Nearest-Neighbor Search Algorithms on Non-Euclidean Manifolds for Computer Vision Applications

Pavan Turaga * and Rama Chellappa
Center for Automation Resarch
University of Maryland
Institute for Advanced Computer Studies
College Park, MD 20742
{pturaga,rama}@umiacs.umd.edu

ABSTRACT

Nearest-neighbor searching is a crucial component in many computer vision applications such as face recognition, object recognition, texture classification, and activity recognition. When large databases are involved in these applications, it is also important to perform these searches in a fast manner. Depending on the problem at hand, nearest neighbor strategies need to be devised over feature and model spaces which in many cases are not Euclidean in nature. Thus, metrics that are tuned to the geometry of this space are required which are also known as geodesics. In this paper, we address the problem of fast nearest neighbor searching in non-Euclidean spaces, where in addition to dealing with the large size of the dataset, the significant computational load involves geodesic computations. We study the applicability of the various classes of nearest neighbor algorithms toward this end. Exact nearest neighbor methods that rely solely on the existence of a metric can be extended, albeit with a huge computational cost. We derive an approximate method of searching via approximate embeddings using the logarithmic map. We study the error incurred in such an embedding and show that it performs well in real experiments.

Keywords

Nearest-neighbor, Manifold, Hashing, Shapes, Grassmann manifold, Region Covariance.

1. INTRODUCTION

Large databases of images and videos are becoming more and more common-place since the growth of personal collections and Internet archives. To make these datasets more easily accessible to users, it is important to develop methods that allow fast retrieval and access to information. In many applications such as content-based image retrieval, texture classification, biometrics, and video mining, the problem is frequently reduced to searching for exemplars similar to a query in a large database. This is more formally known as the similarity search or the nearest-neighbor problem. The problem of nearest-neighbor searching has been studied for many years in the database and algorithms communities involving searching in a n-dimensional Euclidean space.

In the computer vision literature, over the past several years there has been significant research about what are good features and models for representing images and videos. Images and videos are usually represented by a concise set of features or models – such as shape [22], color/intensity histograms [17], SIFT [26], histogram of gradients [10], linear dynamic systems [33], covariance matrices [41] etc. Many of these features and models, however do not live in a Euclidean space. What this means is that the underlying distance function on the space is not the usual L_2/L_p norm but a highly non-linear function which is also in most cases computationally hard to compute. As an example, shape spaces have long been considered as Riemannian manifolds – Kendall's shape space is a complex spherical manifold [22], affine shape spaces are Grassmann manifolds [6]. The space of $d \times d$ covariance matrices or tensors is a product manifold of the special orthogonal group and the diagonal group, thus is not a vector space [30, 41]. The space of linear subspaces is a quotient space of the orthogonal group resulting in the Grassmann manifold [18]. Linear dynamic systems can be identified by the column-space of the observability matrix which is also a Grassmann manifold [39]. Even the commonly used histograms form a simplex in $\mathbb{R}^n$ and do not live in a vector space. Over the years, many advances have been made to understand the geometric properties of these varied spaces, and they have been utilized to devise more accurate inference and classification algorithms [14, 36, 40, 39]. Understanding the geometry allows one to define distances, leading to geodesics etc on these manifolds.

In this paper, we study the problem of fast nearest neighbor searching for data which lies in non-Euclidean manifolds. We provide a formal treatment of this problem from which various algorithms can be derived. At first glance, exact methods that rely solely on distance functions can be easily extended to address this problem - however with a huge pre-processing cost. The significant load in indexing with exact methods arises from the complexity of computing the distance function, and the complexity of computing centroids. These issues are generally not a concern in Euclidean spaces, but they become significant sources of complexity in manifolds. Thus, a straightforward application of these methods is not a feasible solution. Thus, we seek methods that mitigate the complexity involved in: 1) geodesic distance computations, and 2) centroid computations. In general, there is very little that can be done to reduce the complexity of these operations, except try to find efficient computations on a case-by-case basis for each manifold.

As an alternative, in this paper we explore the use of tangent

*Corresponding author

ICVGIP '10, December 12-15, 2010, Chennai, India

plane unwrapping of the manifold into a vector-space and using the L_2 norm on the tangent plane for indexing. This seemingly straightforward approach addresses both the concerns enumerated above. However, the resulting procedure is infact only approximate even when using the 'exact' methods, since we are replacing the geodesic distance with a faster approximate metric on the tangent plane. Thus, if pre-processing cost is not a concern, then one can apply a variety of exact algorithms to manifolds by treating them simply as metric spaces with appropriate definitions of distances and centroids. But, if pre-processing cost is a significant concern or if the application only mandates approximate solutions, then we suggest tangent-plane projections of the data - followed by any standard searching technique - however any technique employed in this manner will only be approximate. The methods make use of concepts of Riemannian geometry and are applicable to any general Riemannian manifold. We then show specific examples in image and video analysis to illustrate the utility of these methods.

