**Manifold Kernel Sparse Representation of Symmetric Positive Definite Matrices and Its Applications**

<table>
<thead>
<tr>
<th>Journal:</th>
<th><em>Transactions on Image Processing</em></th>
</tr>
</thead>
<tbody>
<tr>
<td>Manuscript ID:</td>
<td>TIP-12649-2014</td>
</tr>
<tr>
<td>Manuscript Type:</td>
<td>Regular Paper</td>
</tr>
<tr>
<td>Date Submitted by the Author:</td>
<td>02-Sep-2014</td>
</tr>
<tr>
<td>Complete List of Authors:</td>
<td>Wu, Yuwei; Beijing Inst. of Tech., School of computer science Jia, Yunde; Beijing Institute of Technology, School of Computer Science Li, Peihua; Dalian University of Technology, School of Information and Communication Engineering Zhang, Jian; University of Technology, Sydney, Advance Analytics Institute Yuan, Junsong; Nanyang Technological University, EEE</td>
</tr>
</tbody>
</table>
Manifold Kernel Sparse Representation of Symmetric Positive Definite Matrices and Its Applications

Yuwei Wu, Yunde Jia Member, IEEE, Peihua Li, Jian Zhang Senior Member, IEEE, and Junsong Yuan Member, IEEE

Abstract—Symmetric positive definite (SPD) matrices, as a connected Riemannian manifold, have become increasingly popular to encode image information. Most existing sparse models are still primarily developed in the Euclidean space. They do not consider the non-linear geometrical structure of the data space, hence are not directly applicable to the Riemannian manifold. In this paper, we propose a novel sparse representation method of SPD matrices in the data-dependent manifold kernel space. The graph Laplacian as a smooth operator of the manifold is incorporated into the kernel space to better reflect the underlying geometry of SPD matrices. We also introduce two different positive definite kernel functions which can be easily transformed to the corresponding manifold kernels. The obtained sparse representation varies smoothly along the geodesics of the data manifold and has more discriminating power. Extensive experimental results demonstrate good performance of manifold kernel sparse codes in image classification, face recognition, and visual tracking.


I. INTRODUCTION

Sparse representation (SR) has been an important subject in signal processing and computer vision community with a wide range of applications including visual tracking [1], [2], [3], face recognition [4], [5], [6], and image classification [7], [8], [9]. Given a set of data points \( \mathcal{X} = \{x_1, x_2, \ldots, x_n\} \), the sparse model attempts to find a dictionary \( \mathcal{D} = \{d_1, d_2, \ldots, d_N\} \), where \( d_i \) is so-called basis or atom, such that each \( x_i \) can be linearly reconstructed by a relatively small subset of atoms from \( \mathcal{D} \), meanwhile keeping the reconstruction error as small as possible. The underlying linear process significantly depends on the assumption that the data points and the atoms lie on a vector space \( \mathbb{R}^d \). In many applications, however, data points actually belong to some known Riemannian manifolds such as the space of symmetric positive-definite (SPD) matrices [10], [11], Stiefel and Grassmann manifolds [12]. Most existing sparse models in \( \mathbb{R}^d \) fail to consider the non-linear geometrical structure of the manifold space \( \mathcal{M} \), and hence are not directly applicable to the Riemannian manifold. In this paper, we tackle the problem of the SR in the space of \( d \times d \) SPD matrices, denoted by \( \text{Sym}_d^+ \). To formulate the sparse representation on \( \text{Sym}_d^+ \), one should consider two issues: (1) Unlike the Euclidean space, the Riemannian manifold \( \text{Sym}_d^+ \) has no the global linear structure which allows the SPD matrix to be reconstructed linearly by the atoms in \( \mathcal{D} \). (2) \( \ell_2 \)-norm is inappropriate to measure the intrinsic distance between two SPD matrices.

A direct approach is to seek proper linear decomposition and reconstruction error measures of SPD matrices. Sivalingam et al. [13] proposed a tensor sparse coding method, in which the Logdet divergence is used to measure the reconstruction error. Then the sparse decomposition of an SPD matrix is formulated as a MAXDET optimization problem that can be solved by the interior-point (IP) algorithm. Sivalingam et al. [14] further introduced a dictionary learning method using the Logdet divergence. However, the solutions of the above two approaches are computationally expensive. Sra and Cherian [15] adopted the Frobenius norm as the error metric to learn a generalized dictionary of rank-1 atoms to sparsely represent the SPD matrix.

An alternative method is to embed the manifold-valued data into the vector space \( \mathbb{R}^d \) in order to apply the existing sparsity modeling methods. One commonly used vector space is the tangent space at the mean of the data points in \( \mathcal{M} \). The logarithmic and exponential maps are iteratively used to map the manifold-valued data points to the tangent space, and vice-versa. Exploiting the Log-Euclidean mapping to SPD matrices, Zhang et al. [16] obtained the vectorized Log-Euclidean covariance features for sparse representation. Guo et al. [17] transformed the Riemannian manifold of SPD matrices into the vector space \( \mathbb{R}^d \) under the matrix logarithm mapping. The log-covariance matrix is approximated by a sparse linear combination of the log-covariance matrices of training samples. Yuan et al. [18] also proposed to solve sparse representation for human action recognition by embedding manifolds into tangent spaces. While log-Euclidean approaches benefit from its simplicity, the iterative computation of the logarithmic and exponential maps demands a high computational cost. In addition, the tangent space preserves only the local structure of the manifold, i.e., the true geometry of the manifold is not taken into account, which often results in sub-optimal
performance.

To consider the local manifold structure of the manifold-valued data, many attempts have been made to implicitly map these data into a high-dimensional Reproducing Kernel Hilbert Space (RKHS) by using the nonlinear mapping associated with a kernel function. Harandi et al. [19] tackled the problem of both SR and dictionary learning in $\text{Sym}^+_d$ by adopting the Stein kernel to map the SPD matrices to RKHS. The Stein divergence is only an approximation of Riemannian metric, and is positive definite only for some values of the Gaussian bandwidth parameter. Barachant et al. [20] exploited a Riemannian-based kernel to model the SR of the SPD matrices for brain-computer interface applications. Li et al. [21] also embedded $\text{Sym}^+_d$ into RKHS and developed Log-E kernels for SR and dictionary learning of SPD matrices based on the Log-Euclidean framework. Although Log-E kernels obtained the satisfactory results in face recognition and image classification, their modeling does not explicitly reflect the geometrical structure of the data space.

The key issue of mapping SPD matrices into RKHS while preserving the geometrical structure of the data is the construction of the kernel function. An essential criterion is that the kernel function should be positive definite. The Gaussian kernel is perhaps the most popular positive definite kernels on $\mathbb{R}^d$. Both Jayasumana et al. [11] and Vemulapalli et al. [22] presented the Gaussian kernel based on the Log-Euclidean metric. In practice, however, the nonlinear structure captured by the data-independent kernels, e.g., Gaussian kernel, may not be consistent with the intrinsic manifold structure.

In this paper, we construct a data-dependent manifold kernel function using the kernel deformation principle [23]. The SR on the space of SPD matrices can be performed by embedding the $\text{Sym}^+_d$ into a RKHS using the manifold kernel, as shown in Fig. 1. In RKHS, the input data $\phi(x_i)$ can be approximated by using a sparse linear combination of atoms $\phi(d_i)$ from the dictionary. The graph Laplacian as a smooth operator of the manifold-valued data is incorporated into the kernel space to discover the manifold structure. Different positive definite kernel functions on the space of SPD matrices are introduced, which can be easily transformed to the corresponding manifold kernels to better characterize the underlying geometry of the manifold.

The remainder of this paper is organized as follows. We discuss the preliminaries including Riemannian geometry on SPD matrices and kernel sparse representation in Sect. II. In Sect. III, we introduce the data-dependent manifold kernel on SPD matrices. Then we describe the details of the manifold kernel sparse representation on $\text{Sym}^+_d$, including its objective function and its implementation in Sect. IV. Experimental results are reported and analyzed in Sect. V and the conclusion is given in Sect. VI.

II. PRELIMINARIES

A. Riemannian Geometry on SPD Matrices

SPD matrices usually emerge in the form of covariance matrices defined in Definition 1 [24]. The covariance matrix descriptor, as a special case of SPD matrices, can capture feature correlations compactly in an object region, and therefore has been proven to be effective for pedestrian detection [25], face recognition [26], texture classification [27], and visual tracking [28].

Definition 1: Given a region of interest of an image, let $z_i \in \mathbb{R}^d$, for $i = 1, 2, \cdots, N$, be feature vectors from the region denoted by $R$, then the covariance matrix descriptor of the region $C_R \in \text{Sym}^+_d$ is defined as

$$C_R = \frac{1}{N-1} \sum_{i=1}^{N} (z_i - \mu_R)(z_i - \mu_R)\top,$$

where $\mu_R = \frac{1}{N} \sum_{i=1}^{N} z_i$ is the mean vector, and $N$ is the number of pixels in region $R$. The feature vector $z_i$ may consist of the pixel coordinates, image gray level or color, image gradients, edge magnitude, edge orientation, filter responses, etc. For example, $z = [x, y, I, |I_x|, |I_y|, \sqrt{I_x^2 + I_y^2}]\top$.

