21st International Symposium on
Mathematical Theory of Networks and Systems
July 7-11, 2014. Groningen, The Netherlands

Riemannian Optimization for Elastic Shape Analysis*

Wen Huang[1], Kyle A. Gallivan[2], Anuj Srivastava[3], and P.-A. Absil[4]

Abstract— **In elastic shape analysis, a representation of a shape is invariant to translation, scaling, rotation and re-parameterization and important problems (such as computing the distance and geodesic between two curves, the mean of a set of curves, and other statistical analyses) require finding a best rotation and re-parameterization between two curves. In this paper, we focus on this key subproblem and study different tools for optimizations on the joint group of rotations and re-parameterizations. In this conference paper, we give a first account of a novel Riemannian optimization approach and evaluate its use in computing the distance between two curves and classification using two public data sets. Experiments show significant advantages in computational time and reliability in performance compared to the current state-of-the-art method. Further information will become available in a forthcoming full version of this conference paper.**

I. Introduction

Shape analysis of curves plays an important role in a variety of imaging applications. The basic idea is to isolate contours of objects in images (2D or 3D) and use the shapes of these contours to characterize the original objects. Hence, there is a great interest in tools for shape analysis of planar, closed contours.

Many approaches to shape analysis have been proposed in the literature and used to varying degrees of success in applications, e.g., point-based methods, domain-based shape representations and parameterized curve representations. A large majority of past work on statistical shape analysis has been using landmark-based descriptions [4]. In this setup, one samples contours with a fixed number of points in a pre-determined way, e.g. using uniform spacing, and the ensuing analysis is based on Euclidean analysis of vectors of landmarks. A consequence of this analysis is that the registration of landmarks – which points on one contour match with which points on the other – is already predetermined. This often results in matching parts across shapes that have different geometrical features. An important solution to this and related problems in shape analysis of contours came in the form of elastic shape analysis which has become increasingly important in recent years due to its superior theoretical basis and empirically demonstrated effectiveness. In elastic shape analysis of contour, the objects of study are parameterized contours and, using appropriate metric, one solves for the optimal re-parameterizations of contours during pairwise comparisons. Figure 1 shows geodesics between two closed curves with and without re-parameterization. Using re-parameterization clearly gives a more natural transformation between two curves. More abstractly, imposing re-parameterization leads to shape metrics and statistical summaries of shapes that are invariant to the original parameterizations of the contours. The flexibility in parameterizations of curves helps in improving matching of parts across shapes and has the effect of stretching/bending the curves in optimally deforming one into the other – hence, the name elastic shape analysis. Such a framework was first introduced for general 2D curves by Younes in 1998 [15] and later study by several groups. Klassen et al. [7] narrowed the focus by studying shape analysis of *closed*, planar curves but did not allow the curves to have arbitrary parameterizations. Younes et al. [16] focused on elastic analysis of closed curves using complex representations of 2D coordinates of curves. Srivastava et al. [12] further extended this analysis to include curves in general Euclidean spaces by introducing a novel mathematical representations called the square root velocity (SRV) functions.

An important advantage of SRV functions was that their usage transformed a complicated elastic Riemannian metric into the more standard $\mathbb{L}^2$ metric. Naturally, it preserved the isometry property of the elastic metric despite simplification. The isometry property is that if any two curves are re-parameterized by the same function, then the resulting distance between them under the elastic metric does not change. Thus, one can define re-parameterization (and rotation) orbits of given contours as equivalence classes from the perspective of shape analysis, and induce the $\mathbb{L}^2$ norm from the SRV representation to a quotient space modulo rotation and re-parameterization. This leads to a definition of shape distance between any two curves as the distance between the corresponding orbits in the quotient space.

A fundamental operation in elastic shape analysis, upon which many other important tasks depend, is the accurate and efficient computation of distance between shapes of two curves. This typically involves solving an optimization problem on the joint space of re-parameterizations and rotations. In this paper, we focus on this important subproblem in elastic shape analysis and study different tools for optimizations on the joint group. We develop and analyze a novel optimization approach to solving for optimal re-parameterizations and rotations between two curves, and

*This paper presents research results of the Belgian Network DYSCO (Dynamical Systems, Control, and Optimization), funded by the Interuniversity Attraction Poles Programme initiated by the Belgian Science Policy Office.

[1]Wen Huang is with Department of Mathematics, Florida State University, Tallahassee FL 32306-4510, USA wh08@fsu.edu

[2]Kyle A. Gallivan is with Department of Mathematics, Florida State University, Tallahassee FL 32306-4510, USA kgallivan@fsu.edu

[3]Anuj Srivastava is with Department of Statistics, Florida State University, Tallahassee FL 32306-4330, USA anuj@stat.fsu.edu

[4]P.-A. Absil is with Department of Mathematical Engineering, ICTEAM Institute, Université catholique de Louvain, B-1348 Louvain-la-Neuve, Belgium absil@inma.ucl.ac.be

ISBN: 978-90-367-6321-9

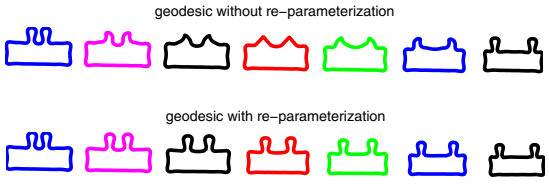

Fig. 1. Geodesics without and with re-parameterization are given by the frameworks of landmark-based Kendall's shape analysis [4] and elastic shape analysis [12] respectively.

evaluate its use in computing the distance between two curves and for classification of closed curves in the plane.

This paper is organized as follows. Section II presents the Riemannian framework for shape analysis including the definition of the elastic metric for open and closed curves in $\mathbb{R}^n$. The proposed Riemannian approach to the solution of the optimization problem that defines the elastic distance metric evaluation is derived in Section III and a brief discussion of its implementation using Riemannian optimization algorithms follows in Section IV. Empirical evaluation of the relative efficiency and effectiveness of the methods is presented in Section V and our conclusions are given in Section VI.