Related Work: At the time of submission to this conference, we were not aware of other attempts at studying indexing for Riemannian manifolds. However, during the review period a paper by Chaudhry and Ivanov with the same goals as ours came to our attention, which was presented at ECCV in September 2010 [8]. Both the papers propose the same general idea of using logarithmic embeddings for this problem. Chaudhry and Ivanov also present extensions to cases where logarithmic maps are not known in closed form, which is an important case not considered in this paper. However in this paper, we also study the errors incurred under such embeddings which is not considered by [8]. The similarity in framework with different extensions make these two papers complementary to each other. In addition to these attempts, the work on approximate nearest subspace search [5] is an example of indexing on non-Euclidean manifolds. However, this work is limited in applicability to subspaces, moreover does not exploit the Riemannian geometric interpretation of subspaces. In the relatively better understood domain of searching in Euclidean spaces, the two most popular methods employed are - 1) exact and 2) approximate. Exact methods usually rely on space partitioning. Examples of this approach include quad-trees, and k-d trees. The resulting data structure is represented as a tree with the root node being the entire dataset, child nodes representing partitions and leaf nodes representing individual data points. We refer the reader to [24] for a comprehensive comparison of exact search methods applied to patch-based image searching. All these methods rely heavily on the assumption that points are in $\mathbb{R}^n$ with the underlying metric being the L_2 norm. The work of [12] shows how to adapt the standard k-d tree algorithm to low-dimensional manifolds whose structure is unknown. Approximate methods in Euclidean spaces became popular due to the introduction of Locality Sensitive Hashing (LSH) algorithms [20, 16]. The original LSH algorithm was proposed by Gionis et al [16] which was tuned for 'Hamming'-spaces i.e. spaces where the underlying distance function is the Hamming distance. This was extended to Euclidean spaces by [13]. A good introduction and survey of these methods can be found in [11]. All these methods can be applied if the manifold of interest can somehow be embedded into a Euclidean space. We discuss here that it is important to consider the Riemannian geometry of manifolds to more systematically address this problem. This is because there exist several analytic manifolds which cannot in any easy way be embedded into an ambient Euclidean space e.g. the space of linear subspaces or the Grassmann manifold is best treated as a quotient space of the orthogonal group and there is no easy natural embedding into an ambient vector space [34].

Nearest-neighbor searching in metric spaces when only an underlying metric is known has also been studied in literature. A number of nearest neighbor algorithms have been devised for indexing data with arbitrary distance functions such as vantage-point trees, metric trees and multi-vantage point trees. An excellent survey of such methods can be found in [9]. These are exact methods of searching in metric spaces. Unfortunately tree-based indexing methods such as these in general suffer from the curse of dimensionality. For large dimensions the performance is not significantly different than simple brute-force search. Approximate search methods in arbitrary metric spaces have also been proposed as in [3]. [20] provided algorithms for approximate searches when the underlying distance function is the L_∞ norm. In some cases the underlying non-Euclidean distance measure can be replaced with another which can be efficiently computed. Several methods have been proposed for embedding arbitrary spaces into a Euclidean or pseudo-Euclidean space [7, 44]. Embedding methods substitute a fast approximate distance for the original distance hence are approximate search methods.

Contributions: We present an analysis of indexing non-Euclidean spaces whose geometric properties are known. We do this by taking recourse to Riemannian geometry, and studying how the traditional ideas of indexing can be extended to these structured spaces. We discuss that exact methods can be deployed, however the computational load is immense. Thus, for large database applications, we present a principled method for approximate searching using tangent-space embedding of data. We also present some results on the error incurred in such embeddings. We also verify in experiments that the proposed method works very well in large scale searching problems.

2. EXAMPLES OF NON-LINEAR MANIFOLDS

We will discuss a few manifolds that frequently appear in vision applications.

1. **Shape Features:** Shapes in images are commonly described by a set of landmarks on the object being imaged. After appropriate translation, scale and rotation normalization it can be shown that shapes reside on a complex spherical manifold [22]. Further, by factoring out all possible affine transformations, it can be shown that shapes reside on a Grassmann manifold. More recently, shapes have been considered to be continuous closed planar curves. The space of such curves can also be characterized as a manifold [38].

2. **Covariance Features:** In recent years, region covariance has proved to be a popular feature to encode local shape and texture. Covariance features as region descriptors were introduced in [41] and have been successfully applied to human detection [40], object tracking [31] and texture classification [41]. Covariance matrices also appear in medical imaging literature where diffusion tensor MRI produces measurements of diffusion of water molecules, where each voxel is associated with a 3×3 symmetric positive definite matrix [30].

3. **Time Warps:** The space of positive and monotonically increasing functions mapping the unit-interval to the unit-interval are usually referred to as time-warp functions. The derivatives of warping functions can be interpreted as probability density functions. The square-root form of pdfs can then be described as a sphere in the space of functions. This was exploited in [43] to recognize human activities. Variability in sampling closed planar curves can also be modeled as a sphere in the space of functions [36].

4. **Subspaces:** In image and object recognition, recent methods have focused on utilizing multiple images of the same object, taken under varying viewpoints or varying illumination conditions, for recognition [23, 19, 2, 27]. The set of face images of the same person under varying illumination conditions is frequently modeled as a linear subspace of 9-dimensions [25]. The set of k-dimensional subspaces of $\mathbb{R}^n$ is called the Grassmann manifold. A related manifold of k-basis vectors of $\mathbb{R}^n$ or $n \times k$ tall-thin orthonormal matrices is called the Stiefel manifold and has found applications in face recognition [28]. The Stiefel manifold also finds applications in finding projections of data with desired statistical properties such as sparsity, discriminability, etc [37].

5. **Dynamic models:** In video analysis, an important task is to describe a sequence of static features using parametric models. One popular dynamical model for such time-series data is the autoregressive and moving average (ARMA) model. Examples include dynamic textures [33], human joint angle trajectories and silhouette sequences [42]. The space spanned by the columns of the observability matrix of the ARMA model can be identified as a point on the Grassmann manifold [39].

An overview of various manifolds that frequently appear in vision literature, and their associated non-linear distance functions are described in table 1. As can be seen, the distance computations (and centroid computations) can be of significant complexity for these manifolds. Thus, metric space indexing can be applied in principle, but the pre-processing and query computations can be significantly involved. The rest of the paper will be geared toward proposing a general framework for mitigating these issues.

