods to determine the class-affinity of sets of points. In fact Weiss [20] has presented a unifying view of spectral clustering methods, and this includes the factorization method. There has been some dedicated effort devoted to solving the object separation problem using spectral clustering methods. Park et al [12] have applied a multi-way min-max cut clustering method to the shape interaction matrix. Here the shape-interaction matrix is used as a cluster indicator matrix and noise compensation is effected using a combination of spectral clustering and subspace separation methods.

In general graph theoretic clustering methods aim to locate clusters of nodes that minimize the cut or disassociation, while maximizing the association. One of the most successful methods is the normalised cut of Shi and Malik [16] which as been applied to image segmentation problems. Pavan and Pelillo [13] have shown how the performance of this method can be improved using a finer measure of cluster cohesion based on dominant-sets. In a recent paper Qiu and Hancock [14] have shown how commute time can be used to characterise the mutual affinity of nodes. The commute time is the expected time taken for a random walk to travel between two nodes and return. It is determined by the Green's function or pseudo inverse of the Laplacian matrix, and can hence be conveniently computed using the Laplacian spectrum.

The commute time has properties that can lead to clusters of nodes that increase both the dissociation and the association. A pair of nodes in the graph will have a small commute time value if one of three conditions is satisfied. The first of these is that they are close together, i.e. the length of the path between them is small. The second case is if the sum of the weights on the edges connecting the nodes is small. Finally, the commute time is small if the pair of nodes are connected by many paths. Hence, the commute time can lead to a finer measure of cluster cohesion than the simple use of edge-weight which underpins algorithms such as the normalized cut [16].

The aim in this paper is to explore whether an embedding based on commute time can be used to solve the problem of computing the shape-interaction matrix in a robust manner. We use the shape-interaction matrix Q as a data-proximity weight matrix, and compute the associated Laplacian matrix (the degree matrix minus the weight matrix). The aim is to embed feature points in a space that preserves commute time. The embedding co-ordinate matrix is found the premultiplying the transpose of the Laplacian eigenvector matrix by the inverse square-root of the eigenvalue matrix. Under the embedding nodes which have small commute time are close, and those which have a large commute time are distant. This allows us to separate the objects in the embedded subspace by applying simple k-means clustering. There are of course many graph-spectral embedding algorithms reported in the literature, and recent and powerful additions include kernel PCA [15], the Laplacian eigenmap [1] and the diffusion map [3]. We explore the relationship of the commute-time embedding to these alternatives.

2 Factorization Method Review

Suppose there are N objects moving independently in a scene and the movement is acquired by an affine camera as F frames. In each frame, P feature points are tracked and the coordinate of the ith point in the fth frame is given by (x_i^f, y_i^f). Let X and Y denote two $F \times P$ matrices constructed from the image coordinates of all the points across all of the frames satisfying: $X = \begin{bmatrix} x_1^1 & x_2^1 & \cdots & x_P^1 \\ x_1^2 & x_2^2 & \cdots & x_P^2 \\ \vdots & \vdots & \ddots & \vdots \\ x_1^F & x_2^F & \cdots & x_P^F \end{bmatrix}$ and $Y = \begin{bmatrix} y_1^1 & y_2^1 & \cdots & y_P^1 \\ y_1^2 & y_2^2 & \cdots & y_P^2 \\ \vdots & \vdots & \ddots & \vdots \\ y_1^F & y_2^F & \cdots & y_P^F \end{bmatrix}$. Each row in the two matrices above corresponds to a single frame and each column corresponds to a single point. The two coordinate matrices can be stacked to form the matrix $W = \left[\frac{X}{Y}\right]_{2F \times P}$.

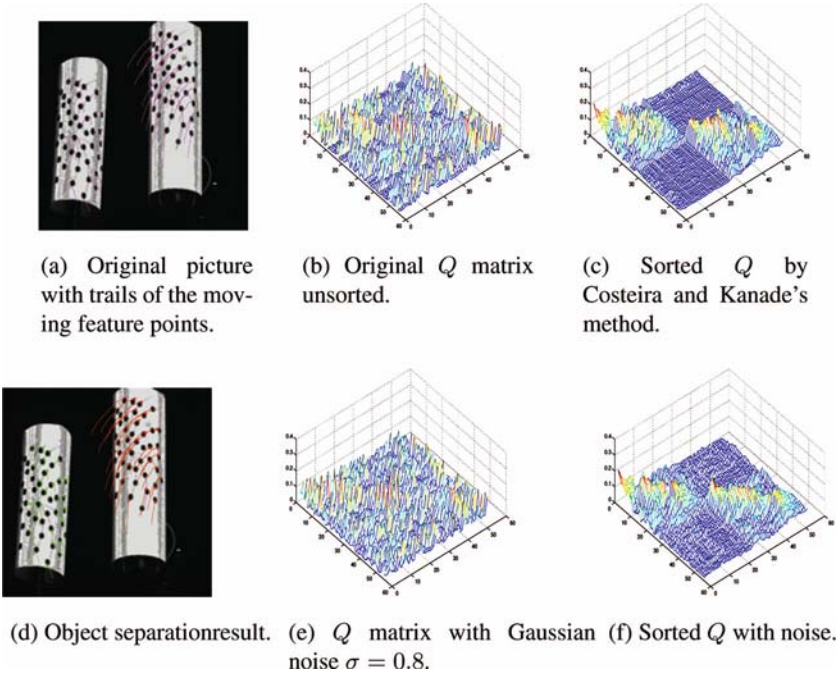
The W matrix can be factorized into a motion matrix M and a shape matrix S thus, $W_{2F \times P} = M_{2F \times r} \times S_{r \times P}$ where r is the rank of W ($r = 4$ in the case of W without noise and outliers). In order to solve the factorization problem, matrix W can be decomposed using SVD by $W = U \Sigma R^T$.