II. Riemannian Framework and Problem Statement

A. Curve representation

The derivation of the basic representation of a shape begins with a parameterized curve, i.e., $\beta(t) : \mathbb{D} \to \mathbb{R}^n$, where $\mathbb{D}$ is the domain of the curve, $\mathbb{D} = [0,1]$ for an open curve and $\mathbb{D} = \mathbb{S}^1$, i.e., the unit circle in $\mathbb{R}^2$, for a closed curve. The shape is taken to be invariant with respect to rescaling, translation, and rotation for inelastic shape analysis, while elastic shape analysis adds invariance with respect to re-parameterization. All four invariants must be taken into account when developing a representation that supports efficient and robust computation.

The framework of Srivastava et al. [12] uses the square root velocity (SRV) function

$$q(t) = \frac{\dot{\beta}(t)}{\sqrt{\|\dot{\beta}(t)\|_2}}$$

as the basis for elastic analysis of a shape defined by the parameterized curve $\beta(t)$. Observe that $\dot{\beta}(t)$ can be recovered from $q(t)$ by $\dot{\beta}(t) = \|q(t)\|_2 q(t)$. Translation is removed automatically by the use of $\dot{\beta}(t)$ in the definition. Rescaling is removed by the normalization of the length of the curve to 1. Since the length of a curve, $\beta(t)$, is $\int_{\mathbb{D}} \|\dot{\beta}(t)\|_2 dt = \int_{\mathbb{D}} \|q(t)\|_2^2 dt$, the normalization requires that $\int_{\mathbb{D}} \|q(t)\|_2^2 dt = 1$ and the set of all SRV functions is the unit sphere in $\mathbb{L}^2(\mathbb{D}, \mathbb{R}^n)$. This sphere is called the preshape space. For open curves in $\mathbb{R}^n$, the domain is $\mathbb{D} = [0,1]$ and the preshape space

$$l_n^o = \{q : [0,1] \to \mathbb{R}^n | \int_0^1 \|q(t)\|_2^2 dt = 1\},$$

is the unit sphere of $\mathbb{L}^2([0,1], \mathbb{R}^n)$. For closed curves, the domain is $\mathbb{D} = S^1$ and the preshape space is

$$l_n^c = \{q : \mathbb{S}^1 \to \mathbb{R}^n | \int_{\mathbb{S}^1} \|q(t)\|_2^2 dt = 1, \int_{\mathbb{S}^1} q(t)\|q(t)\|_2 dt = 0\},$$

where $\int_{\mathbb{S}^1} q(t)\|q(t)\|_2 dt = 0$ is the closure condition.

Removing rotation and re-parameterization is required to define the shape space. This is done by defining an appropriate quotient operation via isometric group actions. This, in turn, defines the distance between curves, the associated optimization problem, and other key tasks such as determining geodesics containing the two curves. Since the approaches taken differ for open and closed curves, they are considered separately below. However, both approaches require the rotation and re-parameterization groups, and their actions. In these two definitions, Γ and l_n are used to indicate the re-parameterization group and preshape space for both open and closed curves. They are distinguished in later discussions by the addition of a superscript o and c respectively.

Definition 2.1: The rotation group for curves in $\mathbb{R}^n$ is

$$\mathrm{SO}(n) = \{O \in \mathbb{R}^{n \times n} | O^T O = I_n, \det(O) = 1\}.$$

and its action is $\mathrm{SO}(n) \times l_n \to l_n : (O, q) \to Oq$.

Definition 2.2: The re-parameterization group for curves in $\mathbb{R}^n$ is

$$\Gamma = \{\gamma : \mathbb{D} \to \mathbb{D} | \gamma \in \mathcal{D}(\mathbb{D}, \mathbb{D})\} \quad (1)$$

and its action is $l_n \times \Gamma \to l_n : (q, \gamma) \to (q \circ \gamma)\sqrt{\dot{\gamma}}$, where $\mathcal{D}(\mathbb{D}, \mathbb{D})$ is the set of diffeomorphisms from $\mathbb{D}$ to itself (an invertible function such that both the function and its inverse are smooth, i.e., in C^∞).

The group operation of Γ is the function composition.

We denote by γ^{-1} the reciprocal (also called inverse) of function γ. To avoid confusion, we use $\frac{1}{\gamma}$ for the pointwise numerical inverse.

B. Open curves in $\mathbb{R}^n$

The preshape space for open curves, l_n^o, is a well-known infinite dimensional manifold. The tangent space of $q \in l_n^o$ is

$$\mathrm{T}_q\, l_n^o = \{v : [0,1] \to \mathbb{R}^n | \int_0^1 q(t)^T v(t) dt = 0\}.$$

The Riemannian metric on l_n^o can be taken as the endowed metric from the embedding space $\mathbb{L}^2([0,1], \mathbb{R}^n)$, i.e.,

$$\langle v_1, v_2 \rangle_{l_n^o} = \langle v_1, v_2 \rangle_{\mathbb{L}^2} = \int_0^1 v_1(t)^T v_2(t) dt,$$

where $v_1, v_2 \in \mathrm{T}_q\, l_n^o$. The distance function on l_n^o induced by this Riemannian metric is (see [12])

$$d_{l_n^o}(x, y) = \cos^{-1} \langle x, y \rangle_{\mathbb{L}^2}. \quad (2)$$

The resulting shape space for open curves is given by the quotient

$$\mathfrak{L}_n^o = \{[\overline{q}] | q \in l_n^o\},$$

where the orbit $\overline{[q]}$ is the closure of the set $[q] = \{O(q \circ \gamma)\sqrt{\dot{\gamma}} | (O, \gamma(t)) \in \mathrm{SO}(n) \times \Gamma^o, q \in l_n^o\}$ (see [11, §1.3.4] for details).

Another way to elaborate this is to first introduce a semigroup:

$$\Gamma_s^o = \{\gamma : [0,1] \to [0,1] | \gamma(0) = 0, \gamma(1) = 1, \\ \gamma \text{ is an absolutely continuous, non-decreasing} \\ \text{and surjective function } \} .$$

It can be shown that Γ_s^o is closed under composition, $\overline{[q]}$ is the orbit of q under the semigroup Γ_s^o and $\mathrm{SO}(n)$ and Γ^o is dense in Γ_s^o (see [11]).