3. PRELIMINARIES

In this section, we briefly recapitulate the mathematical preliminaries needed to study nearest-neighbor searching on non-Euclidean manifolds. A topological space $\mathcal{M}$ is called a manifold if it is *locally Euclidean* i.e. for each $p \in \mathcal{M}$, there exists an open neighborhood U of p and a mapping $\phi : U \to \mathbb{R}^n$ such that $\phi(U)$ is open in $\mathbb{R}^n$ and $\phi : U \to \phi(U)$ is a diffeomorphism. The pair (U, ϕ) is called a *coordinate chart* for the points that fall in U. Let $\mathcal{M}$ be an n-dimensional manifold and, for a point $p \in \mathcal{M}$, consider a differentiable curve $\gamma : (-\epsilon, \epsilon) \to \mathcal{M}$ such that $\gamma(0) = p$. The velocity $\dot{\gamma}(0)$ denotes the velocity of γ at p. This vector has the same dimension as the manifold and is an example of a tangent vector to $\mathcal{M}$ at p. The set of all such tangent vectors is called the tangent space to $\mathcal{M}$ at p. Even though the manifold $\mathcal{M}$ maybe nonlinear, the tangent space $T_p(\mathcal{M})$ is always linear.

Metrics via Geodesic Distances: The task of measuring distances on a manifold is accomplished using a Riemannian metric. A Riemannian metric on a differentiable manifold $\mathcal{M}$ is a map $\langle \cdot, \cdot \rangle$ that smoothly associates to each point $p \in \mathcal{M}$ a symmetric, bilinear, positive definite form on the tangent space $T_p(\mathcal{M})$. Using the Riemannian structure, it becomes possible to define lengths of paths on a manifold. For any two points $p, q \in \mathcal{M}$, one can define the distance between them as the infimum of the lengths of all smooth paths on $\mathcal{M}$ which start at p and end at q:

$$d(p,q) = \inf_{\{\alpha:[0,1]\mapsto\mathcal{M} | \alpha(0)=p, \alpha(1)=q\}} L[\alpha], \text{where} \quad (1)$$

$$L[\alpha] = \int_0^1 \sqrt{\left\langle \frac{d\alpha(t)}{dt}, \frac{d\alpha(t)}{dt} \right\rangle} dt \quad (2)$$

Tangent-plane embedding: If $\mathcal{M}$ is a Riemannian manifold and $p \in \mathcal{M}$, the exponential map $\exp_p : T_p(\mathcal{M}) \to \mathcal{M}$, is defined by $\exp_p(v) = \alpha_v(1)$ where α_v is a specific geodesic. The inverse mapping $\log_p : \mathcal{M} \to T_p$ called the inverse exponential map/logarithmic map at a 'pole', takes a point on the manifold and returns a point on the tangent space of the pole – which is a Euclidean space.

Centroids: Given a set of points on a manifold, the intrinsic mean of the dataset or the Karcher mean [21] is a natural way of generalizing the notion of a centroid. The intrinsic mean is defined as the point μ that minimizes the sum of squared-distance to all other points: $\mu = arg\min_{x \in M} \sum_{i=1}^{N} d(x, x_i)^2$. Computing the intrinsic mean is not usually possible in a closed form. The intrinsic mean is unique only for points that are close together [21]. An iterative procedure is popularly used in estimation of means of points on manifolds [29].

These concepts are illustrated graphically in figure 1.

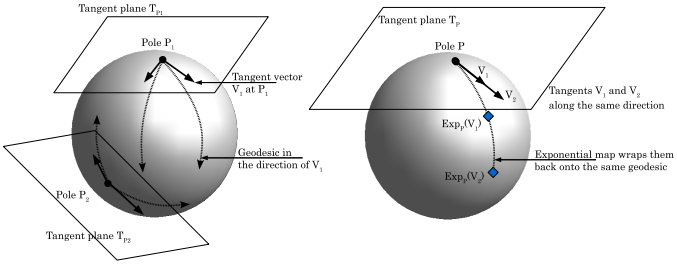


Figure 1: (Left) Figure illustrating the notions of tangent spaces, tangent vectors, and geodesics. Shown in the figure are two points P_1 and P_2 on the manifold and the tangent planes at these points. Geodesics paths are constant velocity curves on the manifold. Tangent vectors correspond to velocities of curves on the manifold. (Right) Figure illustrating the notion of exponential maps and inverse exponential maps. Shown are two points on the tangent plane at pole P. Both points lie along the same tangent vector. The exponential map will map them onto the same geodesic.

4. AN EMBEDDING APPROACH FOR SOLVING NEAREST NEIGHBORS

In this section, we discuss an embedding of points from the manifold into a Euclidean space which allows us to solve the nearest neighbor problem in the embedded space. Let us denote the embedding of points on a manifold to some vector space as $\phi : \mathcal{M} \to \mathbb{R}^n$. One can usually come up with many such mappings, which are atleast locally homeomorphic by the definition of a manifold. But, in general no such mapping exists that can map the entire manifold to $\mathbb{R}^n$ that is isometric i.e. preserves distances. We saw one example of a mapping, namely the inverse exponential map or the logarithmic map in section 3. In this section, we will study this specific method of embedding and show that it enjoys certain favorable properties that make it attractive for indexing and searching. Recall that, if $\mathcal{M}$ is a Riemannian manifold and $p \in \mathcal{M}$, the **exponential map** $\exp_p : T_p(\mathcal{M}) \to \mathcal{M}$, is defined by $\exp_p(v) = \alpha_v(1)$ where α_v is a specific geodesic. The inverse mapping $\log_p : \mathcal{M} \to T_p$ is uniquely defined only around a local neighborhood of the point p. The inverse exponential map/logarithmic map at a 'pole', takes a point on the manifold and returns a point on the tangent space of the pole – which is a Euclidean space.