If the features from the same object are grouped together, then U, Σ and R will have a block-diagonal structure as $W = [U_1 \cdots U_N] \begin{bmatrix} \Sigma_1 & & \\ & \ddots & \\ & & \Sigma_N \end{bmatrix} \begin{bmatrix} R_1^T & & \\ & \ddots & \\ & & R_N^T \end{bmatrix}$ and the shape matrix for object k can be approximated by $S_k = B^{-1} \Sigma_k R_k^T$ where B is an invertible matrix that can be found from M.

In a real multi-body tracking problem, the coordinates of the different objects are potentially permuted into a random order. As a result it is impossible to correctly recover the shape matrix S_k without knowledge of the correspondence order. Since the eigenvector matrix V is related to the shape matrix, the shape interaction matrix was introduced by Costeira and Kanade [4] to solve the multi-body separation problem. The shape interaction matrix is

$$Q = RR^T = \begin{bmatrix} S_1^T \Sigma_1^{-1} S_1 & 0 & \cdots & 0 \\ 0 & S_2^T \Sigma_2^{-1} S_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & S_N^T \Sigma_N^{-1} S_N \end{bmatrix} \tag{1}$$

From Equation 1, the shape interaction matrix Q has the convenient properties that $Q(u, v) = 0$, if points u,v belong to different objects and $Q(u, v) \neq 0$, if points u,v belong to the same object. The matrix Q is also invariant to both the object motion and the selection of the object coordinate systems. This leads to a simple scheme for separating multi-object motions by permuting the elements of Q so that it acquires a block diagonal structure. In Costeira and Kanade's method [4] a greedy algorithm is used to permute the Q matrix into block diagonal form. An illustration is shown in Figure 1(a,b,c,d). This method works well only for the ideal case where is no noise and outliers are not present. In Figures 1 e and f we respectively show the effect of adding



(a) Original picture with trails of the moving feature points.

(b) Original Q matrix unsorted.

(c) Sorted Q by Costeira and Kanade's method.

(d) Object separationresult.

(e) Q matrix with Gaussian noise $\sigma = 0.8$.

(f) Sorted Q with noise.

Fig. 1. A multi-body motion separation example using Costeira and Kanade's method

Gaussian noise to the Q matrix in 1(b) and the resulting permuted matrix. In the noisy case, the block structure is badly corrupted and object separation is almost impossible.

3 Robust Object Separation by Commute Time Clustering

In this section, we will show how the multi-body motion tracking problem can be posed as one of commute time embedding using the Q matrix. The method is motivated by the intuition that since the eigenvectors associated with the different objects span different subspaces, they can be embedded using a spectral method and separated using a simple clustering method.

3.1 Graph Laplacian, Heat Kernel, Green's Function and the Commute Time

Commute time is a concept from spectral graph theory that has close links with the graph Laplacian, the heat kernel and random walks on a graph. In the following sections, we show how to compute commute time and describe the relationships to the graph Laplacian and the heat kernel.

Graph Laplacian and Heat kernel. Let the weighted graph Γ be the triple (V, E, Ω), where V is the set of nodes, E is the set of arcs, and $\Omega = \{w_{u,v}, \forall (u,v) \in E\}$ is a set of weights associated with the edges. Further let $T = diag(d_v; v \in V(\Gamma))$ be

the diagonal weighted degree matrix with $T_u = \sum_{v=1}^{n} w_{u,v}$ and A be the adjacency matrix. The un-normalized weighted Laplacian matrix is given by $L = T - A$ and the normalized weighted Laplacian matrix is defined to be $\mathcal{L} = T^{-1/2}LT^{-1/2}$, and has elements

$$\mathcal{L}_\Gamma(u,v) = \begin{cases} 1 & \text{if } u = v \\ -\frac{w_{u,v}}{\sqrt{d_u d_v}} & \text{if } u \neq v \text{ and } (u,v) \in E \\ 0 & \text{otherwise} \end{cases}$$

The spectral decomposition of the normalized Laplacian is $\mathcal{L} = \Phi'\Lambda'\Phi'^T$, where $\Lambda' = diag(\lambda'_1, \lambda'_2, ..., \lambda'_{|V|})$ is the diagonal matrix with the ordered eigenvalues as elements satisfying: $0 = \lambda'_1 \leq \lambda'_2 \ldots \leq \lambda'_{|V|}$ and $\Phi' = (\phi'_1|\phi'_2|....|\phi'_{|V|})$ is the matrix with the ordered eigenvectors as columns. The corresponding eigendecomposition of the un-normalized Laplacian matrix is $L = \Phi\Lambda\Phi^T$.