In $\text{Sym}^+_d$, SPD matrix lies on a connected Riemannian manifold. In this case, the geodesic distance induced by the Riemannian metric is a suitable choice to consider the manifold structure of the SPD matrices. Two most widely used distance measures in $\text{Sym}^+_d$ are the affine-invariant distance and the Log-Euclidean distance [24]. Typically, the former requires eigenvalue computations, which causes significant slowdowns for the larger matrices. The latter is particularly simple to use and overcomes the computational limitations of the affine-invariant distance.

For any matrices $C_1$ and $C_2$ in $\text{Sym}^+_d$, the logarithmic product $C_1 \circ C_2$ is defined as

$$C_1 \circ C_2 := \exp(\log(C_1) + \log(C_2)).$$

The logarithmic multiplication $\circ$ on $\text{Sym}^+_d$ is compatible with the structure of smooth manifold: $(C_1, C_2) \mapsto C_1 \circ C_2 \in C^\infty \text{ Sym}^+_d$, therefore, is given a commutative Lie group structure $\mathcal{G}$ by $\circ$. The tangent space at the identity element in $\mathcal{G}$ forms a Lie algebra $\mathcal{H}$, a vector space. In a Lie algebra $\mathcal{H}$, the Riemannian manifold of SPD matrices can be mapped to the Euclidean space by matrix logarithm. Analogously, the results of the Euclidean space can be mapped back to the Riemannian space by the matrix exponential. Given a symmetric matrix $C \in \text{Sym}^+_d$, $C = U \Sigma U\top$ is the eigen-decomposition of SPD matrix $C$, where $U$ is an orthonormal matrix and $\Sigma = \text{Diag}(\lambda_1, \lambda_2, \cdots, \lambda_n)$ is a diagonal matrix composed of the eigenvalues. SPD matrix $C$ has a unique matrix logarithm $\log(C)$ and matrix exponential $\exp(C)$:

$$\begin{align*}
\log(C) &= U \cdot \text{Diag}(\log(\lambda_1), \log(\lambda_2), \cdots, \log(\lambda_d)) \cdot U\top, \\
\exp(C) &= U \cdot \text{Diag}(\exp(\lambda_1), \exp(\lambda_2), \cdots, \exp(\lambda_d)) \cdot U\top.
\end{align*}$$

Fig. 1. Data points $x_i$ on the manifold $\mathcal{M}$ of SPD matrices are mapped into RKHS using the data-dependent manifold kernel function. In RKHS, $\phi(x_i)$ can be represented by a linear combination of atoms $\phi(d_i)$ from the dictionary $\mathcal{D}$. 


Log-Euclidean metric, bi-invariant metric on the lie group of SPD matrices, corresponds to a Euclidean metric in the logarithmic domain. The distance between two matrices \( C_1 \) and \( C_2 \) is calculated by
\[
d(C_1, C_2) = \| \log(C_1) - \log(C_2) \|_F,
\]
where \( \| \cdot \|_F \) denotes the matrix Frobenius norm induced by the Frobenius matrix inner product \( \langle \cdot, \cdot \rangle \).

**B. Kernel sparse representation**

Let \( \mathcal{X} = \{x_1, x_2, \ldots, x_n\} \in \mathbb{R}^{d \times n} \) be a data matrix with \( n \) \( d \)-dimensional features extracted from an image, \( \mathcal{D} = \{d_1, d_2, \ldots, d_N\} \in \mathbb{R}^{d \times N} \) be the dictionary where each column represents an atom, and \( \alpha = [\alpha_1, \alpha_2, \ldots, \alpha_n] \in \mathbb{R}^{N \times n} \) be the coding matrix. The goal of sparse representation is to learn a dictionary and corresponding sparse codes such that each input local feature \( x_i \) can be well approximated by the dictionary \( \mathcal{D} \). The general formulation of the sparse representation is expressed as
\[
\min_{\mathcal{D}, \alpha} \sum_{i=1}^{n} \| x_i - \mathcal{D}\alpha_i \|_2^2 + \| \alpha_i \|_1
\]
subject to \( \| d_i \|_2 \leq 1 \), where \( \| x_i - \mathcal{D}\alpha_i \|_2^2 \) measures the approximation error, and \( \| \alpha_i \|_1 \) enforces \( \alpha_i \) to have a small number of nonzero elements. Although the objective function in Eq. (5) is not convex in both variables, it is convex in either \( \mathcal{D} \) or \( \alpha \) the \( \ell_1 \) minimization problem can be solved efficiently with the SPAMS package [29].

Gao et al. [30] proposed a kernel version of sparse representation in the RKHS mapped by an implicit mapping function \( \phi \). Mercer kernels are usually employed to carry out the mapping implicitly. The Mercer kernel is a function \( \mathcal{K}(\cdot, \cdot) \) which can generate a kernel matrix \( \mathcal{K}_{ij} = \mathcal{K}(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle \) using pairwise inner products between mapped samples for all the input data points. The data points \( \mathcal{X} \) and dictionary \( \mathcal{D} \) are transformed to the corresponding feature space:
\[
\mathcal{X} = \{x_1, x_2, \ldots, x_n\} \xrightarrow{\phi} \{\phi(x_1), \phi(x_2), \ldots, \phi(x_n)\}
\]
\( \mathcal{D} = \{d_1, d_2, \ldots, d_N\} \xrightarrow{\phi} \{\phi(d_1), \phi(d_2), \ldots, \phi(d_N)\} \)

Then we substitute the mapped features and dictionary to the kernelized formulation of sparse representation:
\[
\min_{\mathcal{D}, \alpha} \sum_{i=1}^{n} \| \phi(x_i) - \phi(\mathcal{D})\alpha_i \|_2^2 + \| \alpha_i \|_1
\]
subject to \( \mathcal{K}(d_i, d_j) \leq 1, \mathcal{K}^{\top} \alpha_i = 1 \).

While the kernel sparse representation has been extensively developed, most algorithms [30], [31], [32], [9] are still primarily developed for data points lying in vector space. In this work, we focus on the sparse representation of SPD matrices, \( \text{Sym}^+_d \), which forms a Riemannian manifold endowed with an appropriate metric. Motivated by the nonlinear generalization performance of kernel methods for sparse representation [30], [19], [21], we embed \( \text{Sym}^+_d \) into RKHS using the data-dependent manifold kernel which can reflect the underlying geometry of the data.

**III. DATA-DEPENDENT MANIFOLD KERNEL ON \( \text{Sym}^+_d \)**

A. Data-dependent Kernel Function

The key issue of mapping SPD matrices into the RKHS while preserving the geometrical structure of the data is the choice of the kernel function. We adopt a kernel deformation principle [23] to learn a data-dependent kernel function.

Let \( \mathcal{H} \) denote the original RKHS reproduced by the kernel function \( \mathcal{K}(x_i, x_j) \), and \( \mathcal{H} \) denote the deformed RKHS. We assume the following relationship between the two Hilbert spaces:
\[
\langle f, g \rangle_{\mathcal{H}} = \langle f, g \rangle_{\mathcal{H}} + f^\top M g,
\]
where \( f = (f(x_1), \ldots, f(x_n)) \) and \( g = (g(x_1), \ldots, g(x_n)) \) are vectors. \( M \) is a symmetric positive semi-definite matrix that captures the geometry relationship among all the data points. Eq. (8) combines the original ambient smoothness with an intrinsic smoothness measure defined in the deformation term \( f^\top M g \). With the modified data-dependent inner product, \( \mathcal{H} \) becomes better suited compared with \( \mathcal{H} \), and a kernel function \( \mathcal{K}(x_i, x_j) \) associated with \( \mathcal{H} \) is given by
\[
\tilde{\mathcal{K}}(x_i, x_j) = \mathcal{K}(x_i, x_j) - \mu k_{x_i} (I + MK)^{-1} M k_{x_j},
\]
where \( I \) is an identity matrix, \( \mathcal{K} = [\mathcal{K}(x_i, x_j)]_{n \times n} \) is the original kernel matrix in \( \mathcal{H} \), \( k_{x_i} \) and \( k_{x_j} \) denote the column vectors \( k_{x_i} = [\mathcal{K}(x_i, x_1), \ldots, \mathcal{K}(x_i, x_n)]^\top \in \mathbb{R}^{n \times 1} \) and \( k_{x_j} = [\mathcal{K}(x_j, x_1), \ldots, \mathcal{K}(x_j, x_n)]^\top \in \mathbb{R}^{n \times 1} \), respectively. \( \mu \geq 0 \) is the kernel deformation parameter controlling the smoothness of the functions.