Under this embedding, one can now employ the usual Euclidean norm to solve the nearest neighbor search problem. However, one needs to know the error incurred under this embedding. We discuss this issue in the following. Let us assume that we are given a database of points $X = \{x | x \in \mathcal{M}\}$, where $\mathcal{M}$ is the manifold

Manifold	Numerical Representation	Dimension	Commonly used Distance functions	Applications
Spherical manifold	n-vector with unit norm	$n-1$	$d(X_1, X_2) = \cos^{-1}(\left\|x_1^T x_2\right\|)$	Kendall's shape space [22], probability density functions [35]
Stiefel manifold	$n \times k$ orthonormal matrix	$nk - \frac{k(k+1)}{2}$	No closed analytical form.	Face recognition [28], dimensionality reduction [37]
Grassmann manifold	$n \times k$ orthonormal matrix	$k(n-k)$	$d(X_1, X_2) = \sum_i \|\theta_i\|^2$, where $\cos(\theta_i)$ are singular values of $X_1^T X_2$	Image set modeling [23, 19, 2, 27], dynamic models [33, 39]
Covariance matrices	$n \times n$ symmetric pos. def. matrix	$\frac{n(n+1)}{2}$	$d(X_1, X_2) = \sqrt{\sum_{i=1}^{n} ln^2 \lambda_i(X_1, X_2)}$, where $\{\lambda_i\}$ are generalized eigenvalues of $\lambda X_1 v = X_2 v$	Region descriptors [41, 31, 40], diffusion tensor imaging [30]

Table 1: Examples of various manifolds that appear in vision applications. The table shows the highly non-linear and sometimes computationally intensive distance functions on the manifolds. Depending on data, the dimension of the manifold can be quite high.

of interest. We shall denote a specific point $p \in \mathcal{M}$ as a 'pole' which shall be used to define the embedding. Intuitively, the pole is the analog of the 'origin' in $\mathbb{R}^n$. However, unlike $\mathbb{R}^n$ there is no point that naturally serves as the pole for manifolds. Given a dataset of points on a manifold, the intrinsic mean of the dataset or the Karcher mean [21] is a natural choice for the pole. We shall later discuss how to compute it in a fast and efficient manner.

Given a pole p, and two points $x, y \in \mathcal{M}$ we would like to relate the geodesic distance $d^2(x,y)$ to the Euclidean norm in the embedding space given by $\|Log_p(x) - Log_p(y)\|^2$. In general, there is no closed form relation one can obtain between them. For points that are close to the pole, i.e. if $0 \leq d(x,p), d(y,p) \leq \epsilon$, we have

$$\|Log_p(x) - Log_p(y)\|^2 \leq \|Log_p(x)\|^2 + \|Log_p(y)\|^2 \quad (3)$$
$$= d^2(x,p) + d^2(y,p) \leq 2\epsilon^2. \quad (4)$$

By triangle inequality, we also have that $d^2(x,y) \leq 2\epsilon^2$. Thus, for points close to the pole, we have

$$0 \leq d^2(x,y), \|Log_p(x) - Log_p(y)\|^2 \leq 2\epsilon^2. \quad (5)$$

Any more general result than this is very difficult to obtain for arbitrary manifolds. Thus, for close points, and an appropriate choice of pole p, $d^2(x,y) \approx \|Log_p(x) - Log_p(y)\|^2$. We refer the reader to [14] for an example of the exact relation between these quantities for the special case of a sphere.

For points that are far away from the pole, in general the approximation error can be large. In certain cases, as we show next, even in the case of far away points the error in approximation can be bounded with high probability. For two random points $x, y \in \mathcal{M}$, let $e(x,y) = d^2(x,y) - \|Log_p(x) - Log_p(y)\|^2$ be the approximation error due to the embedding. Then, by the triangle inequality we have

$$d^2(x,y) \leq d^2(x,p) + d^2(p,y) = \|Log_p(x)\|^2 + \|Log_p(y)\|^2 \quad (6)$$

Now,

$$E[e(x,y)] = E[d^2(x,y) - \|Log_p(x) - Log_p(y)\|^2] \quad (7)$$
$$\leq E[d^2(x,p) + d^2(p,y) - \|Log_p(x) - Log_p(y)\|^2] \quad (8)$$
$$= E[\|Log_p(x)\|^2 + \|Log_p(y)\|^2 - \|Log_p(x) - Log_p(y)\|^2] \quad (9)$$
$$= 2E[\langle Log_p(x), Log_p(y)\rangle], \quad (10)$$

where, $\langle v_1, v_2\rangle = v_1^T v_2$. Since, we chose the pole as the centroid of the dataset, in many practical situations this results in the embeddings being zero mean i.i.d. in the tangent plane at the pole. Thus $E[\langle Log_p(x), Log_p(y)\rangle] = 0$. Hence, $E[e(x,y)] = 0$. Further, when the manifold is very high dimensional, we can leverage the fact that i.i.d. Gaussian vectors in high dimensions are nearly orthogonal, also known as concentration inequalities for projections in high dimensions [4]. Hence, in most practical applications it can be argued that statistically $d^2(x,y) \approx \|Log_p(x) - Log_p(y)\|^2$ with high probability even for points that are not necessarily close to the pole. We found that this approximation works well in our experiments.

