Rolling Riemannian Manifolds to Solve the Multi-class Classification Problem

Rui Caseiro[1], Pedro Martins[1], João F. Henriques[1], Fátima Silva Leite[1,2], and Jorge Batista[1]

[1]Institute of Systems and Robotics - University of Coimbra, Portugal
[2]Department of Mathematics - University of Coimbra, Portugal ,
{ruicaseiro, pedromartins, henriques, batista}@isr.uc.pt, fleite@mat.uc.pt

Abstract

In the past few years there has been a growing interest on geometric frameworks to learn supervised classification models on Riemannian manifolds [31, 27]. A popular framework, valid over any Riemannian manifold, was proposed in [31] for binary classification. Once moving from binary to multi-class classification this paradigm is not valid anymore, due to the spread of multiple positive classes on the manifold [27]. It is then natural to ask whether the multi-class paradigm could be extended to operate on a large class of Riemannian manifolds. We propose a mathematically well-founded classification paradigm that allows to extend the work in [31] to multi-class models, taking into account the structure of the space. The idea is to project all the data from the manifold onto an affine tangent space at a particular point. To mitigate the distortion induced by local diffeomorphisms, we introduce for the first time in the computer vision community a well-founded mathematical concept, so-called Rolling map [21, 16]. The novelty in this alternate school of thought is that the manifold will be firstly rolled (without slipping or twisting) as a rigid body, then the given data is unwrapped onto the affine tangent space, where the classification is performed.

1. Introduction

Applications in computer vision often involve the study of real world problems where the nonlinear constraints lead to data that lies on curved spaces [19, 28, 3]. When treating cases that cannot be solved within the standard Euclidean tools, it is usual to resort to some local linear approximations or to use *ad hoc* solutions. Those solutions are not always valid, which poses a challenge for several computer vision applications where data often lies in complex manifolds, namely in Riemannian manifolds i.e. a nonlinear, curved yet smooth, metric space (e.g. diffusion tensor processing [20, 1], foreground segmentation [4], object recognition/classification [2, 27, 31, 29], activity recognition, text categorization, shape analysis [28] motion/pose/epipolar segmentation, multi-body factorization [25, 6]). In order to extract all the underlying information of the data it is required to consider the Riemannian structure of the space.

Prior Work : Recently, the development of geometric frameworks to learn supervised classification models on Riemannian manifolds has been addressed in the computer vision community [31, 27]. A popular framework was derived by Tuzel *et al*. [31] for binary classification on Riemannian manifolds. This classifier is an additive model, where a set of weak learners are built by regression over the mappings of the data points on appropriate tangent planes (at the Karcher mean of the positive training points) and combined through boosting. The consideration of the negative samples in the mean computation would bias the result, since they are assumed to be spread on the manifold [31, 27]. This framework was tested to detect pedestrians in images using as descriptor a region covariance matrix [30] (Sym^+ - symmetric positive definite matrices), but the algorithm is valid over any Riemannian manifold and can be combined with several different boosting (classification) methods.

Despite of its popularity, the Tuzel's framework [31] contains an important bottleneck [27]. Learning problems on Riemannian manifolds are generally solved by flattening the manifold via local diffeomorphisms [5], i.e. the manifold is locally embedded into an Euclidean space. However, embedding the manifold using those local diffeomorphisms leads to some problems. The *exponential map* is *onto* but only *one-to-one* in a neighborhood of a point. The inverse mapping (*logarithmic map*) is uniquely defined only around a small neighborhood of that point. It is generally not possible to define global coordinates which make the

This work was supported by the Portuguese Science Foundation (FCT) under the project Differential Geometry for Computer Vision Pattern Recognition (PTDC/EEA-CRO/122812/2010) and through the grants SFRH/BD74152/2010 (Rui Caseiro), SFRH/BPD/90200/2012 (Pedro Martins), SFRH/BD75459/2010 (João F. Henriques).

1063-6919/13 $26.00 © 2013 IEEE
DOI 10.1109/CVPR.2013.13

whole manifold look like an Euclidean space. As argued by Tosato *et al*. [27] : once we try to change the paradigm from binary to multi-class classification the Tuzel's framework [31] is not valid anymore due to the spread of multiple positive classes on the manifold. From this perspective, it is natural to see efforts for solve this bottleneck. Tuzel [31] endowed the Sym^+ manifold with the well-known Affine-Invariant metric, however a thorough analysis of this space opens a new perspective. The space of Sym^+ is a special Riemannian manifold since there is another metric, called Log-Euclidean [1], which allows to overcome the above limitations. As showed in [1] the simple matrix exponential (exp) is a *diffeomorphism* from the Euclidean space of symmetric matrices to the Sym^+ space. The space of Sym^+ endowed with a Log-Euclidean metric is in fact *isomorphic* (the algebraic structure of vector space is conserved) and *isometric* (distances are conserved) with the corresponding Euclidean space of symmetric matrices [1], i.e. the Log-Euclidean framework defines a mapping where the space of Sym^+ is *isomorphic*, *diffeomorphic* and *isometric* to the associated space of symmetric matrices [1]. This mapping is precisely the simple matrix logarithm (log), which can be seen as the logarithm map at the identity [1].

By endowing the space of Sym^+ with the Log-Euclidean metric, Tosato *et al*. [27] proposed a mathematically well-founded multi-class framework designed to operate on this particular manifold (Sym^+). All the data is projected onto a unique tangent space at the identity (simple matrix logarithm), where a typical multi-class LogitBoost algorithm is applied [7]. More recently, by using the tensors (Sym^+) as features aggregator, Carreira *et al*. [2] also embedded all the Sym^+ manifold into an Euclidean space by endowing Sym^+ with the Log-Euclidean metric to perform multi-class classification using linear SVM. In this case, they proposed a novel and very efficient method to perform semantic segmentation, achieving results that outperforms the state-of-the-art being orders of magnitude faster to train and test. However, the Tosato/Carreira's paradigm [27, 2] (embed all the manifold) is not generalizable in the sense that it cannot be applied to other Riemannian manifolds due to the specificity of the mapping/metric used. It is then natural to ask whether the multi-class concept could be extended to operate on a large class of Riemannian manifolds.

Recently a new school of thought emerged [11, 12, 13, 5]. This new paradigm suggests to embed the Riemannian manifold into a Reproducing Kernel Hilbert Space (RKHS) by using Mercer kernels on Riemannian manifolds. Particulary, Hamm *et al*. [11, 12] proposed to use specific Grassmann kernels in order to embed the Grassmann manifold into a RKHS. Harandi *et al*. [13] used the Stein kernel to perform sparse coding and dictionary learning for symmetric positive definite matrices. Caseiro *et al*. [5] proposed a novel kernel-based mean shift on general Riemannian manifolds, by using a general Riemannian kernel function, i.e. *heat kernel*. However, the use of kernel-based algorithms for build classifiers on general Riemannian manifolds is not a good option. Firstly, to our knowledge the heat kernel is the unique Mercer kernel suited to general Riemannian manifolds. Secondly, the calculation of the heat kernel constitute a complex theoretical/technical problem and the computational burden is high. Finally, by using Mercer kernels to implicitly project the data from the manifold we are restricted to use kernel-based classifiers.