Preserving the geometrical structure of the data largely depends on \( M \). The spectral graph theory [9] has indicated that the geometrical structure can be approximated by the graph Laplacian associated to the data points. Consider a graph with \( n \) vertices where each vertex corresponds to a data point \( x_i \in \text{Sym}^+_d \). Define the edge weight matrix \( W \in \mathbb{R}^{n \times n} \) as
\[
W_{ij} = \begin{cases} 1, & \text{if } x_i \in N_\epsilon(x_j) \text{ or } x_j \in N_\epsilon(x_i) \\ 0, & \text{otherwise,} \end{cases}
\]
where \( N_\epsilon(x_j) \) represents the set of \( \epsilon \) nearest neighbors of \( x_j \), which can be effectively computed by Log-Euclidean distance defined in Eq. (4). Let \( L = D - W \), where \( D \) is a diagonal matrix whose elements are column (or row) sums of \( W \), \( D_{ii} = \sum_j W_{ij} \). \( L \) is called graph Laplacian.

Therefore, by setting \( M = L \), we get the following manifold adaptive kernel:
\[
\tilde{\mathcal{K}}(x_i, x_j) = \mathcal{K}(x_i, x_j) - \mu k_{x_i} (I + LK)^{-1} L k_{x_j}.
\]
When \( \mu = 1 \), we are able to better understand the kernel deformation. In this case, Eq. (11) can be rewritten as
\[
\tilde{\mathcal{K}} = \mathcal{K} - \mathcal{K}^\top (I + LK)^{-1} LK
\]
\[
= \mathcal{K} [ (I + LK)^{-1}L(I + LK) - (I + LK)^{-1}LK ]
\]
\[
= \mathcal{K} (I + LK)^{-1} - \mathcal{K} (I + LK)^{-1}K
\]
\[
= \mathcal{K} ((K^{-1} + L)K)^{-1}
\]
\[
= (K^{-1} + L)^{-1}.
\]
Here \( \widetilde{K} = [\widetilde{K}(x_i, x_j)]_{n \times n} \) is the kernel matrix computed by the new kernel function \( \widetilde{K}(\cdot, \cdot) \). The new kernel matrix \( \widetilde{K} \) can be regarded as the “reciprocal mean” of matrix \( K \) and \( L^{-1} \). When \( L \) is “large”, i.e., having a strong geometrical relationship among all the data points, we expect \( \widetilde{K} \) to be significantly deformed by the geometrical relationships.

### B. Kernels for SPD matrices

Since SPD matrices do not lie on the Euclidean space, an arithmetic subtraction would not measure the distance between two SPD matrices. Consequently, traditional kernels (e.g., Gaussian kernel, polynomial kernel, and linear kernel) cannot be directly transformed to manifold adaptive kernels. To address this issue, we adopt a more accurate geodesic distance on the manifold to define kernels on \( Sym^+_d \). Nevertheless, not all geodesic distances yield positive definite kernels. In this paper, we state two positive definite kernels on \( Sym^+_d \) through the true geodesic distance, as illustrated in Theorem 1 and Theorem 2.

**Theorem 1:** Let \( K^G : Sym^+_d \times Sym^+_d \rightarrow \mathbb{R} : K^G(x_i, x_j) = \exp \left( -\gamma \| \log(x_i) - \log(x_j) \|_2^2 \right) \). \( K^G \) defines a positive definite kernel for all \( \gamma \in \mathbb{R} \).

**Proof:** Before the proof of Theorem 1, we use \( K^G = [K^G(x_i, x_j)]_{n \times n} \) to denote the kernel matrix. \( K^G \) is positive definite if and only if \( Z^T K^G Z \geq 0 \), for all \( Z \in \mathbb{R}^n \). Note that \( K^G(x_i, x_i) = 1 \), thus, expanding \( Z^T K^G Z \) yields

\[
Z^T K^G Z = \sum_{i=1}^{n} \sum_{j=1}^{n} z_i z_j K^G_{i,j} = \sum_{i=1}^{n} \sum_{j=1}^{n} z_i z_j + \sum_{i=1}^{n} \sum_{j \neq i} z_i z_j K^G_{i,j} = \left( \sum_{i=1}^{n} z_i \right)^2 - \sum_{i=1}^{n} \sum_{j \neq i} z_i z_j + \sum_{i=1}^{n} \sum_{j \neq i} z_i z_j K^G_{i,j} = \left( \sum_{i=1}^{n} z_i \right)^2 + \sum_{i=1}^{n} \sum_{j \neq i} z_i z_j (K^G_{i,j} - 1).
\]

Since \( K^G_{i,j} \in (0, 1) \), for all \( z_i, z_j \), \( \min \{ z_i z_j (K^G_{i,j} - 1) \} = -z_i z_j \) holds. We get

\[
\min \{ z^T K^G z \} = \left( \sum_{i=1}^{n} z_i \right)^2 - \sum_{i=1}^{n} \sum_{j \neq i} z_i z_j = \sum_{i=1}^{n} (z_i)^2 \geq 0.
\]

**Theorem 2:** Let \( K^L : Sym^+_d \times Sym^+_d \rightarrow \mathbb{R} : K^L(x_i, x_j) = tr(\log(x_i) \log(x_j)) \). \( K^L \) defines a positive definite kernel, where \( tr \) is the matrix trace operation.

**Proof:** Using the notation \( \log(x_1) = A = [a_{ij}]_{d \times d} \), \( \log(x_2) = B = [b_{ij}]_{d \times d} \), we denote \( C = AB = [c_{ij}]_{d \times d} = (\sum_{k=1}^{d} a_{ik}b_{kj})_{d \times d} \). Since \( B \) is a symmetric matrix, we get

\[
tr(\log(x_1) \log(x_j)) = tr(C) = \sum_{i=1}^{d} \sum_{j=1}^{d} a_{ij}b_{ij} = \sum_{i=1}^{d} \sum_{j=1}^{d} a_{ij}b_{ij} = \left( \langle \log(x_i), \log(x_j) \rangle \right).
\]

Therefore, \( tr(\log(x_1) \log(x_j)) \) is an inner product. The induced norm can be used to define the distance which is equal to the geodesic distance. Furthermore, to show that the kernel matrix \( K^L = [K^L(x_i, x_j)]_{n \times n} \) is positive definite, we prove that \( Z^T K^L Z \geq 0 \) for all \( Z \in \mathbb{R}^n \), i.e.,

\[
Z^T K^L Z = \sum_{i=1}^{n} \sum_{j=1}^{n} z_i z_j K^L_{i,j} = \sum_{i=1}^{n} z_i tr(\log(x_i) \cdot \log(x_j)) z_j = tr\left( \left( \sum_{i=1}^{n} z_i \log(x_i) \right)^2 \right) = \left| \sum_{i=1}^{n} z_i \log(x_i) \right|^2 \geq 0.
\]

Based on Theorem 1 and Theorem 2, positive definite kernels \( K^G \) and \( K^L \) can be directly transformed to manifold kernels \( \widetilde{K}^G \) and \( \widetilde{K}^L \) on the Riemannian manifold of SPD matrices, respectively,

\[
\begin{align*}
\widetilde{K}^G(x_i, x_j) &= K^G(x_i, x_j) - \mu(x_i)^T (I + L K^G)^{-1} L K^G \mu(x_j), \\
\widetilde{K}^L(x_i, x_j) &= K^L(x_i, x_j) - \mu(x_i)^T (I + L K^L)^{-1} L K^L \mu(x_j).
\end{align*}
\]

(13)

In the remaining part of this article, notation \( \widetilde{K} \), instead of \( \widetilde{K}^G \) and \( \widetilde{K}^L \), is used to refer to the manifold kernel specified in Eq. (11), just for brevity.

### IV. MANIFOLD KERNEL SPARSE REPRESENTATION ON \( Sym^+_d \)

In the space of \( Sym^+_d \), we cannot use the linear combination of atoms \( \hat{x}_i = \sum_{j=1}^{N} \alpha_{ij} d_j \) to represent the data \( x_i \), since the approximation \( \hat{x}_i \) corresponding to \( x_i \) may not be on the Riemannian manifold. In this section, we perform the SR of SPD matrices by embedding Riemannian manifold into RKHS using the manifold kernels introduced in Sect.III.