Pole and Centroid Computation: Selecting the pole as the centroid entails high computational cost as well because computing the centroid is: 1) an iterative procedure, 2) requires several passes over the dataset till convergence, 3) a batch procedure requiring loading and processing all the available data at once. In our case it is sufficient to find a reasonable approximation to the intrinsic mean. Here, we propose a single-pass, non-iterative, recursive approximation to the intrinsic mean which we call the 'cumulative intrinsic mean'. In Euclidean space, given a set of points $\{X_i\}$, the cumulative/running mean is given by $\mu_{run}^{i+1} = \frac{X_{i+1} + i\mu_{run}^i}{i+1}$, with $\mu_{run}^{(1)} = X_1$. For the case of manifolds, this can be generalized as

$$\mu^{(i+1)} = Exp_{\mu^{(i)}}\left[\frac{1}{i+1} Log_{\mu^{(i)}} X_{i+1}\right] \quad (11)$$

This is a non-iterative procedure, and requires only a single pass over the dataset. Further it is a recursive procedure and does not require storing all the data in memory.

4.1 An Algorithm for Nearest-neighbors

Using the result derived in the previous section, we can now employ any of the variety of nearest-neighbor searching algorithms developed for Euclidean spaces. We note that since the embedding provides approximate distances, any search algorithm using this embedding will necessarily be approximate. Thus, it is not meaningful to employ exact search algorithms on the embedded space. Hence, we discuss just one example of approximate search – Hashing. Any other algorithm can be similarly deployed. Hashing was originally studied in the field of cryptography and text databases which involved entities such as passwords, names, addresses etc where exact matches was the key requirement. Hashing image and video data brought additional challenges since no two semantically related images or videos are exactly the same. This brought about a new class of techniques - LSH [16] - that could answer approximate nearest neighbor queries in a fast manner.

A good introduction and survey of LSH can be found in [11]. Here, we briefly review the basic concepts of Euclidean LSH. LSH

attempts to solve a problem called the (r, ϵ)-NN problem. The problem is described as follows: Given a database of points $X = \{x_i\}$ in $\mathbb{R}^n$, and a query q, if there exists a point $x \in X$ such that $d(x, q) \leq r$, then with high-probability, a point $x' \in X$ is retrieved such that $d(x', q) \leq (1 + \epsilon)r$. LSH solves this problem by constructing a family of hash-functions $\mathcal{H}$ over X called locality-sensitive, if for any $u, v \in X$

$$d(u, v) \leq r \Rightarrow Pr(h(u) = h(v)) \geq p_1 \quad (12)$$

$$d(u, v) \geq (1 + \epsilon)r \Rightarrow Pr(h(u) = h(v)) \leq p_2 \quad (13)$$

Popular choices of h include random projections i.e. $h(v) = sgn(v.r)$ where r is a randomly chosen unit-vector, and sgn is the signum function. In this case, h is binary valued taking values in $\{+1, -1\}$. A generalization of this is termed random projections using 'p-stable' distributions [13], where $h(v) = \lfloor \frac{v.r+b}{w} \rfloor$ where r is a randomly chosen direction whose entries are chosen independently from a stable distribution, and b is a random number chosen between $[0, w]$. In this case, the hash function takes on integer values. A $k-$bit hash is constructed by appending k randomly chosen hash-functions $H(x) = [h_1(x), h_2(x), \ldots h_k(x)]$. Thus, $H \in \mathcal{H}^k$. Then, L hash tables are constructed by randomly choosing $H_1, H_2 \ldots H_L \in \mathcal{H}^k$. All the training examples are hashed into the L hash tables. For a query point q, an exhaustive search is carried out among the examples in the union of the L hash-buckets indexed by q. Appropriate choices of K and L ensure that the algorithm succeeds in finding a (r, ϵ)-NN of the query q with a high probability. To extend this to manifolds, the basic idea is to first pick a pole p such that the approximation in equation (10) is valid. Once this pole is chosen, we embed all points in the tangent space at p. Then we randomly sample a direction $v \in T_p\mathcal{M}$. Using this, one can obtain a family of hash functions such as:

$$h_p(x) = sgn(Log_p(x).v), \text{ where } v \in T_p\mathcal{M}, \quad (14)$$

$$h_p(x) = \left\lfloor \frac{\Pi_{H_p}(x) + b}{w} \right\rfloor \approx \left\lfloor \frac{Log_p(x).v + b}{w} \right\rfloor, \text{ where } v \in T_p\mathcal{M}. \quad (15)$$

A K-bit hash function can now be computed by appending K-random hash functions which are obtained by choosing K-different random choices of v. For a query point q, an exhaustive geodesic distance based search is carried out among the examples in the union of the L hash-buckets indexed by q. This is illustrated in figure 2.

5. APPLICATIONS AND EXPERIMENTS

In this section we demonstrate the utility of the proposed framework to enable fast nearest neighbor searches in non-Euclidean manifolds.

5.1 Region Covariances

In recent years, the region covariance has proved to be a popular feature to encode local shape and texture. Covariance features as region descriptors were introduced in [41] and have been successfully applied to human detection [40], object tracking [31] and texture classification [41]. The space of $d \times d$ covariance matrices is denoted as $Sym^+(d)$. The distance between two covariance matrices is given as $d(C_1, C_2) = \sqrt{\sum_{i=1}^{n} ln^2 \lambda_i(C_1, C_2)}$, where $\{\lambda_i\}$ are the solutions of the generalized eigenvalue problem given by $\lambda C_1 x = C_2 x$ [30]. To perform hashing, we need the exponential and logarithmic maps. The exponential and logarithmic map of a point $y \in T_X$ at X are given by [30]

$$Exp_X(y) = X^{\frac{1}{2}} expm(X^{-\frac{1}{2}} y X^{-\frac{1}{2}}) X^{\frac{1}{2}}, \quad (16)$$