Now we can define a distance between orbits of Γ_s^o, $\overline{[q_1]}$ and $\overline{[q_2]}$ as:

$$d_{\mathfrak{L}_n^o}(\overline{[q_1]}, \overline{[q_2]}) \\ = \inf_{\gamma_1, \gamma_2 \in \Gamma_s^o, O \in \mathrm{SO}(n)} d_{l_n^o}((q_1 \circ \gamma_1)\sqrt{\dot{\gamma}_1}, O(q_2 \circ \gamma_2)\sqrt{\dot{\gamma}_2}) .$$

Since Γ^o is dense in Γ_s^o, for any $\epsilon > 0$, there exists a $\gamma^* \in \Gamma^o$ and an $O^* \in \mathrm{SO}(n)$ such that:

$$|d_{\mathfrak{L}_n^o}(\overline{[q_1]}, \overline{[q_2]}) - d_{l_n^o}(q_1, O^*(q_2 \circ \gamma^*)\sqrt{\dot{\gamma}^*})| < \epsilon .$$

Therefore, our goal is to find such a pair $(O^*, \gamma^*) \in \mathrm{SO}(n) \times \Gamma^o$. Even though this will not be an exact calculation of the shape distance, approximating $d_{\mathfrak{L}_n^o}(\overline{[q_1]}, \overline{[q_2]})$ by

$$d_{l_n^o}(q_1, O(q_2 \circ \gamma)\sqrt{\dot{\gamma}}) \\ = \cos^{-1} \langle q_1, O(q_2 \circ \gamma)\sqrt{\dot{\gamma}} \rangle_{\mathbb{L}^2}, \tag{3}$$

evaluated at (O^*, γ^*) gives a distance for comparing shapes of curves in practical situations.

C. Closed curves in $\mathbb{R}^n$

The preshape space of closed curves, l_n^c, is a submanifold of l_n^o and the Riemannian metric inherited from the embedding space is

$$\langle v_1, v_2 \rangle_{l_n^c} = \langle v_1, v_2 \rangle_{\mathbb{L}^2} = \int_{\mathbb{S}^1} v_1(t)^T v_2(t) dt.$$

The resulting shape space is

$$\mathfrak{L}_n^c = \{\overline{[q]} | q \in l_n^c\},$$

where the orbit $\overline{[q]}$ is the closure of the set $\{O(q \circ \gamma)\sqrt{\dot{\gamma}} | (O, \gamma(t)) \in \mathrm{SO}(n) \times \Gamma^c, q \in l_n^c\}$.

Proceeding as with open curves, a semigroup Γ_s^c, that is closed under composition and in which Γ^c is dense, can be defined.

The distance between orbits $\overline{[q_1]}$ and $\overline{[q_2]}$ of q_1 and q_2 under the semigroup Γ_s^c is

$$d_{\mathfrak{L}_n^c}(\overline{[q_1]}, \overline{[q_2]}) \\ = \inf_{\gamma_1, \gamma_2 \in \Gamma_s^c, O \in \mathrm{SO}(n)} d_{l_n^c}((q_1 \circ \gamma_1)\sqrt{\dot{\gamma}_1}, O(q_2 \circ \gamma_2)\sqrt{\dot{\gamma}_2})$$

and for any $\epsilon > 0$, there exists a $\gamma^* \in \Gamma^c$ and an $O^* \in \mathrm{SO}(n)$ such that:

$$|d_{\mathfrak{L}_n^c}(\overline{[q_1]}, \overline{[q_2]}) - d_{l_n^c}(q_1, O^*(q_2 \circ \gamma^*)\sqrt{\dot{\gamma}^*})| < \epsilon. \tag{4}$$

Unlike the case of open curves, there is no known analytical expression of distance on l_n^c. Since l_n^c is a submanifold of l_n^o, the approximation

$$\arg \min_{\gamma_1, \gamma_2 \in \Gamma_s^c, O \in \mathrm{SO}(n)} d_{l_n^c}((q_1 \circ \gamma_1)\sqrt{\dot{\gamma}_1}, O(q_2 \circ \gamma_2)\sqrt{\dot{\gamma}_2}) \\ \approx \arg \min_{\gamma_1, \gamma_2 \in \Gamma_s^c, O \in \mathrm{SO}(n)} d_{l_n^o}((q_1 \circ \gamma_1)\sqrt{\dot{\gamma}_1}, O(q_2 \circ \gamma_2)\sqrt{\dot{\gamma}_2})$$

is used and the goal is to find a pair $(\gamma^*, O^*) \in \mathrm{SO}(n) \times \Gamma^c$ that satisfies (4). As with open curves, approximating $d_{\mathfrak{L}_n^c}(\overline{[q_1]}, \overline{[q_2]})$ with

$$d_{l_n^o}(q_1, O(q_2 \circ \gamma)\sqrt{\dot{\gamma}}) \\ = \cos^{-1} \langle q_1(t), O(q_2 \circ \gamma(t))\sqrt{\dot{\gamma}(t)} \rangle_{\mathbb{L}^2}. \tag{5}$$

evaluated at (O^*, γ^*) gives a distance for comparing shapes of curves in practical situations.

III. A Riemannian Optimization Method

A Riemannian optimization problem consists of minimizing a real-valued function defined on a Riemannian manifold $\mathcal{M}$. Recent theoretical and algorithmic results and a review of the state-of-the-art can be found in [1], [2], [5], [6], [9], [10]. In order to make use of Riemannian optimization theory and algorithms in the fundamental elastic shape analysis task of efficiently and effectively computing the distance between two curves, we must define an appropriate cost function on a Riemannian manifold, the Riemannian gradient of the cost function, the tangent space of an element in the manifold, the retraction operation on the manifold, and an appropriate vector transport. The definitions of Riemannian gradient, tangent space, retraction and vector transport are standard and can be found in [1], [8].

Several Riemannian optimization algorithms are applicable to the distance computation and a representative set is investigated and compared to the dynamic-programming-based (DP-based) approach of Srivastava et al. [12] for closed curves in this and the following section. Specifically, since the gradient is known and the Hessian is unknown, Riemannian quasi-Newton algorithms given in [5], [6], including the Riemannian trust region symmetric rank-one update method (RTR-SR1), the limited-memory RTR-SR1 (LRTR-SR1), the Riemannian BFGS (RBFGS), the limited-memory RBFGS (LRBFGS), and the Riemannian steepest descent (RSD), are applied to the distance problem and it is shown that a Riemannian approach is more efficient computationally and produces a superior distance computation than the DP-based approach.