The heat equation associated with the graph Laplacian is given by $\frac{\partial \mathcal{H}_t}{\partial t} = -\mathcal{L}\mathcal{H}_t$ where $\mathcal{H}_t$ is the heat kernel and t is time. The solution of the heat-equation is found by exponentiating the Laplacian eigenspectrum i.e.$\mathcal{H}_t = \exp[-t\mathcal{L}] = \Phi' \exp[-t\Lambda']\Phi'^T$. The heat kernel is a $|V| \times |V|$ matrix, and for the nodes u and v of the graph Γ the element of the matrix is $\mathcal{H}_t(u,v) = \sum_{i=1}^{|V|} \exp[-\lambda'_i t]\phi'_i(u)\phi'_i(v)$.

Green's function: Now consider the discrete Laplace operator $\Delta = T^{-1/2}\mathcal{L}T^{1/2}$. The Green's function is the left inverse operator of the Laplace operator Δ, defined by $G\Delta(u,v) = I(u,v) - \frac{d_v}{vol}$, where $vol = \sum_{v \in V(\Gamma)} d_v$ is the volume of the graph. A physical interpretation of the Green's function is the temperature at a node in the graph due to a unit heat source applied to the external node. It is related with the heat kernel $\mathcal{H}_t$ in the following manner

$$G(u,v) = \int_0^\infty d_u^{1/2} \left(\mathcal{H}_t(u,v) - \phi'_1(u)\phi'_1(v)\right) d_v^{-1/2} dt \tag{2}$$

Here ϕ'_1 is the eigenvector associated with the zero eigenvalue 0 and which has k-th element is $\phi'_1(k) = \sqrt{d_k/vol}$. Furthermore, the normalized Green's function $\mathcal{G} = T^{-1/2}GT^{1/2}$ is defined as (see [2] page 6(10)),

$$\mathcal{G}(u,v) = \sum_{i=2}^{|V|} \frac{1}{\lambda'_i}\phi'_i(u)\phi'_i(v) \tag{3}$$

where λ' and ϕ' are the eigenvalue and eigenvectors of the normalized Laplacian $\mathcal{L}$. The corresponding un-normalized Green's function $\bar{G} = T^{-1}G = T^{1/2}\mathcal{G}T^{1/2}$ is given by $G(u,v) = \sum_{i=2}^{|V|} \frac{1}{\lambda_i}\phi_i(u)\phi_i(v)$. where λ and ϕ are the eigenvalue and eigenvectors of the un-normalized Laplacian L.

The normalized Green's function is hence the generalized inverse of the normalized Laplacian $\mathcal{L}$. Moreover, it is straightforward to show that $\mathcal{G}\mathcal{L} = \mathcal{L}\mathcal{G} = I - \phi'_1\phi'^T_1$, and as a result $(\mathcal{L}\mathcal{G})(u,v) = \delta(u,v) - \frac{\sqrt{d_u d_v}}{vol}$. From Equation 3, the eigenvalues of $\mathcal{L}$ and $\mathcal{G}$ have the same sign and $\mathcal{L}$ is positive semidefinite, and so $\mathcal{G}$ is also positive semidefinite. Since $\mathcal{G}$ is also symmetric(see [2] page 4), it follows that $\mathcal{G}$ is a kernel. The same applies to the un-normalized Green's function $\bar{G}$.

Commute Time: We note that the *hitting time* $O(u,v)$ of a random walk on a graph is defined as the expected number of steps before node v is visited, commencing from node u. The *commute time* $CT(u,v)$, on the other hand, is the expected time for the random walk to travel from node u to reach node v and then return. As a result $CT(u,v) = O(u,v) + O(v,u)$. The hitting time $O(u,v)$ is given by [2]

$$O(u,v) = \frac{vol}{d_v}G(v,v) - \frac{vol}{d_u}G(u,v)$$

where G is the Green's function given in equation 2. So, the commute time is given by

$$CT(u,v) = O(u,v)+O(v,u) = \frac{vol}{d_u}G(u,u)+\frac{vol}{d_v}G(v,v)-\frac{vol}{d_u}G(u,v)-\frac{vol}{d_v}G(v,u) \quad (4)$$

As a consequence of (4) the commute time is a metric on the graph. The reason for this is that if we take the elements of G as inner products defined in a Euclidean space, CT will become the norm satisfying: $\|x_i - x_j\|^2 =< x_i - x_j, x_i - x_j >=< x_i, x_i > + < x_j, x_j > - < x_i, x_j > - < x_j, x_i >$.