$$Log_X(Y) = X^{\frac{1}{2}} logm(X^{-\frac{1}{2}} Y X^{-\frac{1}{2}}) X^{\frac{1}{2}}, \quad (17)$$

where $expm$ and $logm$ denote the usual matrix exponential and matrix logarithms. In this experiment, we consider the texture classification problem as detailed in [41]. We use the Brodatz texture database which contains 111 textures [1] (with D-14 missing). Each of the 640×640 image was divided in 4 parts each of size 320×320. Two of these were used in training and two were used in testing. Given an image, we select a rectangular region of random size between 16×16 and 128×128. Within each such window, at each pixel location we compute $F(x, y) = \left[I(x, y), \left|\frac{\partial I}{\partial x}\right|, \left|\frac{\partial I}{\partial y}\right|, \left|\frac{\partial^2 I}{\partial x^2}\right|, \left|\frac{\partial^2 I}{\partial y^2}\right| \right]^T$.

Then, we compute the 5×5 covariance of these feature as the overall descriptor for the window. We randomly sample 100 such windows, giving rise to a collection of 100 covariance matrices as the representation of each image. In testing, we compute 100 covariance matrices in the same manner for each test-image. For each of these matrices, we find the nearest neighbor in the training dataset and assign the image-class of the corresponding nearest neighbor. Then, from the 100 such labels obtained, we use a majority voting rule to assign the texture class to the image. Now, there are $111 \times 2 = 222$ images in the training set. Each image is represented as a set of 100 covariance matrices. This gives rise to a total of $22,200$ covariance matrices in the training set. The testing scheme as described above would require us to find nearest neighbors in this large set. We show in table 2 how the proposed hashing framework improves the search efficiency without significant loss of accuracy.

Method	Geodesic Computations per test image ($\times 10^2$)			**Accuracy**
	Median	**Mean**	**Max**	
Exhaustive 1-NN	22,200	22,200	22,200	95.49%
Hash $K = 30, L = 10$	298.5	1180.4	4894	95.49%
Hash $K = 30, L = 2$	109	935	4012	94.14%
Hash $K = 50, L = 10$	95	746.04	3810	95.04%
Hash $K = 50, L = 2$	14	320.69	2379	93.24%

Table 2: Comparison of geodesic computations and accuracy for covariance based texture classification on Brodatz database.

For comparison, we show in table 3 the state-of-the art recognition accuracies on the Brodatz database using several methods taken from [41]. Note that even though we use the same covariance features and testing methodology of [41], we obtained an accuracy of 95.49% even with exhaustive nearest-neighbors when compared to 97.77% as reported in [41]. This can be attributed to the variation due to the random choice of s windows in the images over which covariance is computed. However, the goal is not to beat their performance or even replicate it exactly, but to show that the same performance can be achieved with significantly fewer geodesic computations.

Method	Performance
Oriented Filters	85.71%
Oriented Filters	94.64%
Symmetric Filters	93.30 %
LM	97.32%
Covariance	97.77%
Covariance + Hash	95.49 %

Table 3: Comparison of accuracies for different methods on the Brodatz database taken from [41].

[1] Obtained from http://www.ux.uis.no/∼tranden/brodatz.html

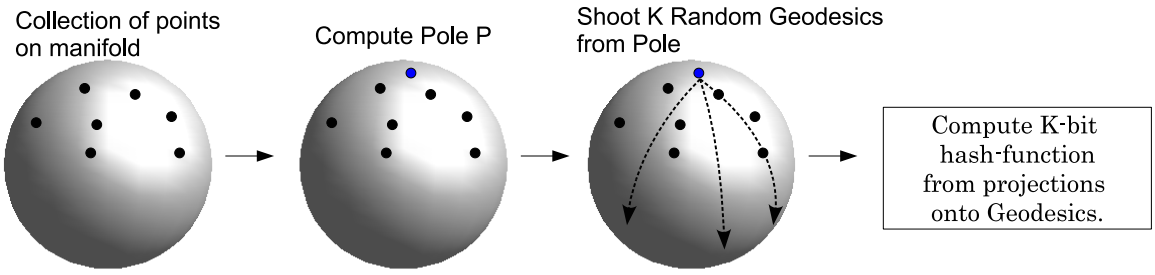


Figure 2: Overall Hashing Framework. Given a set of points on a manifold, we first compute a suitable pole as discussed in section 4. Then, we shoot K random geodesics from the pole. Projections onto these geodesics are computed as described in section 4. From the projections, a hash function is computed such as in equation (14) or (15) which are then appended to form a K-bit hash function. Then, L hash-tables are created by choosing L different K-bit hash functions

5.2 Affine Shape Space: Grassmann manifold

In this experiment, we consider the problem of face recognition using facial geometry. By facial geometry we refer to the location of 2D facial landmarks on images. In several face recognition tasks, the locations of the landmarks have been shown to be extremely informative [45, 32]. A shape is represented by a set of landmark points, given by a $n\times 2$ matrix $L = [(x_1, y_1); (x_2, y_2); \ldots; (x_n, y_n)]$, of the set of n landmarks of the centered shape. Small changes in camera location or change in the pose of the subject can be approximated well as affine transformations on the original base shape. The affine transforms of the shape can be derived from the base shape simply by multiplying the shape matrix L by a 2×2 full rank matrix on the right. For example, let A be a 2×2 affine transformation matrix i.e. $A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$. Then, all affine transforms of the base shape L_{base} can be expressed as $L_{affine}(A) = L_{base} * A^T$. Note that, multiplication by a full-rank matrix on the right preserves the column-space of the matrix L_{base}. Thus, the 2D subspace of $\mathbb{R}^n$ spanned by the columns of the matrix L_{base} is an *affine-invariant* representation of the shape. i.e. $span(L_{base})$ is invariant to affine transforms of the shape. Subspaces such as these can be identified as points on a Grassmann manifold. The Grassmann manifold $\mathcal{G}_{n,d}$ is the space whose points are *d-planes* or *d*-dimensional hyperplanes (containing the origin) in $\mathbb{R}^n$.