A. Cost function

The DP-based approach requires breaking the closed curve into several open curves and taking the minimal solution. Using the Riemannian approach we can handle the closed curve distance problem directly without breaking the closed curve into several open curves. The first step in defining the cost function and associated Riemannian manifold requires reconsidering the representation of Γ^c—the version of Γ in (1) for closed curves—and its group action. Γ^c is

approximated by $\tilde{\Gamma} \times \mathbb{R}$, where $\tilde{\Gamma}$ is Γ (1) with $\mathbb{D} = [0, 2\pi]$, and the Γ^c group action on q

$$(q, \gamma) = q \circ \gamma \sqrt{\dot{\gamma}}, \quad \gamma \in \Gamma^c$$

is replaced with the $\tilde{\Gamma} \times \mathbb{R}$ group action on q defined by

$$(q, (\gamma, m)) = (q(\gamma + m \mod 2\pi))\sqrt{\dot{\gamma}}, \quad (\gamma, m) \in \tilde{\Gamma} \times \mathbb{R}.$$

Note that the addition of $\mathbb{R}$ to the group definition removes the need for break points since the offset has been added as a decision variable.

Srivastava et al.[12] use the cost function

$$H(O, \gamma(t)) = \int_0^1 \|q_1(t) - O(q_2 \circ \gamma(t))\sqrt{\dot{\gamma}(t)}\|_2^2 dt, \quad (6)$$

that has the same extreme points as the cost function used in (5). This is easily seen from

$$\begin{aligned}
&\int_{\mathbb{D}} \|q_1(t) - O(q_2 \circ \gamma(t))\sqrt{\dot{\gamma}(t)}\|_2^2 dt \\
=&\langle q_1, q_1\rangle_{\mathbb{L}^2} + \langle q_2, q_2\rangle_{\mathbb{L}^2} - 2\langle q_1, O(q_2 \circ \gamma)\sqrt{\dot{\gamma}}\rangle_{\mathbb{L}^2} \\
=&2 - 2\cos(\cos^{-1}(\langle q_1, O(q_2 \circ \gamma)\sqrt{\dot{\gamma}}\rangle_{\mathbb{L}^2})) \\
=&2 - 2\cos(d_{l_n^o}(q_1, O(q_2 \circ \gamma)\sqrt{\dot{\gamma}})).
\end{aligned}$$

The cost function on the Riemannian manifold $\mathrm{SO}(n) \times \mathbb{R} \times \tilde{\Gamma}$ is thus

$$\begin{aligned}
&H(O, m, \gamma) \\
=&\int_0^{2\pi} \|q_1(t) - O(q_2(\gamma(t) + m \mod 2\pi))\sqrt{\dot{\gamma}(t)}\|_2^2 dt
\end{aligned}$$

Using the Riemannian manifold structure of $\tilde{\Gamma}$ complicates the basic objects required for a Riemannian optimization algorithm. The tangent space to $\tilde{\Gamma}$ at γ is $\mathrm{T}_\gamma \tilde{\Gamma} = \{v : [0, 2\pi] \to \mathbb{R} | v(0) = v(2\pi) = 0, v \text{ is smooth}\}$. Note that $\tilde{\Gamma}$ is a submanifold of $\mathbb{L}^2([0, 2\pi], \mathbb{R})$, hence it is natural to endow it with the metric from $\mathbb{L}^2([0, 2\pi], \mathbb{R})$. Therefore, the exponential mapping is given by $\mathrm{Exp}_\gamma v = \gamma + v$. When the exponential mapping is used as the retraction in a Riemannian optimization algorithm, the update tangent vector η_i in the update $\gamma_{i+1} = \gamma_i + \eta_i$ must be carefully chosen to guarantee that γ_{i+1} is a diffeomorphism. This is, in general, not easy to guarantee for an arbitrary γ_i on the manifold and we know of no retractions to replace the exponential map that can guarantee a diffeomorphism in a computationally efficient manner. Fortunately, we can approximate $\tilde{\Gamma}$ with another manifold that provides the required computational efficiency while capturing the structure of $\tilde{\Gamma}$ and Γ^c effectively.

Note that any $\gamma \in \tilde{\Gamma}$ and its derivative $\dot{\gamma}$ satisfy the constraints $\gamma(0) = 0, \gamma(2\pi) = 2\pi$ and $\dot{\gamma}(s) > 0$ for all $s \in [0, 2\pi]$. These are equivalent to $\gamma(0) = 0$, $\int_0^{2\pi} \dot{\gamma}(t)dt = 2\pi$ and $\dot{\gamma}(s) > 0$ for all $s \in [0, 2\pi]$. The positivity constraint on the derivative can be guaranteed by replacing $\dot{\gamma}$ with an even power function. For example setting $\dot{\gamma} = l^2$ where $l : [0, 2\pi] \to \mathbb{R}$ yields all of the γ that satisfy the constraints. All three constraints are condensed into the constraints $\int_0^{2\pi} l^2(t)dt = 2\pi$ and $l^2(s) \neq 0$ for all $s \in [0, 2\pi]$. (The method for keeping l^2 away from 0 is discussed in Section IV-B.) Therefore, l is an element of the 2-norm sphere and $\gamma(t)$ can be recovered by $\int_0^t l^2(s)ds$. It follows that $\sqrt{\dot{\gamma}} = |l|$ and the cost function becomes

$$\begin{aligned}
&L(O, m, l) \\
=&\int_0^{2\pi} \|Oq_1(t) - q_2(\int_0^t l^2(s)ds + m \mod 2\pi)|l(t)|\|_2^2 dt.
\end{aligned}$$

While this approach simplifies the constraints considerably, the resulting cost function is only partly smooth due to the presence of $|l(t)|$ and does not satisfy the C^2 smoothness assumption required for the Riemannian quasi-Newton algorithms and other superlinearly convergent Riemannian optimization algorithms (see [5], [6]).

An alternative is to let $l^4 = \dot{\gamma}$. Therefore,

$$l \in \mathcal{L} = \{l : [0, 2\pi] \to \mathbb{R} | \int_0^{2\pi} l^4(t)dt = 2\pi\}.$$

Therefore, l is an element of 4-norm sphere and the cost function becomes

$$\int_0^{2\pi} \|q_1(t) - Oq_2(\int_0^t l^4(s)ds + m \mod 2\pi)l^2(t)\|_2^2 dt$$

which is defined on $\mathrm{SO}(n) \times \mathbb{R} \times \mathcal{L}$. Exploiting the invariance of the norm under isometry, we use the equivalent cost function

$$\begin{aligned}
&L(O, m, l) \\
=&\int_0^{2\pi} \|Oq_1(t) - q_2(\int_0^t l^4(s)ds + m \mod 2\pi)l^2(t)\|_2^2 dt.
\end{aligned}$$

The reason we put O on $q_1(t)$ will be discussed in Section IV-A.