Substituting the spectral expression for the Green's function into the definition of the commute time, it is straightforward to show that

$$CT(u,v) = vol\sum_{i=2}^{|V|}\frac{1}{\lambda_i'}\left(\frac{\phi_i'(u)}{\sqrt{d_u}} - \frac{\phi_i'(v)}{\sqrt{d_v}}\right)^2 \quad (5)$$

In the un-normalized case, it becomes:

$$CT(u,v) = vol\sum_{i=2}^{|V|}\frac{1}{\lambda_i}(\phi_i(u) - \phi_i(v))^2 \quad (6)$$

3.2 Commute Time Embedding

Basics: Equation 5, can be re-written in the following form which makes the relationship between the commute time and the Euclidean distance between the components of the eigenvectors explicit

$$CT(u,v) = \sum_{i=2}^{|V|}\left(\sqrt{\frac{vol}{\lambda_i' d_u}}\phi_i'(u) - \sqrt{\frac{vol}{\lambda_i' d_v}}\phi_i'(v)\right)^2 \quad (7)$$

Hence, the embedding of the nodes of the graph into a vector space that preserves commute time has the co-ordinate matrix

$$\Theta = \sqrt{vol}\Lambda'^{-1/2}\Phi'^T T^{-1/2} \quad (8)$$

The columns of the matrix are vectors of embedding co-ordinates for the nodes of the graph. The term $T^{-1/2}$ arises from the normalisation of the Laplacian. If the commute

time is computed from the un-normalised Laplacian, the corresponding matrix of embedding co-ordinates is

$$\Theta = \sqrt{vol}\Lambda^{-1/2}\Phi^T \tag{9}$$

The embedding is nonlinear in the eigenvalues of the Laplacian. This distinguishes it from principle components analysis (PCA) and locality preserving projection (LPP) which are both linear. As we will demonstrate in the next section, the commute time embedding is just kernel PCA [15] on the Green's function. Moreover, it can be viewed as Laplacian eigenmap since they actually are minimizing the same objective function.

The commute time embedding and Kernel PCA: Let us consider the un-normalized case above. Since the Green's function $\bar{G}$ is the pseudo-inverse of the Laplacian, it discards the zero eigenvalue and the corresponding eigenvector $\mathbf{1}$ of the Laplacian. The columns of the eigenvector matrix are orthogonal which means the eigenvector matrix Φ of $\bar{G}$ satisfies $\Phi^T\mathbf{1} = \mathbf{0}$. Hence, $\sqrt{vol}\Lambda^{-1/2}\Phi^T\mathbf{1} = \mathbf{0}$, and this means that the data is centred. As a result, the covariance matrix for the centred data is

$$C_f = \Theta\Theta^T = vol\Lambda^{-1/2}\Phi^T\Phi\Lambda^{-1/2} = vol\Lambda^{-1} = \Lambda_{\bar{G}} \tag{10}$$

and the kernel or Gram matrix is

$$K = \Theta^T\Theta = vol\Phi\Lambda^{-1/2}\Lambda^{-1/2}\Phi^T = vol\Phi\Lambda^{-1}\Phi^T = vol\bar{G} \tag{11}$$

which is just the Green's function multiplied by a constant. Hence, we can view the embedding as performing kernel PCA on the Green's function for the Laplacian.

The commute time embedding and the Laplacian eigenmap: In the Laplacian eigenmap [1] the aim is to embed a set of points with co-ordinate matrix $\bar{\mathbf{X}} = \{\bar{\mathbf{x}}_1, \bar{\mathbf{x}}_2, ..., \bar{\mathbf{x}}_n\}$ from a R^n space into a lower dimensional subspace R^m with the co-ordinate matrix $\mathbf{Z} = \{\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_m\}$. The original data-points have a proximity weight matrix Ω with elements $\Omega_{u,v} = \exp[-||\bar{\mathbf{x}}_u - \bar{\mathbf{x}}_v||^2]$. The aim is to find the embedding that minimises the objective function $\epsilon = \sum_{u,v} \|\mathbf{z}_u - \mathbf{z}_v\|^2 \Omega(u,v) = tr(\mathbf{Z}^T L\mathbf{Z})$ where Ω is the edge weight matrix of the original data $\bar{\mathbf{X}}$.

To remove the arbitrary scaling factor and to avoid the embedding undergoing dimensionality collapse, the constraint $\mathbf{Z}^T T\mathbf{Z} = I$ is applied. The embedding problem becomes $\mathbf{Z} = \arg\min_{\mathbf{Z}^T T\mathbf{Z}=I} tr(\mathbf{Z}^T L\mathbf{Z})$.

The solution is given by the lowest eigenvectors of the generalized eigen-problem

$$L\mathbf{Z} = \Lambda' T\mathbf{Z} \tag{12}$$

and the value of the objective function corresponding to the solution is $\epsilon^* = tr(\Lambda')$.