A natural way to interpret the Grassmann manifold is as a quotient of the special orthogonal group $SO(n)$. For any $O \in SO(n)$, a geodesic flow in a tangent direction, say, $O^T A$, is given by $\psi_O(A, t) = O^T \exp(tA)$ where exp is the matrix exponential. Using this it can be deduced that, in the case of $\mathcal{G}_{n,d}$ a geodesic flow starting from a point $U = O^T J$ is of the type: $t \mapsto O^T \exp(tA)J$. On $\mathcal{G}_{n,d}$, the exponential map is given by: $O^T \begin{bmatrix} C \\ B^T \end{bmatrix} \equiv O^T AJ \mapsto O^T \exp(A)J$, where A takes the form $\begin{bmatrix} 0 & -B \\ B^T & 0 \end{bmatrix}$. The expression for inverse exponential map is not available analytically for these manifolds and is computed numerically. The details of these computations can be found in [15].

We use the publicly available FG-Net dataset [1], which contains 82 subjects in the age ranges of $0 - 69$. For this dataset, 68 fiducial points are available with each face. We show some sample landmarks from the dataset in figure 3. There are a total of 1002 images. We perform a leave-one-out test to quantity the face recognition performance. This is a challenging dataset as there are many sources of appearance variations for each person such as age, facial hair, expression, pose and illumination. We show the results of leave-one-out nearest neighbor classification on this dataset in table 4. We see that we can get comparable accuracy to exhaustive nearest neighbors with significantly fewer geodesic computations.

Method	Geodesic Computations per test face			**Accuracy**
	Median	**Mean**	**Max**	
Exhaustive 1-NN	1001	1001	1001	22.45%
Hash $K = 11, L = 5$	88	92.07	262	21.35%
Hash $K = 20, L = 50$	33	36.9	122	21.05%
Hash $K = 20, L = 40$	25	28.20	86	19.16%

Table 4: Face Recognition across aging: Comparison of lookups and accuracy for face recognition using landmarks on FG-Net aging data.

6. CONCLUSION

We first discussed the importance of solving nearest-neighbor searching problem on non-Euclidean manifolds for various vision applications. We discussed that the complexity of computing distance functions and centroids sets this problem apart from usual metric space indexing. Then, we provided a formal framework for addressing the problem. From the geometric framework, we derived an approximate method of searching via approximate embeddings using the logarithmic map. We studied the error incurred in such an embedding and showed that this performs well in real experiments. Experiments demonstrate that it is possible to significantly reduce the number of geodesic computations, thereby improving speed, while obtaining comparable performance to exhaustive nearest-neighbors.

7. REFERENCES

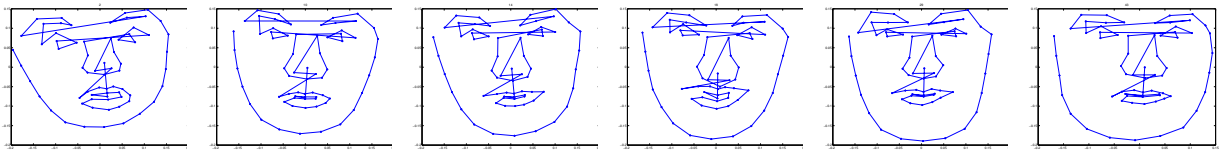
[1] The fg-net aging database. *Available: http://www.fgnet.rsunit.com/*.

[2] O. Arandjelovic, G. Shakhnarovich, J. Fisher, R. Cipolla, and T. Darrell. Face recognition with image sets using manifold density divergence. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 581–588, June 2005.

[3] V. Athitsos, M. Potamias, P. Papapetrou, and G. Kollios. Nearest neighbor retrieval using distance-based hashing. In *Proceedings of the 2008 IEEE 24th International Conference on Data Engineering*, pages 327–336, 2008.

[4] R. G. Baraniuk, M. A. Davenport, R. A. DeVore, and M. B. Wakin. A simple proof of the restricted isometry property for random matrices. *Constructive Approximation*, 28(3):253–263, Dec. 2008.

[5] R. Basri, T. Hassner, and L. Zelnik Manor. Approximate nearest subspace search with applications to pattern



(a) Age 2 (b) Age 10 (c) Age 14 (d) Age 18 (e) Age 29 (f) Age 43

Figure 3: Facial geometric variation across ages. Samples shown correspond to an individual from the FG-net dataset.

recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[6] E. Begelfor and M. Werman. Affine invariance revisited. *IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[7] J. Bourgain. On lipschitz embedding of finite metric spaces in hilbert space. *Israel Journal of Mathematics*, 52(1):46–52, March 1985.

[8] R. Chaudhry and Y. Ivanov. Fast approximate nearest neighbor methods for non-euclidean manifolds with applications to human activity analysis in videos. In *European Conference on Computer Vision*, Crete, Greece, September 2010.

[9] E. Chávez, G. Navarro, R. Baeza-Yates, and J. L. Marroquín. Searching in metric spaces. *ACM Computing Surveys*, 33(3):273–321, 2001.

[10] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 886–893, 2005.

[11] T. Darrell, P. Indyk, and G. E. Shakhnarovich. *Nearest Neighbor Methods in Learning and Vision: Theory and Practice*. MIT Press, 2006.