B. The Riemannian manifold

The Riemannian manifold used to define the constraints for the optimization problem associated with the efficient algorithm to compute the distance function for elastic shape analysis is $\mathrm{SO}(n) \times \mathbb{R} \times \mathcal{L}$. The Riemannian gradient of the cost function, the retraction operation on the manifold, and an appropriate vector transport, can be constructed by considering each on the components of the product [5].

$\mathrm{SO}(n)$ is a well-known Riemannian manifold the structure of which is discussed in the literature [1] and the associated implementation issues are considered in [5].

$\mathcal{L}$ is an infinite dimensional Riemannian manifold. The tangent space of $\mathcal{L}$ at any point is therefore an infinite dimensional linear space with elements that are functions defined on $[0, 2\pi]$. The following lemma characterizes, $\mathrm{T}\,\mathcal{L}$, the tangent bundle of $\mathcal{L}$ and an element of a tangent space. The proofs are left out from this conference paper due to length restrictions.

Lemma 3.1: The tangent space $T_l\mathcal{L}$ of $l \in \mathcal{L}$ is

$$\mathrm{T}_l\,\mathcal{L} = \{v : [0, 2\pi] \to \mathbb{R} | \langle l^3, v\rangle_{\mathbb{L}^2} = 0\},$$

and therefore the projection onto the tangent space is

$$P_l(v) = v - l^3 \frac{\langle v, l^3\rangle_{\mathbb{L}^2}}{\langle l^3, l^3\rangle_{\mathbb{L}^2}}$$

Let the metric of $\mathcal{L}$ be endowed from the embedding space $\mathbb{L}^2([0,2\pi],\mathbb{R})$. Analytical forms of the exponential mapping and parallel vector transport on $\mathcal{L}$ are unknown. However, an efficient retraction and isometric vector transport can be constructed. They are characterized in the following Lemma.

Lemma 3.2: The function $R_l(v) : \mathcal{L} \times \mathrm{T}_l\,\mathcal{L} \to \mathcal{L}$

$$R_l(v) = (2\pi)^{1/4}\frac{l+v}{\|l+v\|_{\mathbb{L}^4}}$$

defines a retraction on $\mathcal{L}$ and the associated differentiated retraction is

$$\mathcal{T}_{R_v}u = (2\pi)^{1/4}\frac{u}{\|l+v\|_{\mathbb{L}^4}} - (2\pi)^{1/4}\frac{(l+v)\langle (l+v)^3, u\rangle_{\mathbb{L}^2}}{\|l+v\|^5_{\mathbb{L}^4}},$$

where $\|\cdot\|_{\mathbb{L}^4} = (\int_0^{2\pi}(\cdot)^4 dt)^{1/4}$ and $u, v \in \mathrm{T}_l\,\mathcal{L}$. An isometric vector transport is given by

$$\mathcal{T}_{S_v}u = u - \frac{2(\tilde{l}_1+\tilde{l}_2)\langle \tilde{l}_2, u\rangle_{\mathbb{L}^2}}{\|\tilde{l}_1+\tilde{l}_2\|^2_{\mathbb{L}^2}}.$$

where $\tilde{l}_1 = l^3/\langle l^3, l^3\rangle_{\mathbb{L}^2}$, $\tilde{l}_2 = l_2^3/\langle l_2^3, l_2^3\rangle_{\mathbb{L}^2}$, $l_2 = R_l(v)$.

For the cost function of interest, an analytical form of the Riemannian gradient can be derived. It is given in the following Lemma.

Lemma 3.3: The Riemannian gradient of the cost function

$$\begin{aligned}&L(O,m,l)\\ &= \int_0^{2\pi} \|Oq_1(t) - q_2(\int_0^t l^4(s)ds + m \mod 2\pi)l^2(t)\|_2^2 dt,\\ &(O,m,l) \in \mathrm{SO}(n)\times\mathbb{R}\times\mathcal{L}\end{aligned}$$

is

$$\begin{aligned}&\nabla L(O,m,l) =\\ &(P_O(-2\int_0^{2\pi} q_2(\int_0^t l^4(s)ds + m \mod 2\pi)l^2(t)q_1(t)^T dt),\\ &-2\int_0^{2\pi}\langle Oq_1(t) - l^2(t)q_2(\int_0^t l^4(s)ds + m \mod 2\pi),\\ &\qquad l^2(t)q_2'(\int_0^t l^4(s)ds + m \mod 2\pi)\rangle_2 dt,\\ &P_l(2y(t)l^3(t) - 2x(t)))\end{aligned}$$

where $P_O U = (U - OU^T O)/2$ is the projection to $\mathrm{T}_O\,\mathrm{SO}(n)$,

$$\begin{aligned}x(t) = \langle Oq_1(t) - l^2(t)q_2(\int_0^t l^4(s)ds + m \mod 2\pi),\\ 2l(t)q_2(\int_0^t l^4(s)ds + m \mod 2\pi)\rangle,\end{aligned}$$

and $y(t)$ is any primitive of

$$\begin{aligned}y'(t) = \langle Oq_1(t) - l^2(t)q_2(\int_0^t l^4(s)ds + m \mod 2\pi),\\ 4l^2(t)q_2'(\int_0^t l^4(s)ds + m \mod 2\pi)\rangle.\end{aligned}$$

IV. Implementation Comments

A. Representation and cost function

We assume all of the curves are continuous. In practice, all of the curves are represented by a set of points and therefore, the q-function of a curve $\beta(t)$ is also represented by points that are on some smooth function. Since O is an isometry in the cost functions, we can apply it to either q_1 or q_2. We apply it to q_1. Therefore representing q_1 does not require the use of an interpolatory function and a vector of points is sufficient.

In all of the cost functions considered, q_2 is composed with some function and, therefore, representing q_2 as a set of points is not sufficient. A suitable function must be used. Since the convergence analysis of Riemannian quasi-Newton optimization algorithms requires a C^2 cost function, an interpolatory cubic spline of the set of points on q_2 with the periodic boundary condition is used but a spline of degree 1, i.e., piecewise linear, or degree 2 are also practical.