Contributions : To the best of our knowledge this is the first work that propose a mathematically well-founded classification paradigm that allows to extend/generalize the Tuzel's [31] and Tosato's [27] frameworks to multi-class models on general Riemannian manifolds (considering the Riemannian structure of the space). The idea is to project all the data from the manifold onto an affine tangent space at a particular point (e.g. identity) and then perform the classification there. To mitigate the distortion induced by local diffeomorphisms, we introduce for the first time in the computer vison community a well-founded mathematical concept, so-called Rolling map [16, 21]. The novelty in this alternate school of thought is that the manifold will be firstly rolled (without slip and twist) as a rigid body, then the given data is unwrapped onto the affine tangent space, where the classification is performed. For the sake of brevity the proof of concept will be done by testing with a multi-class LogitBoost algorithm [27, 7] on the Grassmann manifold [6, 25, 29, 28, 12, 16, 24]. We remark that our paradigm is also valid with others Riemannian manifolds.

2. Rolling Maps on Riemannian Manifolds

In the past few years there has been a growing interest in describing mathematically rolling motions, without slip and twist, of smooth manifolds (due to its analytic and geometric richness) [21, 15, 22, 16]. The study of these kinematic problems proved to be relevant, in part because the knowledge on how to realize such virtual movements allows to solve complicated problems on certain manifolds, by reducing them to similar problems on much simpler manifolds. For example, those rolling movements have been used with great success to compute interpolating curves and solve other optimal control problems on manifolds [15, 22, 16]. For instance, to solve interpolation problems on a manifold, a combination of unwrapping techniques via, local diffeomorphisms, and rolling motions enables to project data from the manifold to its affine tangent space at a point, solve the interpolation problem on the latter and then obtain an interpolating curve on the manifold by wrapping back while unrolling. The resulting curve is defined in explicit form, and has the advantage of being coordinate free [15, 16].

Rollings motions are rigid motions in the embedding space, subject to some *holonomic* constraints (rolling con-

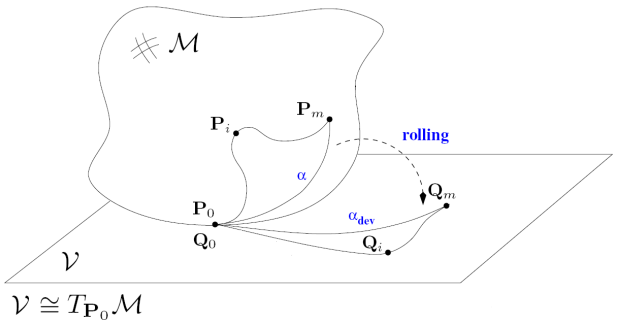

Figure 1. Rolling Map : $\mathcal{M}$ rolls upon $\bar{\mathcal{M}} = \mathcal{V} \cong T_{\mathbf{P}_0}\mathcal{M}$ without slip or twist, along a *rolling curve* $\alpha\colon [0,T] \to \mathcal{M}$ [22].

ditions) and *nonholonomic* constraints (no-slip and no-twist conditions). The most classical of all rolling motions is that of the 2-dimensional sphere rolling over the tangent plane at a point, along a curve, in which the nonholonomic constraints are satisfied by the absence of sliding and spinning and the holonomic constraints compels the sphere to stay tangent to the tangent plane during the movement.

Recalling the general definition of rolling, as in [21], this rolling motion describes how two oriented connected Riemannian manifolds $\mathcal{M}$ and $\bar{\mathcal{M}}$, having the same dimension and both embedded in the same Euclidean space $\Re^n$, roll over each other without slip and twist. Whitney's theorem [10], guarantees that a k-dimensional Riemannian manifold can be isometrically embedded into some Euclidean space $\Re^n$ for an appropriate choice of $n \geq k$.

We assume that $\bar{\mathcal{M}}$ is stationary and $\mathcal{M}$ rolls over $\bar{\mathcal{M}}$. This is a rigid motion and so can be described by the action of the special Euclidean group $\mathrm{SE}_n = \mathrm{SO}_n \ltimes \Re^n$ on $\Re^n$. The symbol $\ltimes$ represents the semi-direct product of the special orthogonal group $(\mathrm{SO}_n, (\cdot))$ by the additive group $(\Re^n, (+))$. We also assume that SO_n acts transitively on $\mathcal{M}$, that is, $\mathrm{SO}_n \circ \mathbf{P} \subset \mathcal{M}$, for any $\mathbf{P} \in \mathcal{M}$. Elements $h \in \mathrm{SE}_n$ are typically represented by pairs $h = (R, s)$, where $R \in \mathrm{SO}_n$ defines a rotation and $s \in \Re^n$ defines a translation. The action of SE_n on $\Re^n$ is usually defined by:

$$\mathrm{SE}_n \times \Re^n \to \Re^n, \qquad (h, \mathbf{P}) \mapsto h \circ \mathbf{P} = R \circ \mathbf{P} + s. \quad (1)$$

In what follows, if $\mathbf{P}$ is a point belonging to a manifold $\mathcal{M}$, $T_{\mathbf{P}}\mathcal{M}$ denotes de tangent space to the manifold $\mathcal{M}$ at the point $\mathbf{P}$ and $(T_{\mathbf{P}}\mathcal{M})^{\perp}$ denotes the normal space to $\mathcal{M}$ (with respect to the Euclidean metric) at $\mathbf{P}$. A rolling motion is described by a rolling map, which is a curve in SE_n satisfying several conditions. We give the formal definition of a rolling map, as presented in [21].