#### A. Sparse coding

Employing manifold kernels specified in Eq. (13) induced by feature mapping function \( \phi : \mathcal{R}^d \rightarrow \mathcal{R}^F \), data points \( X \) on \( Sym^+_d \) are transformed to the corresponding feature space \( \{ \phi(x_1), \phi(x_2), \ldots, \phi(x_n) \} \). The kernel similarity between \( x_i \) and \( x_j \) is defined by \( \widetilde{K}(x_i, x_j) = \phi(x_i)^T \phi(x_j) \). The dictionary \( D \) in the feature space is denoted by \( \{ \phi(d_1), \phi(d_2), \ldots, \phi(d_N) \} \). The similarities between dictionary atoms and the original data points can also be computed using the kernel function as \( \phi(d_1)^T \phi(x_j) = \widetilde{K}(d_1, x_j) \) and
\[ \phi(d_i)^T \phi(d_j) = \tilde{K}(d_i, d_j). \]

For the Riemannian data points \( x \) on Sym\( _d \), we solve a sparse vector \( \alpha \in \mathbb{R}^{N \times n} \) such that \( \phi(x) \) admits the sparse representation \( \alpha \) over the dictionary \( \phi(D) \). Substituting the mapped features and basis to the formulation of sparse coding, kernelized version of sparse coding can be expressed as

\[
\min_{\alpha} \left\| \phi(x) - \phi(D)\alpha \right\|_F^2 + \lambda \left\| \alpha \right\|_1
\]

\[
\text{s.t. } \left\| \alpha \right\| = 1, \quad \alpha \geq 0
\]

For each manifold point \( x_i \), Eq.(14) can be expanded as

\[
\left\| \phi(x_i) - \phi(D)\alpha_i \right\|_F^2 + \lambda \left\| \alpha_i \right\|_1
\]

\[
= \left\| \phi(x_i) - \sum_{j=1}^{N} \phi(d_j)\alpha_{i,j} \right\|_F^2 + \lambda \left\| \alpha_i \right\|_1
\]

\[
= \tilde{K}(x_i, x_i) - 2 \sum_{j=1}^{N} \alpha_{i,j} \tilde{K}(x_i, d_j)
\]

\[
+ \sum_{j=1}^{N} \sum_{i=1}^{N} \alpha_{i,j} \alpha_{i,j} \tilde{K}(d_j, d_j) + \lambda \left\| \alpha_i \right\|_1
\]

\[
= \phi(x_i)^T \phi(x_i) - 2 \alpha_{i}^T \phi(D)^T \phi(x_i)
\]

\[
+ \alpha_{i}^T \phi(D)^T \phi(D)\alpha_{i} + \lambda \left\| \alpha_i \right\|_1
\]

\[
= \tilde{K}_{x_i,x_i} - 2 \alpha_{i}^T \tilde{K}_{x_i,d_j} + \alpha_{i}^T \tilde{K}_{d_j,d_j} \alpha_{i} + \lambda \left\| \alpha_i \right\|_1.
\]

Here, \( \tilde{K}_{DD} \) is a \( N \times N \) matrix. It contains the kernel similarities between all the dictionary atoms, i.e., \( \tilde{K}(d_i, d_j) \), where \( t = 1, 2, \ldots, N \) and \( j = 1, 2, \ldots, N \). \( \tilde{K}_{x_i} \) is a \( N \times 1 \) vector which consists of \( \tilde{K}(x_i, d_j) \), \( j = 1, 2, \ldots, N \). \( \alpha_i \in \mathbb{R}^{N \times 1} \) corresponds the sparse code of \( x_i \). The objective function in Eq.(15) is similar to the sparse coding problem expect for the definition of \( \tilde{K}_{DD} \) and \( \tilde{K}_{x_i} \), which can be calculated by the manifold kernel functions in Eq.(13).

To get the efficient solution of Eq.(15), the symmetric positive definite matrix \( \tilde{K}_{DD} \) is rewritten as \( \tilde{K}_{DD} = U \Sigma U^T \) through Singular Value Decomposition (SVD). \( U \) is an orthonormal matrix and \( \Sigma \) is a diagonal matrix. More specifically,

\[
\tilde{K}_{DD} = U \Sigma U^T = U \Sigma^{1/2} (\Sigma^{1/2})^T U^T.
\]

For simplicity, let \( \Lambda = (\Sigma^{1/2})^T \), then \( \tilde{K}_{DD} = \Lambda^T U \). Because of \( \Lambda^T (\Lambda^T)^{-1} = I \), \( \tilde{K}_{x_i} \) can be given by

\[
\tilde{K}_{x_i} = \Lambda^T (\Lambda^T)^{-1} \tilde{K}_{x_i}.
\]

Similarly, let \( \Theta = (\Lambda^T)^{-1} \Lambda \), then \( \tilde{K}_{DD} = \Lambda^T \Theta \). Since the optimization of \( \alpha_i \) is independent on \( \Theta \), we add \( \Theta^T \Theta \) into Eq. (15) and omit constant \( \tilde{K}(x_i, x_i) \) with no impact on minimizing Eq. (15). Thus, we get

\[
\Theta^T \Theta - 2 \alpha_{i}^T \Lambda^T \Theta + \alpha_{i}^T \Lambda^T \Lambda \alpha_{i} + \lambda \left\| \alpha_i \right\|_1
\]

\[
= \left\| \Theta - \Lambda \alpha \right\|_1 + \lambda \left\| \alpha_i \right\|_1.
\]

Eq. (18) is a standard Lasso problem [33] which can be solved efficiently with the SPAMS package [29]. Since we solve Eq. (18) by fixing the dictionary \( D \), both \( \tilde{K}_{DD} \) and \( (\Lambda^T)^{-1} \) are computed only once.

### B. Dictionary learning

When the kernel sparse codes for the given manifold data points \( \mathcal{X} \) are computed, the dictionary can be updated such that the reconstruction error for each \( x_i \) is minimized. The problem of learning a dictionary, therefore, can be formulated as

\[
\min_{\alpha_i, D} \sum_{i=1}^{n} \left\| \phi(x_i) - \sum_{j=1}^{n} \phi(d_j) \alpha_{i,j} \right\|_F^2 + \lambda \left\| \alpha_i \right\|_1
\]

Writing the first term of the objective in Eq.(19) as a function of \( D \) for dictionary update, we have

\[
f(D) = \sum_{i=1}^{n} \left[ 1 - 2 \sum_{j=1}^{n} \alpha_{i,j} \tilde{K}(x_i, d_j)ight] + \sum_{j=1}^{n} \sum_{i=1}^{n} \alpha_{i,j} \alpha_{i,j} \tilde{K}(d_j, d_j),
\]

where \( \alpha_{i,j} \) denotes the \( i \)-th element in the coefficient vector \( \alpha_j \). After initializing the dictionary \( D \), we solve the optimization by iterative method, repeating two steps (i.e., sparse coding and dictionary update). To update the dictionary atom \( \{d_j\}_{j=1}^{N} \), we compute the derivative of Eq.(20) with respect to \( d_j \):

\[
\frac{\partial f(D)}{\partial d_j} = \sum_{i=1}^{n} \left[ -2 \alpha_{i,j} \frac{\partial \tilde{K}(x_i, d_j)}{\partial d_j} + \sum_{i=1}^{n} \alpha_{i,j} \frac{\partial \tilde{K}(d_i, d_j)}{\partial d_j} \right],
\]

We take \( \tilde{K}^G \) as an example to perform the dictionary learning (\( \tilde{K}^L \) can be carried out by the similar scheme). We set the derivative of Eq.(21) to 0 using the definition of manifold adaptive kernel functions \( \tilde{K}^G(x_i, x_j) \) in Eq.(13) as follows:

\[
-4 \gamma d_j^{-1} \sum_{i=1}^{n} \left( -\alpha_{i,j} \left( \tilde{K}^G(x_i, d_j) \left( \log(d_j) - \log(x_j) \right) \right) + \mu \cdot \sum_{s=1}^{n} \Phi_s \tilde{K}^G(x_s, d_j) \left( \log(d_j) - \log(x_s) \right) \right)
\]

\[
+ \sum_{i=1}^{n} \alpha_{i,j} \alpha_{i,j} \tilde{K}^G(d_i, d_j) \left( \log(d_j) - \log(d_i) \right) - \mu \cdot \sum_{s=1}^{n} \Psi_s \tilde{K}^G(x_s, d_j) \left( \log(d_j) - \log(x_s) \right) \right)
\]

\[
= 0,
\]

where

\[
\begin{align*}
\Phi &= (k_i^G)^T (I + L K^G)^{-1} L \\
\Psi &= (k_i^G)^T (I + L K^G)^{-1} L.
\end{align*}
\]

Here, \( \Phi \in \mathbb{R}^{1 \times n} \) and \( \Psi \in \mathbb{R}^{1 \times n} \). During updating, each dictionary atom is updated independently. At time \( t + 1 \), \( d_j \) is updated using and the results of \( t \). \( d_j^{(t)} \) represents the \( j \)-th
atom in the \( t \)-th iteration. Eq.(22) can be rewritten as

\[
-4 \sum_{i=1}^{n} \left[ -\alpha_{j,i} \left( K^G(x_i, d_j^{(t)}) \right) \left( \log(d_j) + 1 \right) - \log(x_i) \right] - \mu \cdot \sum_{s=1}^{n} \Phi^s K^G(x_s, d_j^{(t)}) \left( \log(d_j) + 1 \right) - \log(x_s)
\]

\[
+ \sum_{i=1}^{N} \alpha_{j,i} \alpha_{t,i} \left( K^G(d_i, d_j^{(t)}) \right) \left( \log(d_j) + 1 \right) - \log(x_i)
\]

\[
- \mu \cdot \sum_{s=1}^{n} \Psi^s K^G(x_s, d_j^{(t)}) \left( \log(d_j) + 1 \right) - \log(x_s)
\]

\[
= 0.
\]

Solving Eq.(24), we obtain the iterative formulation in Eq.(25). Here \( \text{diag}[\cdot] \) denotes a diagonal matrix using the element as its diagonal. \( \mathbf{1}_n \) and \( \mathbf{1}_N \) are \( n \)-dimension and \( N \)-dimension 1 column vectors, respectively. \( \alpha_j \in \mathbb{R}^{1 \times n} \) is a row vector containing the set of coefficients of data points corresponding to the dictionary atom \( d_j \). Note that \( K^G = [K^G(x_i, x_j)]_{n \times n} \) is replaced with \( K \) in Eq. (25) for simplicity. Kernel matrix \( K^{(1)}_{d, D} \in \mathbb{R}^{1 \times N} \) contains the kernel similarity elements \( K^G(d_i^{(t)}, d_k) \) between each atom \( d_i \) and the entire dictionary \( D \) where \( k = 1, 2, \ldots, N \). Similarly, kernel matrix \( K^{(1)}_{d, x} \in \mathbb{R}^{1 \times x} \) contains the kernel similarity elements \( K^G(d_i^{(t)}, x_i) \) between each atom \( d_j \) and all data points. Algorithm 1 outlines the details for dictionary learning.