It should be noted however that there is nothing in the formulation that requires an interpolatory approximation. The discrete points in the representation could be control points for a continuous approximating parameterized curve, e.g., a parameterized B-spline.

Finally, all integrals required by the algorithms are approximated by the Composite Trapezoidal Rule.

B. Regularization

As pointed out above, the function l^2 must not be 0 at any point in $[0,\ 2\pi]$. This is achieved by adding to the cost function a barrier term that prevents the re-parameterization γ as well as its reciprocal from being flat on a nonvanishing interval. In this paper, the cost function for closed curves with the added barrier is

$$\begin{aligned}&\int_0^{2\pi} \|Oq_1(t) - q_2(\int_0^t l^4(s)ds + m \bmod 2\pi)l^2(t)\|_2^2 dt\\ &+ \omega(\int_0^{2\pi}(l^4(t) + \frac{1}{l^4(t)})\sqrt{1+l^8(t)}dt),\end{aligned} \tag{7}$$

C. Escaping local minima

The Riemannian optimization methods mentioned above are local optimization methods that are only guaranteed to converge to a local minimum of the cost function. In order to favor convergence to the global minimum, the initial l_0 are given by a coarse (thus computationally cheap) version of the Coordinate Relaxation method of [12]. Multiple values of m_0 are used and chosen such that the variation of angles along the curve between consecutive values of m_0 are greater than a given threshold. The details will be described in the full version of this paper.

V. Experiments

A. Overview of experiments

In our experiments, the performances of the Riemannian optimization algorithms, RBFGS, LRBFGS, RTR-SR1, LRTR-SR1 and RSD, are compared to identify the preferred

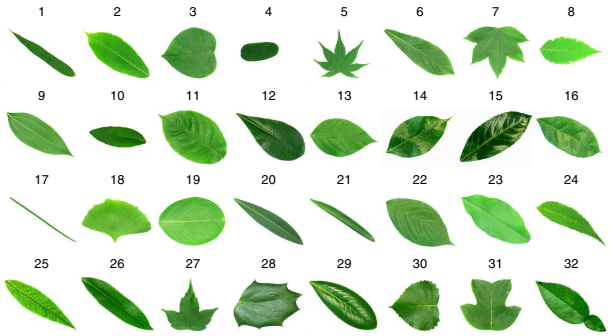

Fig. 2. Samples of leaves from Flavia leaf dataset. One sample per species is illustrated.

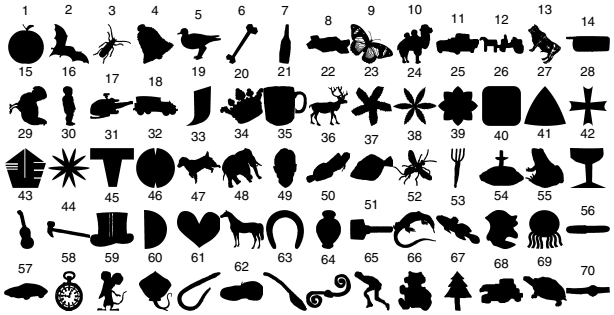

Fig. 3. Samples of curves from MPEG-7 dataset. One sample per cluster is illustrated.

Riemannian method. Then the preferred Riemannian method is compared systematically with the DP-based algorithm used by Srivastava et al. in their experiments [12], i.e., the Coordinate Relaxation method using only one iteration that we denote CR1.

Two public datasets are used in the experiments: the Flavia leaf dataset [14] and the MPEG-7 dataset [13]. The Flavia leaf dataset contains images of 1907 leaves from 32 species. Figure 2 shows an example leaf from each species. MPEG-7 contains 1400 images in 70 clusters each of which contains 20 shapes. Figure 3 shows an example shape from each cluster. The boundary curves of the shapes are extracted using the BWBOUNDARIES function in Matlab and piecewise linear interpolation is used to resample the curve at 100 points in $\mathbb{R}^2$, i.e., a 2×100 matrix.

B. The preferred Riemannian quasi-Newton algorithm

The two public datasets were also used to compare the performances of several Riemannian optimization algorithms in minimizing the cost function (7). For these experiments, the stopping criteria for the Riemannian algorithms require the relative change of the cost function in two successive iterates to be less than 10^{-3}, and the minimum number of iterations is 10. All codes are written in C, compiled with gcc and run on the Florida State University HPC system using Quad-Core 2356 2.3 GHz Opterons [3]. The output time is the average CPU time of 10 runs with identical parameters. (The times observed had very low variance.)

To find the preferred Riemannian method, all of the methods were run on several sets of randomly chosen pairs of shapes from the two datasets. Table I reports, t_{ave}, the average time to compute the distance between two shapes and, L_{ave}, the average cost function value for one of these sets from the Flavia and MPEG-7 shapes. The trends in other sets were similar.

TABLE I

COMPARISON OF RIEMANNIAN METHODS FOR REPRESENTATIVE SETS FROM THE FLAVIA (F) AND MPEG-7 (M) DATASETS: AVERAGE TIME PER PAIR (t_{ave}) IN SECONDS AND AVERAGE COST FUNCTION PER PAIR (L_{ave}).

	RBFGS	LRBFGS	RTR-SR1	LRTR-SR1	RSD
L_{ave}(F)	0.1727	0.1836	0.1772	0.1958	0.2079
t_{ave}(F)	0.4113	0.1525	0.4585	0.2052	0.2218
L_{ave}(M)	0.3639	0.3919	0.3735	0.4407	0.4798
t_{ave}(M)	1.2823	0.4370	1.3352	0.5572	0.7537

The RBFGS and RTR-SR1 methods produce the smallest final cost function values, but this comes at the cost of computational times that are approximately 2 to 3 times those of the other methods. This is easily explained by noting that the computational complexity per step for these two methods is $O(N^2)$ due to the use of a dense matrix vector product. Note this also implies $O(N^2)$ space complexity. Both are less complex than the CR1 method with computational complexity per step of $O(N^3)$ and $O(N^2)$ space complexity.

The RSD method has low computational times due to its relatively low $O(N)$ computational complexity per step, but it does not result in a competitive final cost function value due to the simplicity of the approach.