Definition 1 *A rolling map, describing how $\mathcal{M}$ rolls upon $\bar{\mathcal{M}}$, without slip or twist, along a smooth rolling curve $\alpha\colon [0,T] \to \mathcal{M}$, is a smooth map*

$$\begin{array}{rccccc} h\colon [0,T] & \to & SE_n & = & SO_n \ltimes \Re^n \\ t & \mapsto & h(t) & = & (R(t), s(t)) \end{array}, \quad (2)$$

satisfying the holonomic (rolling conditions) and nonholonomic constraints (no-slip and no-twist conditions):

- ***C1 - Rolling conditions***
  - $h(t) \circ \alpha(t) =: \alpha_{dev}(t) \in \bar{\mathcal{M}}$
  - $T_{h(t)\circ\alpha(t)}(h(t) \circ \mathcal{M}) = T_{\alpha_{dev}(t)}\bar{\mathcal{M}}$
- ***C2 - No-slip conditions***
  - $\dot{h}(t) \circ h(t)^{-1} \circ \alpha_{dev}(t) = 0$
- ***C3 - No-twist conditions***
  - *(Tangential part)* $(\dot{h}(t) \circ h(t)^{-1}) \circ T_{\alpha_{dev}(t)}\bar{\mathcal{M}} \subset (T_{\alpha_{dev}(t)}\bar{\mathcal{M}})^{\perp}$
  - *(Normal part)* $(\dot{h}(t) \circ h(t)^{-1}) \circ (T_{\alpha_{dev}(t)}\bar{\mathcal{M}})^{\perp} \subset (T_{\alpha_{dev}(t)}\bar{\mathcal{M}})$

The curve $\alpha_{\text{dev}}\colon [0,T] \to \bar{\mathcal{M}}$, defined in the first rolling condition (C1), is called the *development curve* of $\alpha(t)$ on $\bar{\mathcal{M}}$. The second rolling condition (C2) means that the tangent spaces coincide at every point of contact.

In order to understand better the nonholonomic constraints, it is necessary to define the actions appearing in the no-slip and no-twist conditions. This is well explained in [16], in terms of the action (1) of SE_n on $\Re^n$, and also included here for the sake of clarity. If $x \in \Re^n$ is a point and $\eta \in \Re^n$ is a vector, i.e., there exists a smooth curve $y \in (-\varepsilon, \varepsilon) \to \Re^n$ such that $\dot{y}(0) = \eta$, then

$$\begin{array}{rcl} \dot{h}(t) \circ x & = & \frac{\mathrm{d}}{\mathrm{d}\,\sigma}(h(\sigma) \circ x)\big|_{\sigma=t}; \\ \left(\dot{h}(t) \circ h^{-1}(t)\right) \circ x & = & \frac{\mathrm{d}}{\mathrm{d}\,\sigma}((h(\sigma) \circ h^{-1}(t)) \circ x)\big|_{\sigma=t}; \\ \left(\dot{h}(t) \circ h^{-1}(t)\right) \circ \eta & = & \frac{\mathrm{d}}{\mathrm{d}\,\sigma}((\dot{h}(t) \circ h^{-1}(t)) \circ y(\sigma))\Big|_{\sigma=0}. \end{array} \quad (3)$$

In our work we are interested in the particular case when $\bar{\mathcal{M}}$ is the affine tangent space at a point $\mathbf{P}_0 \in \mathcal{M}$ hereafter denoted by $\mathcal{V}$ to simplify notations, and $\alpha(0) = \mathbf{P}_0$. $\mathcal{V}$ is also a k-dimensional subspace embedded in $\Re^n$.

2.1. Algorithm

Let $\mathbf{X}_1, ..., \mathbf{X}_m \in \mathcal{M}$, be an input set of m samples used to train a multi-class classifier. Recall that the idea is to project all the data $\mathbf{X}_1, ..., \mathbf{X}_m$ from the manifold $\mathcal{M}$ onto the affine tangent space $\mathcal{V}$ at a particular point $\mathbf{P}_0 \in \mathcal{M}$ and then perform the classification in $\mathcal{V}$. In order to apply the rolling map, the samples $\mathbf{X}_1, ..., \mathbf{X}_m$ in $\mathcal{M}$ must be sorted (ordered) according to Algorithm 1, to obtain the sorted samples $\mathbf{P}_1, ..., \mathbf{P}_m \in \mathcal{M}$. $\mathbf{P}_0 \in \mathcal{M}$ is fixed a priori (at this point we do not care how $\mathbf{P}_0$ is selected), T is a positive real number and $0 = t_0 < t_1 < ... < t_m = T$ is a partition of the time interval $[0,T]$.

Input: $\{\mathbf{X}_1, ..., \mathbf{X}_m\} \in \mathcal{M}$, $\mathbf{P}_0 \in \mathcal{M}$, $\Delta = \frac{T}{m}$, $t_0 = 0$

for i = *1,...,m* **do**

- D_j = GeodesicDistance ($\mathbf{P}_{i-1}$,$\mathbf{X}_j$), $j = 1, ..., m$
- $k = \arg\min_j \{D_j\}$, $j = 1, ..., m$
- $t_i = i\Delta$ and $\mathbf{P}_i = \mathbf{X}_k$
- Point $\mathbf{X}_k$ will not be considered anymore

Algorithm 1:

The algorithm, proposed next, to solve the classification problem is based on rolling and unwrapping techniques which perform the projection of the data $\in \mathcal{M}$ onto $\mathcal{V} \cong T_{\mathbf{P}_0}\mathcal{M}$. This combined technique mitigates distortions that might occur when the projection is performed using local diffeomorphisms only. For each $i = 1, \cdots, m$, the point $\mathbf{P}_i$ is mapped to $\mathbf{Q}_i \in \mathcal{V}$.

Rolling and unwrapping Algorithm:

1. Compute a smooth curve $\alpha\colon [0, T] \to \mathcal{M}$ joining $\mathbf{P}_0$ (at $t = 0$) to $\mathbf{P}_m$ (at $t = T$).
2. Roll $\mathcal{M}$ over $\mathcal{V}$ along the curve $\alpha(t)$, using the corresponding rolling map $h(t)$. This generates a smooth curve $\alpha_{dev}\colon [0, T] \to \mathcal{V}$ joining the points $\mathbf{Q}_0 = \mathbf{P}_0$ and $\mathbf{Q}_m = h(t) \circ \mathbf{P}_m$
3. Use a suitable local diffeomorphism $\phi\colon \mathcal{M} \supset \Omega \to \mathcal{V}$, with $\mathbf{P}_0 \in \Omega$, between $\mathcal{M}$ and the affine tangent space $\mathcal{V}$, satisfying $\phi(\mathbf{P}_0) = \mathbf{P}_0$, $D\phi(\mathbf{P}_0) = Id$, and project the sorted data $\mathbf{P}_1, ..., \mathbf{P}_{m-1}$ to $\mathcal{V}$ while rolling along α. This combined technique of rolling and unwrapping projects the sorted data $\mathbf{P}_1, ..., \mathbf{P}_{m-1}$ onto $\mathcal{V}$ to produce $\mathbf{Q}_1, ..., \mathbf{Q}_{m-1}$, given explicitly by
$$\mathbf{Q}_i = \phi(\mathbf{A}) + \alpha_{\text{dev}}(t_i) - \mathbf{P}_0, \tag{4}$$
where $\mathbf{A} = (h(t_i) \circ \mathbf{P}_i) - \alpha_{\text{dev}}(t_i) + \mathbf{P}_0$.
4. Identify the affine tangent space $\mathcal{V}$ with $T_{\mathbf{P}_0}\mathcal{M}$, define an orthonormal coordinate system for the tangent space and compute a minimal (vectorial) representation $\mathbf{q}_1, ..., \mathbf{q}_m \in \Re^d$ of $\mathbf{Q}_1, ..., \mathbf{Q}_m$.
5. Train the classifier on the Euclidean space $T_{\mathbf{P}_0}\mathcal{M}$, using the data $\mathbf{q}_1, ..., \mathbf{q}_m$ instead of the sorted data $\mathbf{P}_1, ..., \mathbf{P}_m \in \mathcal{M}$.