For the commute-time embedding the objective function minimised is

$$\epsilon' = \frac{\sum_{u,v} \|\mathbf{z}_u - \mathbf{z}_v\|^2 \Omega(u,v)}{\sum_u \mathbf{z}_u^2 d_u} = tr(\frac{\mathbf{Z}^T L\mathbf{Z}}{\mathbf{Z}^T T\mathbf{Z}})$$

To show this, let $\mathbf{Z} = Y^T = (\sqrt{vol}\Lambda'^{-1/2}\Phi'^T T^{-1/2})^T$, we have

$$\begin{aligned}\epsilon' &= tr(\frac{\sqrt{vol}\Lambda'^{-1/2}\Phi'^T T^{-1/2} L T^{-1/2}\Phi'\Lambda'^{-1/2}\sqrt{vol}}{\sqrt{vol}\Lambda'^{-1/2}\Phi'^T T^{-1/2} T T^{-1/2}\Phi'\Lambda'^{-1/2}\sqrt{vol}}) \\ &= tr(\frac{\Lambda'^{-1/2}\Phi'^T\mathcal{L}\Phi'\Lambda'^{-1/2}}{\Lambda'^{-1/2}\Phi'^T\Phi'\Lambda'^{-1/2}}) = tr(\frac{\Lambda'^{-1/2}\Lambda'\Lambda'^{-1/2}}{\Lambda'^{-1}}) = tr(\Lambda') = \epsilon^*\end{aligned}$$

Hence, the commute time embedding not only aims to maintain proximity relationships by minimizing $\sum_{u,v}\left\|\mathbf{z}_u - \mathbf{z}_v\right\|^2 \Omega(u,v)$, but it also aims to assign large co-ordinates values to nodes (or points) with large degree (i.e. it maximizes $\sum_u \mathbf{z}_u^2 d_u$). Nodes with large degrees are the most significant in a graph since they have the largest number of connecting edges. In the commute time embedding, these nodes are furthest away from the origin and are hence unlikely to be close to one-another.

The commute time and the diffusion map: Finally, it is interesting to note the relationship with the diffusion map embedding of Lafon *et al* [3]. The method commences from the random walk on a graph which has transition probability matrix $P = T^{-1}A$, where A is the adjacency matrix. Although P is not symmetric, it does have a right eigenvector matrix Ψ, which satisfies the equation $P\Psi = \Lambda_P\Psi$.

Since $P = T^{-1}A = T^{-1}(T - L) = I - T^{-1}L$ and as result $(I - T^{-1}L)\Psi = \Lambda_P\Psi$,i.e. $T^{-1}L\Psi = (I - \Lambda_P)\Psi$, and as result $L\Psi = (I - \Lambda_P)T\Psi$, which is identical to Equation 12 if $\mathbf{Z} = \Psi$ and $\Lambda' = I - q\Lambda_P$. The embedding co-ordinate matrix for the diffusion map is $Y = \Lambda^t\Psi^T$, where t is real. For the embedding the diffusion distance between a pair of nodes is $D_t^2(u,v) = \sum_{i=1}^m (\lambda_P)_i^{2t}\left(\psi_i(u) - \psi_i(v)\right)^2$. Clearly if we take $t = -1/2$ the diffusion map is equivalent to the commute time embedding and the diffusion time is equal to the commute time.

The diffusion map is designed to give a distance function that reflects the connectivity of the original graph or point-set. The distance should be small if a pair of points are connected by many short paths, and this is also the behaviour of the commute time. The advantage of the diffusion map or distance is that it has a free parameter t, and this may be varied to alter the properties of the map. The disadvantage is that when t is small, the diffusion distance is ill-posed. The reason for this is that according to the original definition of the diffusion distance for a random walk ($D_t^2(u,v) = \left\|p_t(u,\cdot) - p_t(v,\cdot)\right\|^2$), and as a result the distance between a pair of nodes depends on the transition probability between the nodes under consideration and all of the remaining nodes in the graph. As a result if t is small, then the random walk will not have propagated significantly, and the distance will depend only on very local information. There are also problems when t is large. When this is the case the random walk converges to its stationary state with $P^t = T/vol$ (a diagonal matrix), and this gives zero diffusion distance for all pairs of distinct nodes. So it is a critical to control t carefully in order to obtain useful embeddings.