The limited memory methods of LRBFGS and LRTR-SR1, do not require an $N \times N$ dense matrix vector product or $N \times N$ dense matrix storage. The final cost function values they achieve are not as small as those from RBFGS and RTR-SR1 but they are close, e.g., within 8% for LRBFGS.

The computational complexity per step for LRBFGS is $O(Nkm_s)$ where k is the number of iterations and m_s is the number of initial conditions. Given two curves and a fixed stopping criterion, the m_s and k do not vary that much as N increases. Therefore, practically, the complexity of LRBFGS is $O(N)$. The effect of the substantial difference in the complexities of LRBFGS and CR1 is illustrated in Figure 4 for a representative pair of leaf shapes from the 27th species of the Flavia dataset. Boundary curves were extracted with different number of points N to test the relationship between N and time costs for LRBFGS and CR1. The break points of CR1 are chosen to be every 4 points and the time cost is the average of 10 runs with identical parameters.

In summary, all of the Riemannian algorithms were competitive with CR1 in terms of complexity and LRBFGS with its acceptable optimization of the cost function and its low computational and storage complexity is chosen as the preferred Riemannian algorithm for use further comparisons to CR1 from the point of view of quality of shape distance computations.

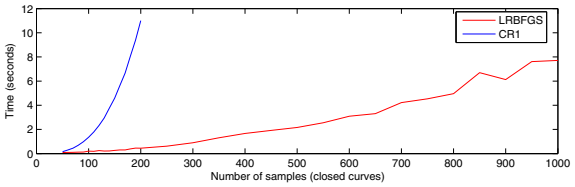

Fig. 4. Comparison of complexities of CR1 and LRBFGS.

C. Performance comparison for Flavia and MPEG-7 datasets

In order to compare the cost and efficacy of the preferred Riemannian algorithm, LRBFGS, to those of the current state-of-the-art, CR1, all pairwise distances in the Flavia and MPEG-7 data sets were computed ($1,819,278$ and $980,700$ pairs respectively) using the testing environment described in Section V-B. For CR1, the effect of the number of break points was considered by running each pair with a break point every 2, 4, 8 and 16 points given a fixed initial point, i.e., the sets are nested.

In addition to comparing the computation times and cost function values for the two algorithms, the quality of the distance computations was assessed using the one-nearest-neighbor (1NN) metric of cluster (species) preservation for the MPEG-7 (Flavia) shapes. The 1NN metric, μ, computes the percentage of points whose nearest neighbor are in the same cluster, i.e.,

$$\mu = \frac{1}{n}\sum_{i=1}^{n} C(i),$$

$$C(i) = \begin{cases} 1 & \text{if point } i \text{ and its nearest} \\ & \text{neighbor are in the same cluster} \\ 0 & \text{otherwise} \end{cases}.$$

A very significant improvement in the final value of the cost function achieved by LRBFGS compared to the value achieved by CR1 is observed. For the Flavia dataset, LRBFGS reduces the cost function more than CR1 in 59.63%, 66.88%, 77.18% and 86.42% of the pairs when choosing break points every 2, 4, 8 and 16 points for CR1 respectively. For the MPEG-7 dataset, LRBFGS minimizes the cost function better than CR1 in 80.08%, 83.28%, 87.61% and 92.24% of the pairs when choosing break points every 2, 4, 8 and 16 points for CR1 respectively.

The distribution of the ratio of the cost function value of CR1 to that of LRBFGS is shown in the histograms in Figure 5. Ratios where LRBFGS was more than 4 times better are not include for presentation purposes. The maximum ratios for the Flavia data set (and the number of ratios exceeding 4) were 7545 (79), 9854 (193), 17526 (544) and 23181 (1921) when choosing break points every 2, 4, 8 and 16 points for CR1 respectively. The maximum ratios for the MPEG-7 data set (and the number of ratios exceeding 4) were 11750 (1), 11750 (3), 11750 (10) and 31882 (125) when choosing break points every 2, 4, 8 and 16 points for CR1 respectively. The amazingly large ratios beyond 4 occur for pairs of shapes that are fairly close in shape where LRBFGS achieves a very small cost function value. Not only is it clear from this data that, in general, LRBFGS reduces the cost function more than CR1, but also in the cases when CR1 produces a smaller cost function value it is usually very close to the value produced by LRBFGS.

Of course, if the improvement in the reduction of the cost function requires a very large increase in computation time then the argument in favor of LRBFGS and the other Riemannian methods weakens. The histograms of computation times for CR1 and LRBFGS for the MPEG-7 and Flavia datasets in Figure 6 show that LRBFGS has a significant advantage in computation time. The largest computation time for LRBFGS is smaller than all computation times of CR1 using $N/2$ and $N/4$ break points for both Flavia and MPEG-7 datasets and also when using $N/8$ break points for the Flavia dataset. When using $N/8$ break points for the MPEG-7 dataset only 6.52% of the pairs have CR1 computation times smaller than the largest LRBFGS computation time. When using $N/16$ break points 0.555% and 46.7% of the pairs have CR1 computation times smaller than the largest LRBFGS computation time for the Flavia and MPEG-7 datasets respectively. Therefore, the Riemannian approach to the cost function and its optimization using LRBFGS yields superior optimization in significantly less time than CR1 for the vast majority of the pairs computed.

A more careful examination of the the times indicates that the Riemannian approach has another advantage. The computation time for a pair of shapes using CR1 is approximately proportional to the number of break points used. There is very little variation between computation times when using the same number of break points as is seen in the CR1 spikes in Figure 6.

Figure 6 also shows that the, much smaller, computation times for LRBFGS have significant variation. We point out that in the Riemannian methods, a method of automatically selecting the position and number of initial conditions is used and the number of chosen initial conditions reflects the complexity of the curves. The method will be described in a full version of this paper. The computation time per initial condition (PIC) for LRBFGS varies only slightly as is shown also in Figure 7, and the computation time for a pair of shapes is approximately proportional to the number of initial conditions used. Since the number of initial conditions is a simple measure of the complexity of one or both of the shapes in the pair, the Riemannian methods have the additional advantage of only requiring a computation time that reflects the difficulty of the problem.