Remark: The formula (4) for computing the projected points $\mathbf{Q}_i$ is well defined. In particular, in spite of the additions and subtractions, $\mathbf{A} \in \mathcal{M}$ and $\mathbf{Q}_i \in \mathcal{V}$. To understand this, one needs to use the following two facts:

(i) The rotational part of the rolling map $h = (R, s)$ acts transitively on the manifold $\mathcal{M}$, i.e., $R(t) \circ \mathcal{M} \subset \mathcal{M}$. This implies that the translational part of the rolling map is related with the development curve through $s(t) = \alpha_{\text{dev}}(t) - \mathbf{P}_0$, and $s(t) \in T_{\mathbf{P}_0}\mathcal{M}$.

(ii) $\mathcal{V}$, being the affine tangent space at $\mathbf{P}_0$, is defined as: $\mathcal{V} = \{\mathbf{P}_0 + \mathbf{W},\ \mathbf{W} \in T_{\mathbf{P}_0}\mathcal{M}\}$.

Now, using (i) and the fact that $h(t) \circ \mathbf{P} = R(t) \circ \mathbf{P} + s(t)$, for any $\mathbf{P} \in \mathcal{M}$, we can write

$$\begin{aligned} \mathbf{A} &= (h(t_i) \circ \mathbf{P}_i) - \alpha_{\text{dev}}(t_i) + \mathbf{P}_0 \\ &= (h(t_i) \circ \mathbf{P}_i) - s(t_i) = R(t_i) \circ \mathbf{P}_i \in \mathcal{M} \end{aligned}.$$

Also, due to (i) and (ii),

$$\mathbf{Q}_i = \phi(\mathbf{A}) + \alpha_{\text{dev}}(t_i) - \mathbf{P}_0 = \underbrace{\phi(\mathbf{A})}_{\in \mathcal{V}} + \underbrace{s(t_i)}_{\in T_{\mathbf{P}_0}\mathcal{M}} \in \mathcal{V}.$$

Classification: Given a test point $\mathbf{Z} \in \mathcal{M}$, the goal is to project this point onto the same tangent space $\mathcal{V} \cong T_{\mathbf{P}_0}\mathcal{M}$ where the classifier was trained. This is performed by using the Algorithm 2. The output of this algorithm is the time $t^{\mathbf{Z}}$ where the test sample $\mathbf{Z}$ belongs in the sorted set of train samples $\mathbf{P}_1, ..., \mathbf{P}_m \in \mathcal{M}$. In order to project the test point $\mathbf{Z} \in \mathcal{M}$ we only need to use Eq. 4 to obtain

$$\mathbf{Z} \mapsto \phi(\mathbf{A}) + \alpha_{\text{dev}}(t^{\mathbf{Z}}) - \mathbf{P}_0, \tag{5}$$

with $\mathbf{A} = (h(t^{\mathbf{Z}}) \circ \mathbf{Z}) - \alpha_{\text{dev}}(t^{\mathbf{Z}}) + \mathbf{P}_0$.

Input: $\{\mathbf{P}_1, ..., \mathbf{P}_m\} \in \mathcal{M}$, Test Point = $\mathbf{Z} \in \mathcal{M}$

- D_j = GeodesicDistance ($\mathbf{Z}$,$\mathbf{P}_j$), $j = 1, ..., m$
- $k = \arg\min_j \{D_j\}$, $j = 1, ..., m$
- $t^{\mathbf{Z}} = 0.5(t_k + t_{k+1})$;

Algorithm 2:

3. Rolling Maps for Grassmann Manifolds

Rolling maps for Grassmann Manifolds are presented in [16]. At first glance, it is not clear that these rolling maps are according to the general definition in previous section, but they are. What happens it that the authors of [16] took advantage of the fact that Grassmann manifolds can be represented in matrix form and have adapted the definition of rolling in order to preserve the matrix struture. This simplifies substantially all the computations. For the sake of completeness, we include in this section part of the content of [16] in which concerns rolling Grassmann manifolds on the affine tangent space at a point, in particular the kinematic equations for this rolling motions, derived from the holonomic and nonholonomic constraints.

Let $\mathcal{G}_{k,n}$ (short notation = $\mathcal{G}$) represent the Grassmann manifold of all k-dimensional subspaces of $\Re^n$. Considering that a k-dimensional subspace in $\Re^n$ can be uniquely

associated with an $(n \times n)$ orthogonal projection matrix $P = P^T$ of rank k, the Grassmann manifold is defined as a particular subset of the symmetric matrices Sym_n:

$$\mathcal{G}_{k,n} := \{P \in \text{Sym}_n \mid P^2 = P,\ \text{rank}(P) = k\}. \quad (6)$$

In this representation, $\mathcal{G}_{k,n}$ and the affine tangent space at a point are considered embedded in Sym_n, equipped with the metric induced by the Frobenius norm (this is the Euclidean metric for matrices). In this case the *rolling motion* is described by the action of the group $G = \text{SO}_n \ltimes \text{Sym}_n$ on $\mathbf{S} \in \text{Sym}_n$ by the rule

$$G \times \text{Sym}_n \to \text{Sym}_n,\ \ ((\Theta, X), \mathbf{S}) \mapsto \Theta \mathbf{S} \Theta^T + X, \quad (7)$$

where $\Theta \in \text{SO}_n$ and $X \in \text{Sym}_n$. A smooth curve $\alpha\colon [0, T]$ in $\mathcal{G}$, with $\alpha(0) = \mathbf{P}_0$, is given as

$$t \to \alpha(t) = \Theta(t) \mathbf{P}_0 \Theta^T(t), \quad (8)$$

for some $\Theta(t) \in \text{SO}_n$ satisfying $\Theta(0) = I$ (I the identity matrix). The goal now is to determine conditions on the mapping

$$h\colon [0, T] \to \text{SO}_n \ltimes \text{Sym}_n,\ \ t \to h(t) = (\Theta^T(t), X(t)) \quad (9)$$

so that it is a rolling map of $\mathcal{G}$ over the affine tangent space at a point $\mathbf{P}_0 \in \mathcal{G}$, along the curve $\alpha(t)$, and development curve $\alpha_{\text{dev}}(t) = h(t) \circ \alpha(t)$ given by

$$\alpha_{\text{dev}}(t) = \Theta^T(t) \alpha(t) \Theta(t) + X(t) = \mathbf{P}_0 + X(t). \quad (10)$$