V. EXPERIMENTS

In this section, we evaluate the proposed manifold kernel sparse representation using three applications including visual tracking, face recognition and image classification. The experiments are implemented in MATLAB on an Intel Core2 2.5 GHz processor with 4GB RAM. The source code is available at http://site.lab.bit.edu.cn/micslalab/~wuyuwei/Publications.htm (The password of unzipping is Trans_for_reviewers).

A. Visual tracking

Motivated by recent advances of sparse coding for visual tracking [3], [34], [35], [36], we employ the sparse coding of SPD matrices as the object representation. Tracking is then carried out within a Bayesian inference framework, in which the bin-ratio similarity function [37] of sparse histograms between the candidate and the template are used to construct the observation model.

\[
\log(d_j^{(t+1)}) = \log(d_j) \text{diag}[K_{d, D}] \alpha_j - \log(x) \text{diag}[K_{d, D}] \alpha_j + \mu \log(x) \text{diag}[K_{d, D}] \alpha_j + \mu \log(x) \text{diag}[K_{d, D}] \alpha_j - \mu \log(x) \text{diag}[K_{d, D}] \alpha_j + \mu K_{d, D} \alpha_j + \mu \Psi 1_N \alpha_j - \mu K_{d, D} \alpha_j + \mu \Psi 1_N \alpha_j
\]

(25)

Algorithm 1: Dictionary learning on \( \text{Sym}^1(n) \) using kernel trick

Input:

- Original data points \( x = \{x_1, x_2, \ldots, x_n\} \) on the Riemannian manifold of SPD matrices, where each \( x_i \in \text{Sym}^1(n) \) is a SPD matrix.
- The Riemannian kernel function \( K(x_i, x_j) \).
- The number of iterations \( \text{iter} \).

Output:

- The Riemannian dictionary \( D = \{1, \ldots, N\} \).

1. Compute \( \tilde{K}_{x_i} = K(x_i, x_i), i = 1, 2, \ldots, n \).
2. Initialize the dictionary \( D^1 = \{d_j\}_{j=1}^N \) by Riemannian clustering using Karcher mean.
3. for \( t = 1 \rightarrow \text{iter} \) do
   4. Compute \( \tilde{K}_{D^t} = \tilde{K}_{D^t} d_j, d_k \), \( t, j, k = 1, 2, \ldots, N \);
   5. Compute \( \tilde{K}_{D^t} = K(d_i, x_j), i = 1, 2, \ldots, n \);
   6. Compute \( \|\Theta - \Delta \alpha_i\|^2 + \lambda \|\alpha_i\|_1 \), \( \forall \, i \in \mathcal{X} \).
   for \( j = 1 \rightarrow N \) do
      7. compute \( \log(d_j^{(t+1)}) \) according to Eq. (25);
      8. \( d_j^{(t+1)} = \text{exp}(\log(d_j^{(t+1)})\); 
      9. \( d_j^{(t+1)} \leftarrow d_j^{(t+1)} \)
   end
end

1) Experimental Setup: To take local appearance information of patches into consideration, we resize the object image to 32 × 32 pixels and extract 36 overlapped 12 × 12 sliding windows (or local patches) within the object region with 4 pixels as the step length. Following [25], the covariance descriptors are computed from the feature vector \( [x; y; I_x; |I_x|; |I_y|; \sqrt{(I_x)^2 + (I_y)^2}; I_{xx}; I_{yy}] \). The covariance matrix for each image patch, therefore, is an 8 × 8...
SPD matrix. With the overlapped patches extracted from the object region in the first frame, the $k$-means clustering is performed in the Log-Euclidean framework [24] to obtain the dictionary $\mathcal{D}$ with 72 atoms. The sparse coefficient vector of each patch is normalized and concatenated to form a histogram representation by $[\alpha_1, \alpha_2, \cdots, \alpha_{56}]^T$. Due to the space limit, we only provide the corresponding tracking results of the $K^G$ kernel sparse coding of SDP matrices in this paper.

We compare our tracker with the state-of-the-art sparsity-based tracking algorithms including L1 [38], APGL1 [39], LSK [36], MTT [2], LSST [35], SCM [34], and MLSAM [3]. We run our method on 8 challenging video sequences which suffer from heavy occlusions, illumination changes, pose variations, motion blur, scale variations and complex backgrounds. Some representative results are presented in this section. The parameters $\beta$ and $\mu$ are set as 1 and 0.01, respectively.

2) Quantitative comparisons: One widely used evaluation method to measure the tracking results is the center location error. It is based on the relative position errors (in pixels) between the central locations of the tracked object and those of the ground truth. Ideally, an optimal tracker is expected to have a small error. From Table I, we can see that our algorithm achieves lowest tracking errors in almost all the sequences.

However, when the tracker lost the object for some frames, the output location can be random and therefore the av-

![Fig. 2. Precision plot for 8 representative sequences. The performance score of each tracker is shown in the legend (best viewed on high-resolution display).](image1)

![Fig. 3. Success rate curve for 8 representative sequences. The performance score of each tracker is shown in the legend (best viewed on high-resolution display).](image2)

<table>
<thead>
<tr>
<th></th>
<th>APGL1</th>
<th>LSST</th>
<th>MTT</th>
<th>SCM</th>
<th>MLSAM</th>
<th>L1</th>
<th>LSK</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deer</td>
<td>41.9</td>
<td>52.8</td>
<td>7.6</td>
<td>6.5</td>
<td>9.9</td>
<td>56.9</td>
<td>96.4</td>
<td>6.4</td>
</tr>
<tr>
<td>Freeman1</td>
<td>11.9</td>
<td>29.8</td>
<td>123.1</td>
<td>6.9</td>
<td>99.1</td>
<td>54.5</td>
<td>92.4</td>
<td>7.8</td>
</tr>
<tr>
<td>David3</td>
<td>237.0</td>
<td>71.5</td>
<td>58.7</td>
<td>73.1</td>
<td>6.0</td>
<td>250.1</td>
<td>227.0</td>
<td>6.9</td>
</tr>
<tr>
<td>Boy</td>
<td>58.5</td>
<td>284.8</td>
<td>492.0</td>
<td>98.7</td>
<td>98.7</td>
<td>93.5</td>
<td>176.3</td>
<td>39.8</td>
</tr>
<tr>
<td>Deer</td>
<td>88.1</td>
<td>5.2</td>
<td>43.0</td>
<td>51.1</td>
<td>5.0</td>
<td>83.8</td>
<td>180.6</td>
<td>3.0</td>
</tr>
<tr>
<td>Shaking</td>
<td>26.3</td>
<td>89.7</td>
<td>12.8</td>
<td>102.6</td>
<td>12.7</td>
<td>41.3</td>
<td>176.5</td>
<td>10.3</td>
</tr>
<tr>
<td>Skating1</td>
<td>122.3</td>
<td>121.6</td>
<td>55.1</td>
<td>10.0</td>
<td>23.0</td>
<td>174.2</td>
<td>114.2</td>
<td>8.4</td>
</tr>
<tr>
<td>Dudek</td>
<td>151.1</td>
<td>159.1</td>
<td>255.8</td>
<td>35.1</td>
<td>15.4</td>
<td>117.2</td>
<td>44.0</td>
<td>9.5</td>
</tr>
</tbody>
</table>
Average center location errors may not evaluate the tracking performance correctly. In this paper, the precision plot is also adopted to measure the overall tracking performance. It shows the percentage of frames whose estimated location is within the given threshold distance (e.g., 20 pixels) of the ground truth. More accurate trackers have higher precision at lower thresholds. If a tracker loses the object it is difficult to reach a higher precision. We provide the precision plot results of 8 trackers over eight representative sequences, as shown in Fig. 2. We see that the proposed tracker achieves the most robust and accurate tracking performance on most video sequences.

The tracking overlap rate indicates stability of each algorithm as it takes the size and pose of the target object into account. It is defined by $score = \frac{\text{area}(\text{ROI}_T \cap \text{ROI}_G)}{\text{area}(\text{ROI}_T \cup \text{ROI}_G)}$, where $\text{ROI}_T$ is the tracking bounding box and $\text{ROI}_G$ is the ground truth. This can be used to evaluate the success rate of any tracking approach. Table II gives the average overlap rates. Overall, the proposed tracker outperforms state-of-the-art methods.