[12] S. Dasgupta and Y. Freund. Random projection trees and low dimensional manifolds. In *Proceedings of the 40th annual ACM symposium on Theory of computing*, pages 537–546, 2008.

[13] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. *Proceedings of the Symposium on Computational geometry*, pages 253–262, 2004.

[14] P. T. Fletcher, C. Lu, S. M. Pizer, and S. C. Joshi. Principal geodesic analysis for the study of nonlinear statistics of shape. *IEEE Transactions on Medical Imaging*, 23(8):995–1005, August 2004.

[15] K. Gallivan, A. Srivastava, X. Liu, and P. VanDooren. Efficient algorithms for inferences on grassmann manifolds. *12th IEEE Workshop Statistical Signal Processing*, 2003.

[16] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. *Proceedings of the International Conference on Very Large Data Bases*, pages 518–529, 1999.

[17] J. Hafner, H. S. Sawhney, W. Equitz, M. Flickner, and W. Niblack. Efficient color histogram indexing for quadratic form distance functions. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 17(7):729–736, 1995.

[18] J. Ham and D. D. Lee. Extended grassmann kernels for subspace-based learning. *Neural Information Processing Systems*, pages 601–608, 2008.

[19] J. Hamm and D. D. Lee. Grassmann discriminant analysis: a unifying view on subspace-based learning. In *International Conference on Machine Learning*, pages 376–383, June 2008.

[20] P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proc. of 30th STOC*, pages 604–613, 1998.

[21] H. Karcher. Riemannian center of mass and mollifier smoothing. *Communications on Pure and Applied Mathematics*, 30(5):509–541, 1977.

[22] D. Kendall. Shape manifolds, Procrustean metrics and complex projective spaces. *Bulletin of London Mathematical society*, 16:81–121, 1984.

[23] T. K. Kim, J. Kittler, and R. Cipolla. Discriminative learning and recognition of image set classes using canonical correlations. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 29(6):1005–1018, June 2007.

[24] N. Kumar, L. Zhang, and S. Nayar. What is a good nearest neighbors algorithm for finding similar patches in images? In *European Conference on Computer Vision*, pages 364–378, 2008.

[25] K.-C. Lee, J. Ho, and D. J. Kriegman. Acquiring linear subspaces for face recognition under variable lighting. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 27(5):684–698, May 2005.

[26] D. G. Lowe. Object recognition from local scale-invariant features. *IEEE International Conference on Computer Vision*, page 1150, 1999.

[27] Y. M. Lui and J. R. Beveridge. Grassmann registration manifolds for face recognition. In *European Conference on Computer Vision*, pages 44–57, Marseille, France, October 2008.

[28] Y. M. Lui, J. R. Beveridge, and M. Kirby. Canonical stiefel quotient and its application to generic face recognition in illumination spaces. In *Biometrics: Theory, Applications, and Systems*, August 2009.

[29] X. Pennec. Intrinsic statistics on riemannian manifolds: Basic tools for geometric measurements. *Journal of Mathematical Imaging and Vision*, 25(1):127–154, 2006.

[30] X. Pennec, P. Fillard, and N. Ayache. A riemannian framework for tensor computing. *International Journal of Computer Vision*, 66(1):41–66, 2006.

[31] F. Porikli, O. Tuzel, and P. Meer. Covariance tracking using model update based on lie algebra. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 728–735, 2006.

[32] J. Shi, A. Samal, and D. Marx. How effective are landmarks and their geometry for face recognition? *Computer Vision and Image Understanding*, 102(2):117–133, 2006.

[33] S. Soatto, G. Doretto, and Y. N. Wu. Dynamic textures. *ICCV*, 2:439–446, 2001.

[34] M. Spivak. *A Comprehensive Introduction to Differential Geometry*, volume One. Publish or Perish, Inc., Houston, Texas, third edition, 1999.

[35] A. Srivastava, I. Jermyn, and S. Joshi. Riemannian analysis of probability density functions with applications in vision. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[36] A. Srivastava, S. H. Joshi, W. Mio, and X. Liu. Statistical shape analysis: Clustering, learning, and testing. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 27(4), 2005.

[37] A. Srivastava and X. Liu. Tools for application-driven linear dimension reduction. *Neurocomputing*, 67:136–160, 2005.

[38] A. Srivastava, W. Mio, E. Klassen, and S. Joshi. Geometric analysis of continuous, planar shapes. *Proc. 4th International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, 2003.

[39] P. K. Turaga, A. Veeraraghavan, and R. Chellappa. Statistical analysis on stiefel and grassmann manifolds with applications in computer vision. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[40] O. Tuzel, F. Porikli, and P. Meer. Pedestrian detection via classification on riemannian manifolds. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 30(10):1713–1727, 2008.

[41] O. Tuzel, F. M. Porikli, and P. Meer. Region covariance: A fast descriptor for detection and classification. *European Conference on Computer Vision*, pages II: 589–600, 2006.

[42] A. Veeraraghavan, A. Roy-Chowdhury, and R. Chellappa. Matching shape sequences in video with an application to human movement analysis. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 27(12):1896–1909, 2005.

[43] A. Veeraraghavan, A. Srivastava, A. K. Roy Chowdhury, and R. Chellappa. Rate-invariant recognition of humans and their activities. *IEEE Trans. on Image Processing*, 18(6):1326–1339, June 2009.

[44] X. Wang, J. T. Wang, K.-I. Lin, D. Shasha, B. A. Shapiro, and K. Zhang. An index structure for data mining and clustering. *Knowledge and Information Systems*, 2(2):161–184, 2000.

[45] L. Wiskott, J.-M. Fellous, N. Krüger, and C. von der Malsburg. Face recognition by elastic bunch graph matching. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(7):775–779, 1997.