For simplicity, assume that the base point $\mathbf{P}_0$ is

$$\mathbf{P}_0 = \begin{bmatrix} I_k & 0 \\ 0 & 0 \end{bmatrix}. \quad (11)$$

The kinematic equations for rolling having any other $\mathbf{P} \in \mathcal{G}$ as base point, can be easily obtained from this particular case due to the fact that $\mathbf{P} = \Theta \mathbf{P}_0 \Theta^T$, for some $\Theta \in \text{SO}_n$. So, even if the base point is chosen to satisfy a certain critera, all the mathematics involving rolling can be easily recalculated [16]. The kinematic equations when the base point is $\mathbf{P}_0$ are:

$$\dot{X}(t) = \begin{bmatrix} 0 & \Psi(t) \\ \Psi^T(t) & 0 \end{bmatrix}, \quad (12)$$

$$\dot{\Theta}(t) = \Theta(t) \begin{bmatrix} 0 & -\Psi(t) \\ \Psi^T(t) & 0 \end{bmatrix}, \quad (13)$$

where $\Theta(0) = I, X(0) = 0$. The (matrix) function $\Psi\colon \Re \to \Re^{k \times (n-k)}$ plays the role of a control function. Choosing Ψ is equivalent to choosing the rolling curve. When $\Psi(t) = \Psi$ is constant, the kinematic equations may be solved explicitly as:

$$\Theta(t) = \exp\left(t \begin{bmatrix} 0 & -\Psi \\ \Psi^T & 0 \end{bmatrix} \right), \quad (14)$$

$$X(t) = t \begin{bmatrix} 0 & \Psi \\ \Psi^T & 0 \end{bmatrix}, \quad (15)$$

and, in this case, the rolling curve is a geodesic in $\mathcal{G}$ with development being a geodesic in the affine tangent space. Fortunately, there exist an explicit formula for the exponential of matrices with the special block structure in (18), and Θ can be given by

$$\Theta(t) = \begin{bmatrix} (I_k - BB^T)^{1/2} & -B \\ B^T & (I_{n-k} - B^T B)^{1/2} \end{bmatrix}, \quad (16)$$

where

$$B := \Psi \frac{\sin(\Psi^T \Psi)^{1/2}}{(\Psi^T \Psi)^{1/2}} \quad (17)$$

is represented by a series expansion. Using this special representation of $\mathcal{G}_{k,n}$ as projection matrices and this particular value for $\mathbf{P}_0$, a typical element $\xi \in T_{\mathbf{P}_0}\mathcal{G}$ is defined as

$$\xi = \begin{bmatrix} 0 & Y \\ Y^T & 0 \end{bmatrix} \quad (18)$$

where Y is any real $k \times (n-k)$ matrix. Since the $T_{\mathbf{P}_0}\mathcal{G}$ is the space of symmetric matrices, there are only $d = n(n+1)/2$ independent coefficients. This minimal representation defines an orthonormal coordinate system for $T_{\mathbf{P}_0}\mathcal{G}$.

In order to implement the rolling and unwrapping algorithm presented in the previous section, we now need to determine the matrix Ψ, given the initial $\alpha(0) = \mathbf{P}_0$ and final $\alpha(T) = \mathbf{P}_m$ points of the geodesic curve defined by $\alpha(T) = \mathbf{P}_m = \Theta(T) \mathbf{P}_0 \Theta^T(T)$. This is not a trivial task since in general is not possible to invert the matrices $\mathbf{P}_0$ and $\mathbf{P}_m$. However, Srivastava *et al.* [24] defined a simple solution to this problem (we refer to [24] for more details).

4. Multi-Class Classification on $\mathcal{G}$

Let $\mathcal{S} = \{\mathbf{P}_i, y_i\}_{i=1,...,m}$ be an input set of m samples to train a multi-class classifier (C classes), with $\mathbf{P}_i \in \mathcal{G}$ and label $y_i \in \{1, ..., C\}$. All the samples of $\mathcal{S}$ are projected from the manifold onto an Euclidean space, i.e. affine tangent space $\mathcal{V} \cong T_{\mathbf{P}_0}\mathcal{G}$ using the Rolling map ($\mathbf{P}_0$ = identity). The mapped set is defined as $\mathcal{S}_T = \{\mathbf{q}_i, y_i\}_{i=1,...,m}$, with $\mathbf{q}_i \in \Re^d$. Given the set $\mathcal{S}_T \in \mathcal{V}$, the classification objective is to define a function $F(\mathbf{q}_i)\colon \mathcal{V} \mapsto \{1, ..., C\}$ [27]

$$F(\mathbf{q}_i) = \arg\max_j \{F_j(\mathbf{q}_i)\},\ \ j = 1, ..., C \quad (19)$$

The strong classifier F_j (*single-class*) can be defined as a sum of L class-specific weak learners $\{f_{lj}\}_{l=1,...,L}$, i.e. $F_j(\mathbf{q}_i) = \sum f_{lj}(\mathbf{q}_i)$, $l = 1, ..., L$. Following the Tosato's work [27] we use a standard multi-class LogitBoost algorithm [7] to learn these weak classifiers. Basically, the weak learners are used to define the inter-class decision boundaries. In this case, the multi-class weak learners

		τ_1			τ_2		
$\mathcal{G}$		**Performance (%)**			**Performance (%)**		
n	k	LogI	LogM	RmI	LogI	LogM	RmI
$\mathcal{C}$ = 3 | $\mathcal{P}$ = 400							
10	4	77.87	81.53	85.04	79.33	80.48	83.34
20	5	79.27	84.02	92.86	81.19	83.80	93.77
30	6	75.91	79.87	97.02	76.71	78.27	99.31
40	7	76.73	83.20	94.82	77.80	82.71	96.38
$\mathcal{C}$ = 3 | $\mathcal{P}$ = 500							
10	4	74.66	86.45	97.00	74.71	86.71	98.96
20	5	79.12	81.28	96.31	80.66	79.82	98.04
30	6	76.85	81.17	94.18	77.63	79.66	95.19
40	7	81.07	86.49	94.51	83.26	86.75	95.63
$\mathcal{C}$ = 4 | $\mathcal{P}$ = 400							
10	4	73.51	83.18	93.31	74.18	83.34	95.03
20	5	78.20	78.34	93.13	80.43	76.89	94.79
30	6	72.87	80.50	89.41	73.33	79.77	89.83
40	7	75.08	76.03	97.11	76.28	73.81	99.10
$\mathcal{C}$ = 4 | $\mathcal{P}$ = 500							
10	4	72.87	71.29	87.42	73.66	67.82	87.51
20	5	74.94	81.24	91.02	76.41	81.09	92.32
30	6	63.85	80.25	88.39	61.63	79.77	88.81
40	7	69.91	78.77	90.53	69.71	77.80	91.66
Average		75.16	80.85	92.62	76.05	79.90	93.72