Some embedding examples: [Figure 2 shows four synthetic examples of point- configurations (left-hand panel) and the resulting commute time embeddings (right-hand panel). Here we have computed the proximity weight matrix Ω by exponentiating the Euclidean distance between points. The main features to note are as follows. First, the embedded points corresponding to the same point-clusters are cohesive, being scattered

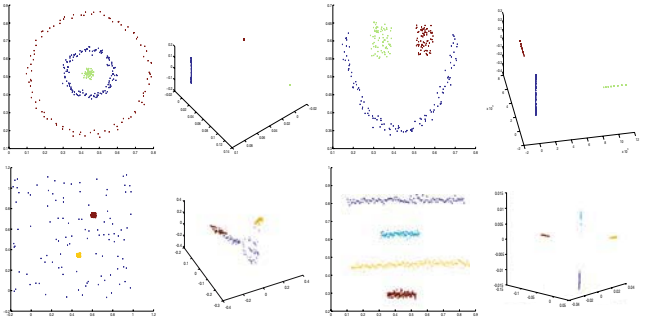


Fig. 2. Commute time embedding examples

around approximately straight lines in the subspace. Second, the clusters corresponding to different objects give rise to straight lines that are orthogonal.

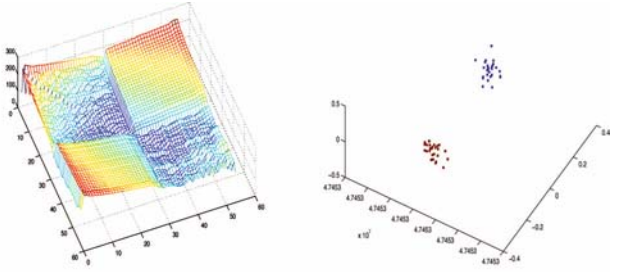
Robustness of the commute time embedding: From Equation 9 we can see that the co-ordinates of the commute time embedding depend on the eigenvalues and eigenvectors of the Laplacian matrix. Hence, the stability of the embedding depends on the stability of the eigenvalue and eigenvector matrices. According to Weyl's theorem, the variation of the eigenvalues of a perturbed matrix is bounded by the maximum and the minimum eigenvalues of the perturbing matrix. However, the eigenvectors are less stable under perturbation. Despite this anticipated problem, the commute time matrix is likely to be relatively stable under perturbations in graph structure. According to Rayleigh's Principle in the theory of electrical networks, commute time can neither be increased by adding an edge or a node, nor decreased by deleting a single edge or a node. In fact, the impact of deleting or adding an edge or a node to the commute time between a pair of nodes is negligible if they are well connected. This property reduces the impact of outliers in motion tracking, since outliers are dissimilar to the object point-clusters.

3.3 Commute Times Applied to the Multi-body Motion Tracking Problem

Having discussed some of the properties of the commute time embedding, in this section we return to the issue of how it may be used for multi-body motion analysis. As we have already seen, the shape interaction matrix Q introduced in the factorization method is invariably contaminated by noise and this limits its effectiveness. Our aim is to use commute time as a shape separation measure. Specifically, we use the commute time to refine the block structure of the Q matrix and group the feature points into objects.

Object Separation Steps: The algorithm we propose for this purpose has the following steps:

1. Use the shape interaction matrix Q as the weighted adjacency matrix A and construct the corresponding graph Γ.
2. Compute the Laplacian matrix of graph Γ using $L = T - Q$.



(a) Sorted commute time matrix.

(b) Clustered points in the commute time subspace for two objects.

Fig. 3. Multi-body motion separation re-casted as a commute time clustering problem

3. Find the eigenvalue matrix Λ and eigenvector matrix Φ of L using $L = \Phi\Lambda\Phi^T$.
4. Compute the commute time matrix CT using Λ and Φ from Equation 6.
5. Embed the commute time into a subspace of R^n using Equation 8 or 9.
6. Cluster the data points in the subspace using the k-means algorithm [11].

To illustrate the effectiveness of this method, we return to example used earlier in Section 2. First, in the ideal case, the Q matrix will have a zero value for the feature points belonging to different objects. As a result the graph Γ, constructed from Q, will have disjoint subgraphs corresponding to the nodes belonging to different objects. The partitions give rise to infinite commute times, and are hence unreachable by the random walk. However, when we add noise (Q with 0.8 Gaussian noise) and the clustering steps listed above we still recover a good set of objects (see Figure 1(d)). This is illustrated in Figure 3. Here, in Figure 3 sub-figure (a) shows the commute time matrix of graph Γ and sub-figure (b) shows the embedding in a 3D subspace. It is clear that the commute time matrix gives a good block-diagonal structure and the points are well clustered in the embedding space even when significant noise is present.

4 Experiments

In this section we conduct experiments with the commute time method on both synthetic data and real-world motion tracking problems. To investigate the robustness of the method, we add Gaussian noise to the data sets and compare the results with some classical methods.

4.1 Synthetic Data

Figure 4 shows a sequence of five consecutive synthetic images with 20 background points(green dots) and 20 foreground points(red dots) moving independently. We have added Gaussian noise of zero mean and standard deviation σ to the coordinates of these 29 points, and then cluster them into two groups.