Generally, the tracking result is considered as a success when the score is greater than the given threshold $t_s$. It may not be fair or representative for tracker evaluation using one success rate value at a specific threshold (e.g., $t_s = 0.5$). Further, we count the number of successful frames at the thresholds varied from 0 to 1 and plot the success rate curve for our tracker and the compared trackers. The area under curve (AUC) of each success rate plot is employed to rank the tracking algorithms. The robust trackers have higher success rate at higher thresholds. The success rate curve of 8 representative sequences is illustrated in Fig. 3. We can see that the proposed method gets the best tracking performances on most video sequences.
3) Qualitative comparisons: We report the corresponding tracking results of the eight trackers (highlighted by the bounding boxes in different colors) over the representative frames of the 8 video sequences, as shown in Fig. 4. In the “Shaking” sequence, the target undergoes pose variation, illumination change, and partial occlusion. The SCM, L1, and LSK trackers drift from the object quickly when the spotlight blinks suddenly (e.g., 960). MLSAM and our trackers are able to successfully track the surfer throughout the sequence with relatively accurate sizes of the bounding box. MTT and APGL1 methods are able to track the object in this sequence but with lower success rate than our method. In the “Skating1” sequence, the dancer continuously changes her pose on a stage with complex background as well as drastic illumination variations. L1, APGL1, LSST, MTT and LSK methods cannot track the object correctly. The MLSAM method performs slightly better. Our tracker loses the object at the frame 9359, but recover at the frame 9368. Overall, SCM and our methods outperform the other trackers.

“Boy” and “Deer” sequences are used to evaluate whether our method is able to tackle the fast motion. In the “Boy” sequences, a boy jumps irregularly where the object undergoes fast motion and out-of-plane. It is difficult to predict his locations. Most methods fail to track the object at the beginning of the sequence (e.g., 9295). In contrast, our method achieve relatively lower center location errors and higher success rates than the other methods. In the “Deer” sequence, the appearance change caused by motion blur is more drastic. APGL1, MTT, SCM, L1 and LSK trackers do not perform well in some frames (e.g., 9240, 9263). Though LSST and MLSAM trackers are able to keep track of the object to the end, the proposed approach achieves both the lowest tracking error and the highest overlap rate.

In the “Dudek”, “Freeman1” and “Sylvester” sequences, the object suffer from large pose and view changes. For the “Dudek” sequence, we see that LSK tracker loses the target very quickly at the beginning of the sequence (e.g., 92363). The LSST, MTT, and APGL1 trackers fails when the scale change occurs (e.g., 92853). In contrast, our method gets both relatively low center location error and high overlap rate, as shown in Table I and Table II. In the “Freeman1” sequence, Although the LSST tracker obtains slightly better results than MTT, MLSAM, LSK and L1 trackers, it loses the object after drastic pose change (e.g., 92265). In comparison, APGL1 and our trackers track the object successfully. We note that the SCM perform better than the other methods. For the “Sylvester” sequence, APGL1, LSST, L1 and LSK trackers are not able to locate the object on the whole sequence. In contrast, MTT, SCM, MLSAM and our methods can track the object well and provides tracking boxes that are much more accurate and consistent.

In the “David3” sequence, the person suffers from partial occlusion as well as drastic pose variations. It is difficult to handle both of these two challenges. The SCM, APGL1, L1 and MTT methods fail to track the object after the person walks behind a tree (e.g., 9284). The LSST and LSK methods lose the object after the person changes his direction (e.g., 92159). In comparison, only MLSAM and our trackers succeed throughout this sequence.

### B. Image Classification

We evaluate our method on the Scene 15 dataset [40] and Brodatz dataset [41]. The Scene 15 dataset contains 4485 images of 15 different scenes, where 8 categories are originally collected by Oliva et al., 5 are provided by Li et al., and 2 are added by Lazebnik et al. [40]. The number of images per category ranges from 200 to 400. The dataset contains not only indoor scenes, e.g., store, living-room, but also outdoor scenes, e.g., streets and mountain. To be consistent with previous work [21], we use the same setting to extract covariance descriptors. 64 covariance descriptors of $16 \times 16$ pixel patches are computed over a grid with spacing of 8 pixels. The raw feature vectors are orientation histogram of 8 bins. We randomly select 100,000 covariance matrices from the total covariance descriptors of all images to learn the dictionary. In order to give a better estimation of the generalization performance, the reported results of the dataset are the averages of 20 independent experiments.

In the Brodatz dataset each class corresponds to only one image. All 111 texture images are used to train the dictionary. Keeping consistent with the previous work [19], [21], we normalize each training image to $256 \times 256$ pixels, and break down into nonoverlapping blocks of $32 \times 32$ pixels. A $5 \times 5$ covariance descriptor is then computed from each of these blocks using the feature vector $[I_{xx}, \|I_x\|, \|I_y\|, \|I_{xx}\|, \|I_{yy}\|]^{\top}$. 20 blocks from each image were randomly selected for the dictionary learning. We evaluated the recognition performance with the number of training samples fixed at 20, 25, and 30 covariance matrices per class and the remaining ones for testing. The reported performance is obtained by averaging over 10 different random splits of train and test sets.

1) Parameter Selection: The kernel deformation parameter $\mu$ is an important factor in manifold kernel sparse representation. To ease the parameter selection, we experimentally test its effect on the performance of image classification on the Brodatz dataset. In the manifold kernel $K^C, K^C(x_i, x_j) = \exp(-\gamma \|\log(x_i) - \log(x_j)\|_F^2)$, we fix its parameter $\gamma = 1/d$, where $d$ is the dimensionality of the covariance descriptor. We list the results based on different $\mu$ ranging from 0.001 to

### TABLE III

**AVERAGE CLASSIFICATION RATE (%) ON SCENE 15 DATASET.**

<table>
<thead>
<tr>
<th>Num. of atoms</th>
<th>32</th>
<th>64</th>
<th>128</th>
<th>256</th>
<th>512</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log-E kernel [21]</td>
<td>$75.84 \pm 0.64$</td>
<td>$79.27 \pm 0.65$</td>
<td>$80.92 \pm 0.44$</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Our $K^L$ kernel</td>
<td>$73.49 \pm 0.58$</td>
<td>$77.99 \pm 0.50$</td>
<td>$81.57 \pm 0.46$</td>
<td>$82.46 \pm 0.47$</td>
<td>$82.69 \pm 0.43$</td>
</tr>
<tr>
<td>Our $K^G$ kernel</td>
<td>$74.21 \pm 0.47$</td>
<td>$78.56 \pm 0.56$</td>
<td>$83.35 \pm 0.39$</td>
<td>$84.18 \pm 0.42$</td>
<td>$84.31 \pm 0.44$</td>
</tr>
</tbody>
</table>
1 in Fig. 5(a). We observe that our method can obtain the good performance when $\mu$ is fixed at either 0.015 or 0.2. In addition, when $K^{G}$ is used in the sparse representation, the kernel parameter $\gamma$ is also very important and affects the classification accuracy. To depict the relationship between $\gamma$ and classification accuracy, we set $\gamma = 3^{n}/d$, where $n$ ranges from −3 to 2 with step size 1. The relationship between $\gamma$ and classification accuracy are shown in Fig. 5(b). We see that our method can achieve promising result in a wide range of $\gamma$ value. For simplicity, in the following experiments, we set $\gamma = 1/d$.

2) Result Comparisons: For the Scene 15 dataset, we apply the manifold kernel sparse coding to feature quantization in the framework of Bag-of-Words image representation, which achieves good performance in solving the image classification problem. Table III shows detailed results of classification experiments using 100 images per class for training and the rest for testing. Experimental results show that the proposed kernels have comparable performance while $K^{G}$ is more better ($\mu = 0.015$, $\gamma = 0.03$). The probable reason is that, by using a data-dependent kernels on RKHS, we can discover the non-linear structure of the RKHS to reflect the underlying geometry of the data. Thus, each image can be more accurately represented.

To explore the geometrical information of the samples, we randomly selected three classes from the Scene 15 dataset. The data distribution obtained by embedding the manifold-values data into vector space $\mathbb{R}^{d}$ [16] is shown in Fig. 6(a) in a 3-D space for ease of presentation. We can see that there is high overlapping among the manifolds corresponding to different scenes. Fig. 6(b) illustrates the distribution obtained by our manifold kernel sparse representation. We see that there is a clear separation for the manifolds corresponding to different scenes. Hence, more discriminative information can be exploited for classification.

We also notice that as the number of atoms in the dictionary grows, the average classification rates increase in Table III. But the results are very close. The underlying reason is that our dictionary learning method is generative without discriminative information, therefore, the good representational capability does not necessarily mean promising discriminability.

Furthermore, we use the Brodatz dataset [41] to evaluate the performance of our method on texture classification. The KNN classifier is employed for the classification task($k = 3$). The results are reported in Table IV. It can be seen that our $K^{L}$ kernel ($\mu = 0.015$) achieve the comparable performance with Log-Euclidean method [21]. $K^{G}$ kernel ($\mu = 0.015$, $\gamma = 0.02$) performs better than other approaches, because the radial basis function $K^{G}$ can further reflect the geometrical structure of the manifold data than the polynomial kernel $K^{L}$. To further analyze our results, we plot the average classification accuracy curves using $K^{G}$ versus the number of atom matrices in Fig. 7. We see that as the number of atoms increases, the improvement of our method grows. Our method gets the best result when the number of atoms is equal to 95.