Table 1. Rates (%) : Synthetic Data

$f_{lj} : \mathcal{V} \mapsto \Re$ are built from the combination of binary weak learners $g_{lj} : \mathcal{V} \mapsto \Re$ that solve a binary problem, one class against the others. The binary weak classifiers $g_{lj} : \mathcal{V} \mapsto \Re$ are learned by solving a weighted regression problem and combined as follows [27]

$$f_{lj}(\mathbf{q}_i) = \frac{C-1}{C}\left(g_{lj}(\mathbf{q}_i) - \frac{1}{C}\sum_{k=1}^{C} g_{lk}(\mathbf{q}_i)\right) \qquad (20)$$

For more details, please refer to [27, 7]. This algorithm is a great example of the Rolling map importance in the multi-class classification problem on general Riemannian manifolds. As argued by Tosato [27] in the case of Sym^+ : the operations referred above are only valid due to the fact that all $g_{lk}(\cdot)$ functions live in the same Euclidean tangent space. If the binary classification had been solved projecting each class onto a different space, similarly to [31], the association of the outputs would have been much more complex and obscure [27]. The tangent space given by the Rolling map constitutes a more appropriate solution for general Riemannian manifolds than the simple local diffeomorphisms.

5. Experimental Results

The proposed paradigm was evaluated on synthetic (Sec. 5.1) and on real data (Secs. 5.2 and 5.3), using a multi-class LogitBoost algorithm [27, 7] on the Grassmann manifold. Specifically, we compare the projection of the data using the Rolling map (**RmI**) versus two typical naive approaches, in which the projection is done using the standard logarithmic map (i.e. local diffeomorphism) at the identity (**LogI**) and at the Karcher mean of all samples (**LogM**). This evaluation, serve mainly as a proof of concept, which is reasonable given the novelty of the idea. The main goal is to prove that by considering the global structure of the space, the classification results are improved, i.e. the Rolling improves the

				τ_1			τ_2		
$\mathcal{G}$				**Performance (%)**			**Performance (%)**		
Scales	Bins	υ_2	$2s$	LogI	LogM	RmI	LogI	LogM	RmI
$\mathcal{C}$ = 3 | $\mathcal{P}$ = 300									
2	32	32	4	73.17	78.33	89.30	74.77	77.73	90.50
2	48	48	4	70.32	80.19	89.04	71.22	80.05	90.16
3	32	32	6	70.76	75.71	92.36	71.77	74.45	94.31
$\mathcal{C}$ = 3 | $\mathcal{P}$ = 350									
2	32	32	4	67.96	82.26	88.22	68.01	82.39	88.90
2	48	48	4	73.91	82.77	95.31	75.45	83.02	97.76
3	32	32	6	70.88	75.88	93.34	71.67	74.41	95.29
$\mathcal{C}$ = 4 | $\mathcal{P}$ = 300									
2	32	32	4	66.58	73.65	85.68	67.04	72.37	86.47
2	48	48	4	71.02	79.19	91.93	72.59	79.31	94.28
3	32	32	6	57.47	76.26	84.83	55.65	75.64	85.41
$\mathcal{C}$ = 4 | $\mathcal{P}$ = 350									
2	32	32	4	68.53	74.83	88.17	69.72	74.10	89.84
2	48	48	4	72.65	68.87	83.21	74.88	66.65	83.64
3	32	32	6	66.20	75.91	93.48	66.82	75.45	96.47
Average				69.12	76.98	89.57	69.96	76.29	91.08

Table 2. Rates (%) : Real Data - Histograms (ETH-80)

results by reducing the distortion induced by local diffeomorphisms. Let $\mathcal{C}$ be the number of classes, $\mathcal{P}$ be the number of points per class and τ is a stopping rule of minimal number of observations as in [27].

5.1. Simulations on Synthetic Data

We generate synthetic matrices on the Grassmann manifold using the method presented in [6]. An orthogonal $n \times n$ matrix $\mathcal{S}$ can be defined as a product of $(0.5n^2 - 0.5n)$ orthogonal matrices of the form $R_n^{\nu}(\theta)$, for $1 \leq \nu \leq n-1$ (refer to [6]). The orthogonal matrix $\mathcal{S}$ can be defined as $\mathcal{S} = \prod_{\nu=1}^{n-1} \mathcal{S}_n^{\nu}$, with $\mathcal{S}_n^{\nu} = \prod_{j=\nu}^{n-1} R_n^j(\theta_{\nu,j})$. For each one of the $\mathcal{C}$ classes, we generated $(0.5n^2 - 0.5n)$ angles $\{\theta_{\nu,j}\}$. Each angle $\{\theta_{\nu,j}\}$ is randomly drawn from one of $(0.5n^2 - 0.5n)$ bins in $[0, \pi]$ [6]. Then the angles are corrupted with random noise. Given the set of matrices $\{\mathcal{S}\}$ for each class, we form the matrices $\mathbf{X}$ by taking the first k orthonormal columns of each $\mathcal{S}$. The projection matrices are computed as $\mathbf{P} = \mathbf{X}\mathbf{X}^T$ [6].

5.2. Real Data - Histograms

In our real tests using histograms we select two well-known datasets, ETH-80 [18] and CIFAR-10 [17], typically used in visual object categorization. In order to extract features from an image and convert in Grassmann points, we follow the method presented in [6]. The features were obtained extracting the magnitude of the image gradient and the Laplacian at three different scales. A maximum of five different scales and two types of histograms were used, that is, $\upsilon_1 = \{1, 2, 4, 6, 8\}$ where υ_1 is the variance of the Gaussian filter and $\upsilon_2 = \{32, 48\}$ where υ_2 is the number of bins of the histogram [6]. Let s be the number of scales used, for each of the $2s$ images a υ_2-bin histogram. Each histogram is normalized such that the l_2-norm is equal to 1 [6]. We form a feature matrix by stacking the $2s$ aforementioned υ_2-bin histograms as columns and then we take the SVD (Singular Value Decomposition) of the resulting $\upsilon_2 \times 2s$ matrix. Its singular vectors span a subspace of dimension $2s$ in $\Re^{\upsilon_2}$.