We have compared our method with Costeira and Kanade's greedy algorithm [4], Ichimura's discrimination criterion method [9] and Kenichi's subspace separation

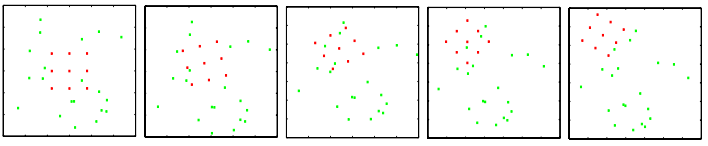
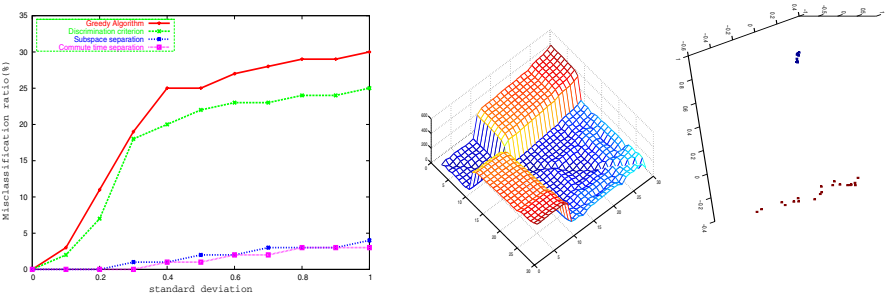


Fig. 4. Synthetic image sequence



(a) Method comparison. (b) Sorted commute time matrix. (c) Embedded subspace.

Fig. 5. Synthetic data

method [10]. In Figure 5 we plot the average misclassification ratio over an increasing σ on the different algorithms. The results are based on an average of 50 trials for each method. From the figure, it is clear that our method performs significantly better than the greedy method and the discrimination criterion method. It also has a margin of advantage over the subspace separation method.

For an example with a Gaussian noise with $\sigma = 0.5$, the commute time matrix and the embedded subspace are shown in Figure 5(b) and 5(c) respectively. It is clear that even in the noise contaminated case, the commute time matrix still maintains a good block-diagonal structure. Moreover, under the embedding the points are easily separated.

4.2 Real-World Motion Tracking

In this section we experiment with the commute time method on real-world multi-body motion tracking problems. Figure 6 shows five real video sequences with the successfully tracked feature points using the commute time method. The full sequences can be found in the supplementary material web-site.

The first three rows are for the data used by Sugaya and Kanatani in [19, 18]. Here there is one moving object and a moving camera. A successful tracking method will separate the moving object from the moving background. The forth and fifth rows in Figure 6 are two video sequences captured using a Fuji-Film 2.0M camera(320×240 pixels). For each of sequence, we detected feature points using the KLT [17], and tracked the feature points using the commute time method. Due to the continuous loss of the feature points in the successive frames by the KLT algorithm, we use only ten frames each

from the sequences with 117 and 116 feature points respectively. Compared to the data from Sugaya and Kanatani, we increase the number of detected moving objects from one to two, which makes the separation more difficult.

In the case of the forth row of Figure 6, our method not only separates the ducks correctly from the moving background, but it also separates the moving ducks from each other. The fifth row of Figure 6 is the most difficult one with two independently moving hands and a moving background. it also separates the wall from the floor correctly.

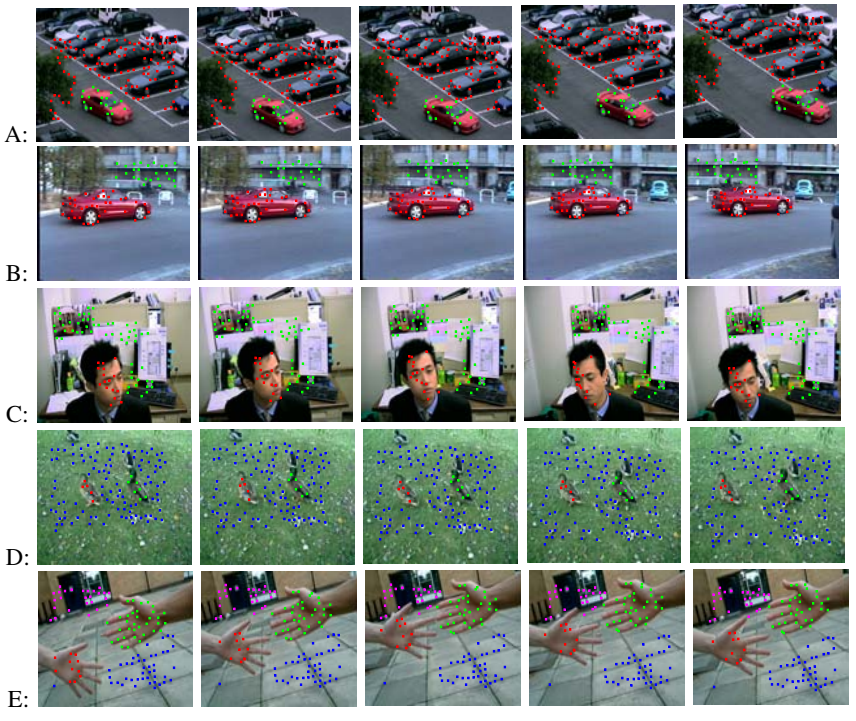


Fig. 6. Real-world video sequences and successfully tracked feature points

For the same sequences, we compared our results with Costeira and Kanade's greedy algorithm, Ichimura's discrimination criterion method, Kanatani's subspace separation method and Sugaya and Kanatani's multi-stage learning method. The comparison is shown in Table 1.