C. Face recognition

In this section, we present experimental results for face recognition on benchmarks including the FERET dataset [42], the Extended Yale B dataset [43], and the YTC dataset [44]. The sparse representation classifier [4] is adopted for the classification task. We used the “b” subset of the FERET dataset for evaluation of recognition performance. the subset includes 1400 images from 198 subjects. In our experiments, the images are resized to $64 \times 64$ pixels. The Extended Yale B

![Fig. 5. The effect of the parameters $\mu$ and $\gamma$ on the Brodatz dataset. (a) The relationship between the classification accuracy and the parameter $\mu$. (b) The relationship between the classification accuracy and the parameter $\gamma$.](image)

![Fig. 6. Visualization of three manifolds corresponding to three scenes from the Scene 15 dataset. (a) Distribution obtained by embedding the manifold-values data into vector space $\mathbb{R}^{d}$ [16]. (b) Distribution obtained by our manifold kernel sparse representation.](image)

![Fig. 7. The average classification accuracy curves vs. the number of atom matrices on the Brodatz dataset.](image)
dataset contains about 2,414 frontal face images of 38 subjects (around 59 – 64 images for each person). The face images are taken under varying illumination conditions. We normalize face images of size $54 \times 48$. The YTC dataset has 1910 video clips of 47 subjects collected from YouTube. Most videos are low resolution and high compression ratio which result in noisy and low-quality image frames. Each image frame is automatically detected using a cascaded face detector [45] and then resized to a $30 \times 30$ intensity image.

1) Experimental Setup: To obtain the manifold kernel sparse representation, a $43 \times 43$ covariance descriptor is used to describe a face image using the feature vector

$$[I(x, y), x, y, |G_{00}(x, y)|, \cdots, |G_{47}(x, y)|]^	op,$$

where $I(x, y)$ is the intensity value at position $(x, y)$, and $G_{uv}(x, y)$ is the response of a 2D Gabor filters along 5 orientations and 8 angles. For each pixel $(x, y)$, the dimensionality of the Gabor features is 40. For the FERET dataset, images marked “ba”, “bj” and “bk” are used as training data, and images with “bd”, “be”, “bf” and “bg” labels are as test data. For the Extended Yale B dataset, we randomly split the dataset into two halves. One half containing 32 images for each person is used as the dictionary, and the other one is used for testing. We follow the settings in [6], we partition the whole YTC dataset into 5 folds, and each fold contains 9 videos for each person. For each fold, 3 face videos are randomly selected for training and the remaining ones are for testing. On the Extended Yale B dataset and YTC dataset, we conduct experiments 10 times by randomly selecting training and testing sets, and report the average result for each dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>Dimensionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRC [4]</td>
<td>90.9</td>
</tr>
<tr>
<td>GSRC [5]</td>
<td>88.1</td>
</tr>
<tr>
<td>Log-E kernel [21]</td>
<td>96.4</td>
</tr>
<tr>
<td>Ours $\mathcal{K}_L$ kernel</td>
<td>97.4</td>
</tr>
<tr>
<td>Ours $\mathcal{K}_G$ kernel</td>
<td>98.1</td>
</tr>
</tbody>
</table>

Table VII tabulates the average recognition rates on the YTC dataset. We compare our approach with 4 state-of-the-art face recognition methods, including CDL [46], DFRV [47], Log-E kernel [21], and SFDL [6]. We see that our approach with $\mathcal{K}_G$ performs better than the other compared methods, and $\mathcal{K}_L$ achieves comparable results with SFDL. This is because the covariance matrix descriptor is able to capture both spatial and statistical properties of each pixel in an object region with a low dimensional representation, and is robust to the variations in illumination, view, and pose.

VI. CONCLUSION

In this paper, we have presented a manifold kernel sparse representation method for symmetric positive definite (SPD) matrices. The sparse representation on the space of SPD matrices can be performed by embedding the SPD matrices into a reproducing kernel Hilbert spaces (RKHS) using the data-dependent manifold kernel function. The graph Laplacian as a smooth operator of the manifold-valued data is also incorporated into the manifold kernel space to discover the underlying geometry structure of the manifold. Experimental results of visual tracking, face recognition and image classification show that our algorithm outperforms existing sparse coding based approaches, and compares favorably to the state-of-the-art methods.

ACKNOWLEDGEMENT

The authors would like to thank Zhen Dong and Shiyi Zhang for their efforts on conducting experiments.

REFERENCES


Yuwei Wu received the Ph.D. degree in computer science from Beijing Institute of Technology (BIT), Beijing, China, in 2014. He is now a research fellow at Rapid-Rich Object Search (ROSE) Lab, School of Electrical & Electronic Engineering, Nanyang Technological University (NTU), Singapore. He has strong research interests in computer vision, medical image processing and object tracking.

Yunde Jia (M’11) received the M.S. and Ph.D. degrees in mechatronics from the Beijing Institute of Technology (BIT), Beijing, China, in 1986 and 2000, respectively. He is currently a Professor of computer science with BIT, and serves as the Director of the Beijing Laboratory of Intelligent Information Technology, School of Computer Science. He has previously served as the Executive Dean of the School of Computer Science, BIT, from 2005 to 2008. He was a Visiting Scientist at Carnegie Mellon University, Pittsburgh, PA, USA, from 1995 to 1997, and a Visiting Fellow at the Australian National University, Acton, Australia, in 2011. His current research interests include computer vision, media computing, and intelligent systems.

Peihua Li received the B.S. and M.S. degrees from Harbin Engineering University, China, in 1993 and 1996, respectively, and the Ph.D. degree in computer science and technology from Harbin Institute of Technology, China, in 2002. From January to April 2003, he visited Microsoft Research Asia. From September 2003 to August 2004, he was a post-doctoral fellow at INRIA, IRISA, Rennes, France. From 2004 to 2013, he was with School of Computer Science and Technology, Heilongjiang University, Now he is with School of Information and Communication Engineering, Dalian University of Technology. He received the best Ph.D. dissertation award in 2004 from Harbin Institute of Technology, and honorary nomination of National Excellent Doctoral dissertation in China in 2005. His research interests include computer vision, pattern recognition and machine learning.

Jian Zhang (SM’04) received the B.Sc. degree from East China Normal University, Shanghai, China, in 1982; the M.Sc. degree in computer science from Flinders University, Adelaide, Australia, in 1994; and the Ph.D. degree in electrical engineering from the University of New South Wales (UNSW), Sydney, Australia, in 1999. From 1997 to 2003, he was with the Visual Information Processing Laboratory, Motorola Labs, Sydney, as a Senior Research Engineer, and later became a Principal Research Engineer and a Foundation Manager with the Visual Communications Research Team. From 2004 to July 2011, he was a Principal Researcher and a Project Leader with National ICT Australia, Sydney, and a Conjoint Associate Professor with the School of Computer Science and Engineering, UNSW. He is currently an Associate Professor with the Advanced Analytics Institute, School of Software, Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney, and also a Visiting Researcher with the Neville Roach Laboratory, National ICT Australia, Kensington, Australia. He is the author or co-author of 90 paper publications, book chapters, and ten patents filed in the U.S., including six issued patents. His current research interests include social multimedia signal processing, image and video processing, machine learning, pattern recognition, human-computer interaction and intelligent healthcare systems.

Dr. Zhang was the General Chair of the IEEE International Conference on Multimedia and Expo in 2012. He is Associated Editors for the IEEE Transactions on Circuits and Systems for Video Technology and the EURASIP Journal on Image and Video Processing.

Junsong Yuan (M08) is currently a Nanyang Assistant Professor and program director of video analytics at School of EEE, Nanyang Technological University, Singapore. He received Ph.D. from Northwestern University, USA, and M.Eng. from National University of Singapore. Before that, he graduated from Special Class for the Gifted Young of Huazhong University of Science and Technology, China. His research interests include computer vision, video analytics, large-scale visual search and mining, human computer interaction etc. He has published over 100 technical papers, and filed three US patents and two provisional US patents. He serves as area chair for IEEE Winter Conf. on Computer Vision (WACV14), IEEE Conf. on Multimedia Expo (ICME14), Asian Conf. on Computer Vision (ACCV14), organizing chair for ACCV14, and cochairs workshops at IEEE Conf. Computer Vision and Pattern Recognition (CVPR1213) and IEEE Conf. on Computer Vision (ICCV13). He currently also serves as associate editor for The Visual Computer journal (TVC) and Journal of Multimedia. He received Nanyang Assistant Professorship and Tan Chin Tuan Exchange Fellowship from Nanyang Technological University, Outstanding EECS Ph.D. Thesis award from Northwestern University, Best Doctoral Spotlight Award from CVFPR09, and National Outstanding Student from Ministry of Education, P.R.China. He recently gives tutorials at IEEE ICIP13, FG13, ICME12, SIGGRAPH VRCA112, and PCM12.
IEEE COPYRIGHT AND CONSENT FORM

To ensure uniformity of treatment among all contributors, other forms may not be substituted for this form, nor may any wording of the form be changed. This form is intended for original material submitted to the IEEE and must accompany any such material in order to be published by the IEEE. Please read the form carefully and keep a copy for your files.