$\mathcal{G}$				τ_1 Performance (%)			τ_2 Performance (%)		
Scales	Bins	v_2	$2s$	LogI	LogM	RmI	LogI	LogM	RmI
$\mathcal{C}$ = 3 | $\mathcal{P}$ = 400									
2	32	32	4	65.92	70.33	81.30	68.13	70.18	82.57
2	48	48	4	63.07	72.19	81.04	63.86	72.97	82.17
3	32	32	6	63.51	67.71	84.36	64.52	66.26	87.15
$\mathcal{C}$ = 3 | $\mathcal{P}$ = 500									
2	32	32	4	60.71	74.26	80.22	59.82	75.57	80.45
2	48	48	4	66.66	74.77	87.31	68.74	76.34	91.09
3	32	32	6	63.63	67.88	85.34	64.20	66.01	88.12
$\mathcal{C}$ = 4 | $\mathcal{P}$ = 400									
2	32	32	4	59.33	65.65	77.68	59.25	64.16	78.14
2	48	48	4	63.77	71.19	83.93	65.91	72.48	87.52
3	32	32	6	50.22	68.26	76.83	45.58	68.08	76.86
$\mathcal{C}$ = 4 | $\mathcal{P}$ = 500									
2	32	32	4	61.28	66.83	80.17	62.67	66.43	82.38
2	48	48	4	65.40	60.87	75.21	68.85	57.50	74.94
3	32	32	6	58.95	67.91	85.48	59.18	68.05	90.34
Average				61.87	68.98	81.57	62.55	68.66	83.47

Table 3. Rates (%) : Real Data - Histograms (CIFAR-10)

5.3. Real Data - Illumination/Pose Invariance

Following the Hamm's work [12, 11] in this section we test with subspace representations using the Yale-Face [8], CMU-PIE [23] and ETH80 [18] image datasets. In particularly, these datasets are suited to use with subspace-based learning techniques since they contain naturally factorized structures : illumination-invariant face recognition for the Yale-Face / CMU-PIE datasets and pose-invariant object categorization for the ETH-80 dataset. In the Yale-Face/CMU-PIE databases [8, 23], a set comprises images of all illumination conditions of a person at a fixed pose. For the ETH-80 database [18], a set comprises images of all possible poses of an object from a category. By considering each set as a point in the Grassmann manifold, it is possible realize illumination-invariant and pose-invariant learning tasks, respectively [12, 11]. Following the works in [12, 11] we use as image features the pixel intensities (normalised) and we compute subspaces using SVD. Let N be the number of pixels in an image. Following the idea presented in [27], each image was divided in patches of 5×5 pixels (with some degree of overlapping) and it was learned one classifier for each patch. Let $N_p = 25$ be the number of pixels in a patch and N_s be the number of elements in each set (illumination/pose sets).

Let D be the $N_p \times N_s$ data matrix containing all illuminations/poses of each set and patch. The k-dimensional subspace of $\Re^{N_p}$ is computed from the SVD of the matrix D. In all these tests were used 4 classes $\mathcal{C} = 4$ and extra training samples were generated using bootstrapping.

$\mathcal{G}$		τ_1 Performance (%)			τ_2 Performance (%)		
N	k	LogI	LogM	RmI	LogI	LogM	RmI
504	2	59.87	63.53	75.04	61.56	62.48	76.93
504	4	61.27	66.02	77.86	63.65	66.21	78.66
504	6	57.91	61.87	82.02	58.61	60.00	84.90
504	8	58.73	65.20	79.82	59.84	64.99	81.60
896	2	58.66	70.45	84.00	58.24	71.37	86.37
896	4	63.12	65.28	83.31	64.93	63.61	85.34
896	6	60.85	65.17	81.18	61.53	63.44	82.14
896	8	65.07	70.49	81.51	67.85	71.42	82.63
Average		60.68	66.00	80.59	62.02	65.44	82.32

Table 4. Rates (%) : Real Data - Ilumination (Yale-Face)

$\mathcal{G}$		τ_1 Performance (%)			τ_2 Performance (%)		
N	k	LogI	LogM	RmI	LogI	LogM	RmI
504	2	54.87	58.53	65.04	56.79	57.48	66.32
504	4	56.27	61.02	72.86	59.11	61.63	73.55
504	6	52.91	56.87	77.02	53.51	54.73	80.49
504	8	53.73	60.20	74.82	54.88	60.28	76.82
896	2	53.66	65.45	79.00	52.76	67.02	81.79
896	4	58.12	60.28	78.31	60.20	58.41	80.64
896	6	55.85	60.17	76.18	56.42	58.22	77.09
896	8	60.07	65.49	76.51	63.45	67.08	77.64
Average		55.68	61.00	74.96	57.14	60.60	76.79

Table 5. Rates (%) : Real Data - Ilumination (CMU-PIE)

Illumination-Invariance - Yale-Face DataSet : Using a special camera rig, it is feasible to obtain multi-view, multi-lighting images simultaneously [12, 11]. This is the case of the Yale-Face and the Extended Yale-Face datasets [8]. These datasets together comprise pictures of 38 subjects with 9 different poses and 45 different lighting conditions. The original images are gray-valued and contain background objects. Following [12, 11], face regions were cropped/aligned from the original pictures, by manually select some key points (mouth, nose tip and center of eyes) for each image. In order to obtain two different values for N, after the cropped, the images were resized to 24×21 pixels ($N = 504$) and to 32×28 pixels ($N = 896$). Finally, they were normalized to have the same variance. All the different 45 illumination conditions of a person under a fixed pose were used to calculate subspaces ($N_s = 45$). The illumination is modeled by subspaces of size $k = \{2, 4, 6, 8\}$.

Illumination-Invariance - CMU-PIE DataSet : The CMU-PIE database [23] is also an example of a multi-view, multi-lighting face dataset obtained with a camera rig [11]. This dataset comprises images from 68 subjects under 13 different poses and 43 different lighting conditions. The original images are color-valued and contain background objects. Following [11], the images were converted to gray and face regions were cropped/aligned from the original pictures, by manually select some key points (mouth, nose tip and center of eyes) for each image.

In order to obtain two different values for N, after the cropped, the images were resized to 24×21 pixels ($N = 504$) and to 32×28 pixels ($N = 896$) and normalized to have the same variance. All the different 43 illumination conditions of a person under a fixed pose were used to calculate subspaces ($N_s = 43$). The illumination is modeled

$\mathcal{G}$		τ_1 Performance (%)			τ_2 Performance (%)		
N	k	LogI	LogM	RoI	LogI	LogM	RoI
576	2	67.37	69.03	77.54	69.29	67.98	75.02
576	4	68.77	71.52	85.36	71.61	72.13	86.05
576	6	65.41	67.37	89.52	66.01	65.23	92.99
576	8	66.23	70.70	87.32	67.38	70.78	89.32
1024	2	66.16	75.95	91.50	65.26	77.52	94.29
1024	4	70.62	70.78	90.81	72.70	68.91	93.14
1024	6	68.35	70.67	88.68	68.92	68.72	89.59
1024	8	72.57	75.99	89.01	75.95	77.58	90.14
Average		68.18	71.50	87.46	69.64	71.1	88.81

Table 6. Rates (%) : Real Data - Pose (ETH-80)

by subspaces of size $k = \{2, 4, 6, 8\}$.