Table 1 lists the accuracies of different methods measured by the number of correctly classified points over the total number of points in percentage. The percentage is averaged over 50 trails for each method. From the table, it is clear that the greedy algorithm gives the worst results. The discrimination criterion method and the subspace separation method perform better due to their robustance to the noise. The multi-stage learning method delivers significantly better results due to its adaptive capabilities, but failed on our data. The failures are most pronounced when there are several moving ob-

Table 1. Separation accuracy for the sequences in Fig. 6

	A	B	C	D	E
Costeira-Kanade	60.3	71.3	58.8	45.5	30.0
Ichimura	92.6	80.1	68.3	55.4	47.2
Subspace Separation	59.3	99.5	98.9	80.6	67.2
Multi-stage Learning	100.0	100.0	100.0	93.7	81.5
Commute Time Separation	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**

jects and an inconsistent moving background. Our method gives the best performance and achieves 100% accuracy.

5 Conclusion

In this paper, we have described how the multi-body motion tracking problem can be cast into a graph spectral setting using a commute time embedding method together with k-means clustering. The commute time is conveniently computed using the Laplacian eigensystem. We have shown how the commute time embedding is linked to kernel PCA, the Laplacian eigenmap and the diffusion map. We have compared our embedding method with a number of alternative tracking algorithms on both synthetic and real world data. Here it offers a convincing margin of improvement for noise-contaminated multi-body motion tracking.

Acknowledgements. The authors would like to thank João Costeira and Jared Jacobs for generously providing their data and code for this work.

References

1. M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.
2. F.R.K. Chung and S.-T. Yau. Discrete green's functions. In *J. Combin. Theory Ser.*, pages 191–214, 2000.
3. R.R. Coifman, S. Lafon, A.B. Lee, M. Maggioni, B. Nadler, F. Warner, and S.W. Zucker. Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. *National Academy of Sciences*, 102(21):7426–7431, 2005.
4. J. Costeira and T. Kanade. A multibody factorization method for independently moving objects. *IJCV*, 29(3):159 – 179, 1997.
5. Z. Fan, J. Zhou, and Y. Wu. Inference of multiple subspaces from high-dimensional data and application to multibody grouping. In *CVPR*, pages 661–666, 2004.
6. Z. Fan, J. Zhou, and Y. Wu. Multibody motion segmentation based on simulated annealing. In *CVPR*, pages 776–781, 2004.
7. C.W. Gear. Multibody grouping from motion images. *IJCV*, 29(2):130–150, 1998.
8. A. Gruber and Y. Weiss. Multibody factorization with uncertainty and missing data using the em algorithm. In *CVPR*, pages 707–714, 2004.
9. N. Ichimura. Motion segmentation based on factorization method and discriminant criterion. In *ICCV*, pages 600–605, 1999.

10. K. Kanatani. Motion segmentation by subspace separation and model selection. In *ICCV*, pages 301–306, 2001.
11. J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, pages 281–297, 1967.
12. J. Park, H. Zha, and R. Kasturi. Spectral clustering for robust motion segmentation. In *ECCV*, pages 390–401, 2004.
13. M. Pavan and M. Pelillo. A new graph-theoretic approach to clustering and segmentation. In *CVPR03*, pages I: 145–152, 2003.
14. H. Qiu and E.R. Hancock. Image segmentation using commute times. In *BMVC*, pages 929–938, 2005.
15. B. Sch, A. Smola, and K. Muller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.
16. J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE PAMI*, 22(8):888–905, 2000.
17. J. Shi and C. Tomasi. Good features to track. In *CVPR*, pages 593–600, 1994.
18. Y. Sugaya and K. Kanatani. Outlier removal for motion tracking by subspace separation. *IEICE Trans. INF and SYST*, E86-D(6):1095–1102, 2003.
19. Y. Sugaya and K. Kanatani. Multi-stage unsupervised learning for multi-body motion segmentation. *IEICE Trans. INF and SYST*, E87-D(7):1935–1942, 2004.
20. Y. Weiss. Segmentatoin using eigenvectors: a unifying view. In *ICCV.*, pages 975–982, 1999.
21. Y. Wu, Z. Zhang, T. S. Huang, and J. Y. Lin. Multibody grouping via orthogonal subspace decomposition. In *CVPR*, pages 252–257, 2001.
22. L. Zelnik-Manor and M. Irani. Degeneracies, dependencies and their implications in multi-body and multi-sequence factorizations. In *CVPR*, pages 287–293, 2003.