TITLE OF PAPER/ARTICLE/REPORT, INCLUDING ALL CONTENT IN ANY FORM, FORMAT, OR MEDIA (hereinafter, “the Work”):

Manifold Kernel Sparse Representation of Symmetric Positive Definite Matrices and Its Applications

COMPLETE LIST OF AUTHORS:

Yuwei Wu, Yunde Jia, Peihua Li, Jian Zhang, and Junsong Yuan

IEEE PUBLICATION TITLE (Journal, Magazine, Conference, Book):

IEEE Transactions on Image Processing

COPYRIGHT TRANSFER

1. The undersigned hereby assigns to The Institute of Electrical and Electronics Engineers, Incorporated (the “IEEE”) all rights under copyright that may exist in and to:
   (a) the above Work, including any revised or expanded derivative works submitted to the IEEE by the undersigned based on the Work; and (b) any associated written or multimedia components or other enhancements accompanying the Work.

CONSENT AND RELEASE

2. In the event the undersigned makes a presentation based upon the Work at a conference hosted or sponsored in whole or in part by the IEEE, the undersigned, in consideration for his/her participation in the conference, hereby grants the IEEE the unlimited, worldwide, irrevocable permission to use, distribute, publish, license, exhibit, record, digitize, broadcast, reproduce and archive, in any format or medium, whether now known or hereafter developed: (a) his/her presentation and comments at the conference; (b) any written materials or multimedia files used in connection with his/her presentation; and (c) any recorded interviews of him/her (collectively, the “Presentation”). The permission granted includes the transcription and reproduction of the Presentation for inclusion in products sold or distributed by IEEE and live or recorded broadcast of the Presentation during or after the conference.

3. In connection with the permission granted in Section 2, the undersigned hereby grants IEEE the unlimited, worldwide, irrevocable right to use his/her name, picture, likeness, voice and biographical information as part of the advertisement, distribution and sale of products incorporating the Work or Presentation, and releases IEEE from any claim based on right of privacy or publicity.

4. The undersigned hereby warrants that the Work and Presentation (collectively, the “Materials”) are original and that he/she is the author of the Materials. To the extent the Materials incorporate text passages, figures, data or other material from the works of others, the undersigned has obtained any necessary permissions. Where necessary, the undersigned has obtained all third party permissions and consents to grant the license above and has provided copies of such permissions and consents to IEEE.

☐ Please check this box if you do not wish to have video/audio recordings made of your conference presentation.

See reverse side for Retained Rights/Terms and Conditions, and Author Responsibilities.

GENERAL TERMS

- The undersigned represents that he/she has the power and authority to make and execute this assignment.
- The undersigned agrees to indemnify and hold harmless the IEEE from any damage or expense that may arise in the event of a breach of any of the warranties set forth above.
- In the event the above work is not accepted and published by the IEEE or is withdrawn by the author(s) before acceptance by the IEEE, the foregoing copyright transfer shall become null and void and all materials embodying the Work submitted to the IEEE will be destroyed.
- For jointly authored Works, all joint authors should sign, or one of the authors should sign as authorized agent for the others.

(1) ____________________
Author/Authorized Agent for Joint Authors

(2) ____________________
U.S. GOVERNMENT EMPLOYEE CERTIFICATION (WHERE APPLICABLE)

This will certify that all authors of the Work are U.S. government employees and prepared the Work on a subject within the scope of their official duties. As such, the Work is not subject to U.S. copyright protection.

(2) ____________________
Authorized Signature

(2) ____________________
Date

(Authors who are U.S. government employees should also sign signature line (1) above to enable the IEEE to claim and protect its copyright in international jurisdictions.)
This will certify that all authors of the Work are employees of the British or British Commonwealth Government and prepared the Work in connection with their official duties. As such, the Work is subject to Crown Copyright and is not assigned to the IEEE as set forth in the first sentence of the Copyright Transfer Section above. The undersigned acknowledges, however, that the IEEE has the right to publish, distribute and reprint the Work in all forms and media.

(3) 

Authorized Signature

Date

(Authors who are British or British Commonwealth Government employees should also sign line (1) above to indicate their acceptance of all terms other than the copyright transfer.)

IEEE COPYRIGHT FORM (continued)

RETAINED RIGHTS/TERMS AND CONDITIONS

General

1. Authors/employers retain all proprietary rights in any process, procedure, or article of manufacture described in the Work.
2. Authors/employers may reproduce or authorize others to reproduce the Work, material extracted verbatim from the Work, or derivative works for the author’s personal use or for company use, provided that the source and the IEEE copyright notice are indicated, the copies are not used in any way that implies IEEE endorsement of a product or service of any employer, and the copies themselves are not offered for sale.
3. In the case of a Work performed under a U.S. Government contract or grant, the IEEE recognizes that the U.S. Government has royalty-free permission to reproduce all or portions of the Work, and to authorize others to do so, for official U.S. Government purposes only, if the contract/grant so requires.
4. Although authors are permitted to re-use all or portions of the Work in other works, this does not include granting third-party requests for reprinting, republishing, or other types of re-use. The IEEE Intellectual Property Rights office must handle all such third-party requests.
5. Authors whose work was performed under a grant from a government funding agency are free to fulfill any deposit mandates from that funding agency.

Author Online Use

6. Personal Servers. Authors and/or their employers shall have the right to post the accepted version of IEEE-copyrighted articles on their own personal servers or the servers of their institutions or employers without permission from IEEE, provided that the posted version includes a prominently displayed IEEE copyright notice and, when published, a full citation to the original IEEE publication, including a link to the article abstract in IEEE Xplore. Authors shall not post the final, published versions of their papers.
7. Classroom or Internal Training Use. An author is expressly permitted to post any portion of the accepted version of his/her own IEEE-copyrighted articles on the author’s personal web site or the servers of the author’s institution or company in connection with the author’s teaching, training, or work responsibilities, provided that the appropriate copyright, credit, and reuse notices appear prominently with the posted material. Examples of permitted uses are lecture materials, course packs, e-reserves, conference presentations, or in-house training courses.
8. Electronic Preprints. Before submitting an article to an IEEE publication, authors frequently post their manuscripts to their own web site, their employer’s site, or to another server that invites constructive comment from colleagues. Upon submission of an article to IEEE, an author is required to transfer copyright in the article to IEEE, and the author must update any previously posted version of the article with a prominently displayed IEEE copyright notice. Upon publication of an article by the IEEE, the author must replace any previously posted electronic versions of the article with either (1) the full citation to the IEEE work with a Digital Object Identifier (DOI) or link to the article abstract in IEEE Xplore, or (2) the accepted version only (not the IEEE-published version), including the IEEE copyright notice and full citation, with a link to the final, published article in IEEE Xplore.

INFORMATION FOR AUTHORS

Author Responsibilities

The IEEE distributes its technical publications throughout the world and wants to ensure that the material submitted to its publications is properly available to the readership of those publications. Authors must ensure that their Work meets the requirements as stated in section 8.2.1 of the IEEE PSPB Operations Manual, including provisions covering originality, authorship, author responsibilities and author misconduct. More information on IEEE’s publishing policies may be found at http://www.ieee.org/publications_standards/publications/rights/pub_tools_policies.html. Authors are advised especially of IEEE PSPB Operations Manual section 8.2.1.B12: “It is the responsibility of the authors, not the IEEE, to determine whether disclosure of their material requires the prior consent of other parties and, if so, to obtain it.” Authors are also advised of IEEE PSPB Operations Manual section 8.1.1B: “Statements and opinions given in work published by the IEEE are the expression of the authors.”

Author/Employer Rights

If you are employed and prepared the Work on a subject within the scope of your employment, the copyright in the Work belongs to your employer as a work-for-hire. In that case, the IEEE assumes that when you sign this Form, you are authorized to do so by your employer and that your employer has consented to the transfer of copyright, to the representation and warranty of publication rights, and to all other terms and conditions of this Form. If such authorization and consent has not been given to you, an authorized representative of your employer should sign this Form as the Author.

IEEE Copyright Ownership

It is the formal policy of the IEEE to own the copyrights to all copyrightable material in its technical publications and to the individual contributions contained therein, in order to protect the interests of the IEEE, its authors and their employers, and, at the same time, to facilitate the appropriate re-use of this material by others. The IEEE distributes its technical publications throughout the world and does so by various means such as hard copy,
microfiche, microfilm, and electronic media. It also abstracts and may translate its publications, and articles contained therein, for inclusion in various compendiums, collective works, databases and similar publications.

THIS FORM MUST ACCOMPANY THE SUBMISSION OF THE AUTHOR'S MANUSCRIPT.
Questions about the submission of the form or manuscript must be sent to the publication’s editor.
Please direct all questions about IEEE copyright policy to:
IEEE Intellectual Property Rights Office, copyrights@ieee.org, +1-732-562-3966 (telephone)