Pose-Invariance - ETH-80 DataSet : The ETH-80 dataset [18] was projected for testing techniques of object categorization under varying poses [12, 11]. This dataset comprises pictures of 8 object categories and 10 object instances for each category, captured under 41 different poses (orientations). The original images are color-valued without background objects. Following [12, 11], the images were converted to gray and the background was kept. In order to obtain two different values for N, after the cropped, the images were resized to 24×24 pixels ($N = 576$) and to 32×32 pixels ($N = 1024$) and normalized to have the same variance. All the different 41 poses of an object from a category were used to calculate subspaces ($N_s = 41$). The illumination is modeled by subspaces of size $k = \{2, 4, 6, 8\}$.

6. Conclusions and Future Work

To the best of our knowledge, this is the first work that propose an avenue that allows to generalize multi-class models to general Riemannian manifolds. We introduce for the first time in the vision community the Rolling map paradigm. This map allows to solve some complex problems on manifolds, by simplifying them to simpler ones.

In the future we intend to explore other possible paradigms to perform multi-class classification on the Riemannian manifolds, namely nonparametric regression estimators with manifold-valued input [14].

We believe that Rolling motions could be useful to solve several other problems frequently occurring in vision. The specific Rolling map on the Grassmann manifold could be used to devise view-invariant techniques to analyse facial human expressions [26]. Recently, the Grassmann manifold has been also used with sucess in the domain adaptation [9]. In this case the Rolling map combined with interpolation techniques can be a promising solution to explore more complex curves on that manifold and generalize to linear classifiers. Interpolation on the Essential manifold, on the Fundamental manifold, or on their generalization to higher tensors is certainly useful. It would be interesting to devise techniques that allow for the real-time computation of additional virtual camera views.

References

[1] V. Arsigny, P. Fillard, X. Pennec, and N. Ayache. Geometric means in a vector space structure on symmetric positive-definite matrices. *Journal Matrix Analysis Applications*, 29(1):328–347, 2007. 1, 2

[2] J. Carreira, R. Caseiro, J. Batista, and C. Sminchisescu. Semantic segmentation with second-order pooling. In *ECCV*, 2012. 1, 2

[3] R. Caseiro, J. F. Henriques, and J. Batista. Using directional statistics to learn cast shadows from a multi-spectral light sources physical model. In *ICIP*, 2010. 1

[4] R. Caseiro, J. F. Henriques, P. Martins, and J. Batista. A nonparametric riemannian framework on tensor field application to foreground segmentation. In *ICCV*, 2011. 1

[5] R. Caseiro, J. F. Henriques, P. Martins, and J. Batista. Semi-intrinsic mean shift on riemannian manifolds. In *ECCV*, 2012. 1, 2

[6] H. Cetingul and R. Vidal. Intrinsic mean shift for clustering on stiefel and grassmann manifolds. In *CVPR*, 2009. 1, 2, 6

[7] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: A statistical view of boosting. *The Annals of Statistics*, 28(2):337–407, 2000. 2, 5, 6

[8] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *TPAMI*, 23(6):643–660, 2001. 7

[9] B. Gong, Y. Shi, F. Sha, and K. Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *CVPR*, 2012. 8

[10] V. Guillemin and A. Pollack. *Differential Topology*. Prentice-Hall, 1974. 3

[11] J. Hamm. *Subspace-based Learning with Grassmann Kernels*. Thesis PhD - University Pennsylvania, 2008. 2, 7, 8

[12] J. Hamm and D. Lee. Grassmann discriminant analysis : a unifying view on subspace-based learning. In *ICML*, 2008. 2, 7, 8

[13] M. T. Harandi, C. Sanderson, R. Hartley, and B. C. Lovell. Sparse coding and dictionary learning for symmetric positive definite matrices: A kernel approach. In *ECCV*, 2012. 2

[14] M. Hein. Robust nonparametric regression with metric-space valued output. In *NIPS*, 2009. 8

[15] K. Hupper and F. S. Leite. Smoothing interpolation curves on manifolds with applications to path planning. In *Mediterranean Conference on Control and Automation*, 2002. 2

[16] K. Hupper and F. S. Leite. On the geometry of rolling and interpolation curves on Sn, SOn, Grassmann manifolds. *J. Dynamical Control Systems*, 13(4):467–502, 2007. 1, 2, 3, 4, 5

[17] A. Krizhevsky. Learning multiple features from tiny images. In *Technical Report*, 2009. 6

[18] B. Leibe and B. Schiele. Analyzing appearance and contour based methods object categorization. In *CVPR*, 2003. 6, 7, 8

[19] Y. M. Lui. Advances in matrix manifolds for computer vision. *Image Vision Computing*, 30(6-7):380–388, 2012. 1

[20] X. Pennec. *Statistical Computing on Manifolds for Computational Anatomy*. Habilitation à Diriger Recherche, 2006. 1

[21] R. Sharpe. *Differential Geometry*. Springer, 1996. 1, 2, 3

[22] Y. Shen, K. Hupper, and F. S. Leite. Smooth interpolation of orientation by rolling and wrapping for robot motion planning. In *IEEE Int. Conf. Robotics Automation*, 2006. 2, 3

[23] T. Sim, S. Baker, and M. Bsat. Cmu pose, illumination, and expression database. *TPAMI*, 25(12):1615–1618, 2003. 7

[24] A. Srivastava and E. Klassen. Bayesian and geometric subspace tracking. *Advances Applied Probability*, 36(1):43–56, 2004. 2, 5

[25] R. Subbarao and P. Meer. Nonlinear mean shift over riemannian manifolds. *IJCV*, 84(1):1–20, 2009. 1, 2

[26] S. Taheri, P. Turaga, and R. Chellappa. Towards view-invariant expression analysis using analytic shape manifolds. In *FG*, 2011. 8

[27] D. Tosato, M. Farenzena, M. Cristani, M. Spera, and V. Murino. Multi-class classification on riemannian manifolds for video surveillance. In *ECCV*, 2010. 1, 2, 5, 6, 7

[28] P. Turaga, A. Veeraraghavan, A. Srivastava, and R. Chellappa. Statistical analysis on manifolds and its applications to video analysis. *in Video Search and Mining, Studies in Computational Intelligence, Springer-Verlag*, 2010. 1, 2

[29] P. Turaga, A. Veeraraghavan, A. Srivastava, and R. Chellappa. Statistical computations on grassmann and stiefel manifolds for image and video-based recognition. *TPAMI*, 33(11):2273–2286, 2011. 1, 2

[30] O. Tuzel, F. Porikli, and P. Meer. Region covariance: A fast descriptor for detection and classification. In *ECCV*, 2006. 1

[31] O. Tuzel, F. Porikli, and P. Meer. Pedestrian detection via classification on riemannian manifolds. *TPAMI*, 30(10):1713–1727, 2008. 1, 2, 6