Table II shows the average time cost and 1NN for both datasets. The trends are as expected given the examples in Figures 2, 3. For the MPEG-7 dataset, the shapes in different clusters are very distinct compared to the significantly greater similarity of shapes in certain pairs of species in the Flavia dataset, e.g., species 1 and 21. Therefore, the μ values are expected to be higher for MPEG-7 distances since the distinctions are easier to make while lower μ values are expected for Flavia distances. For CR1 it is expected that

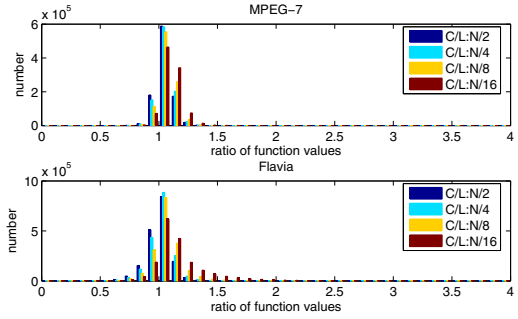

Fig. 5. Histograms of ratios of the CR1 cost function value to the LRBFGS cost function value (C/L) for MPEG-7 and Flavia datasets. $N/i, i = 2, 4, 8, 16$ denote the number of break points in CR1. Bins are $(0, 0.1), \ldots, (0.9, 1.0), \ldots, (3.9, 4.0)$.

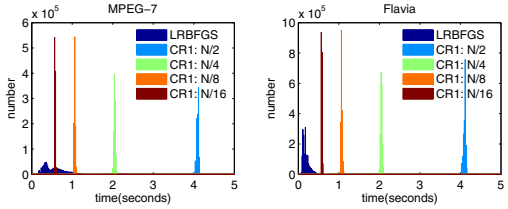

Fig. 6. Histograms of computation times of LRBFGS and CR1 for MPEG-7 and Flavia datasets.

μ values would increase as the number of break points increases. All of these trends are observed in the μ data.

The comparison of μ achieved by LRBFGS to those of CR1 shows a clear advantage to LRBFGS. LRBFGS achieves a value of μ higher than CR1 using the densest set of break points. Not surprisingly, given the distributions of computation times discussed earlier, the average time for LRBFGS is smaller than the average time for even the sparsest set of CR1 break points $(N/16)$.

VI. Conclusion

We have explored the computation of the elastic distance metric for open and closed curves in $\mathbb{R}^2$ and reviewed CR1, the DP-based algorithm of Srivastava et al., [12], the approximations upon which it is based, its computational complexity, its difficulties, and its performance in terms of time and cost function reduction. As an alternative to CR1, we have derived a Riemannian approach to computing the elastic distance metric and developed an efficient implementation using various Riemannian optimization algorithms.

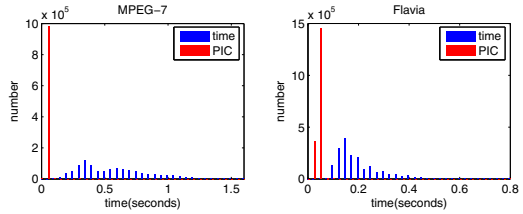

Fig. 7. Histograms of computation times of LRBFGS and computation times per initial condition (PIC) of LRBFGS for MPEG-7 and Flavia datasets.

TABLE II

The average computation time in seconds (AT) and 1NN of LRBFGS and CR1 with break points chosen to be every 2, 4, 8 and 16 points for the Flavia dataset (F) and MPEG-7 dataset (M).

	LRBFGS	CR1			
		$N/16$	$N/8$	$N/4$	$N/2$
AT(F)	0.2000	0.5754	1.0683	2.0541	4.1077
1NN(F)	87.78%	80.02%	82.01%	84.58%	87.31%
AT(M)	0.5668	0.5744	1.0665	2.0507	4.1010
1NN(M)	98.00%	95.21%	97.00%	97.21%	97.00%

Empirical comparisons of the Riemannian approach using LRBFGS and CR1 and shapes from the MPEG-7 and Flavia datasets were performed. The results demonstrate that the Riemannian approach produces more useful distance estimates, as measured by the 1NN metric for clustering, in significantly less time and that the computational time required adapts to the complexity of the shapes being compared.

The efficiency and efficacy of the Riemannian approach to computing the elastic distance metric promises to improve considerably shape analysis computations that are based upon distance computations, e.g., the Karcher mean of a set of shapes, geodesic paths between shapes, and inferences on shapes. These improvements will be demonstrated in future work.

References

[1] P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, Princeton, NJ, 2008.
[2] C. G. Baker. *Riemannian manifold trust-region methods with applications to eigenproblems*. PhD thesis, Florida State University, 2008.
[3] Florida State University Research Computing Center. FSU high performance computing system.
[4] I. L. Dryden and K. V. Mardia. *Statistical shape analysis*. Wiley, 1998.
[5] W. Huang. *Optimization algorithms on Riemannian manifolds with applications*. PhD thesis, Florida State University, 2013.
[6] W. Huang, P.-A. Absil, and K. A. Gallivan. A Riemannian symmetric rank-one trust-region method. *Mathematical Programming*, 2014.
[7] E. Klassen, A. Srivastava, W. Mio, and S. H. Joshi. Analysis of planar shapes using geodesic paths on shape spaces. *IEEE transactions on pattern analysis and machine intelligence*, 26(3):372–83, March 2004.
[8] B. O'Neill. *Semi-Riemannian geometry*. Academic Press Incorporated [Harcourt Brace Jovanovich Publishers], 1983.
[9] C. Qi. *Numerical optimization methods on Riemannian manifolds*. PhD thesis, Florida State University, 2011.
[10] W. Ring and B. Wirth. Optimization methods on Riemannian manifolds and their application to shape space. *SIAM Journal on Optimization*, 22(2):596–627, January 2012.
[11] D. T. Robinson. *Functional data analysis and partial shape matching in the square root velocity framework*. PhD thesis, Florida State University, 2012.
[12] A. Srivastava, E. Klassen, S. H. Joshi, and I. H. Jermyn. Shape analysis of elastic curves in Euclidean spaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(7):1415–1428, September 2011.
[13] Temple University. Shape similarity research project.
[14] S. G. Wu, F. S. Bao, E. Y. Xu, Y.-X. Wang, Y.-F. Chang, and Q.-L. Xiang. A leaf recognition algorithm for plant classification using probabilistic neural network. *2007 IEEE International Symposium on Signal Processing and Information Technology*, pages 11–16, 2007.
[15] L. Younes. Computable elastic distances between shapes. *SIAM Journal on Applied Mathematics*, 58(2):565–586, April 1998.
[16] L. Younes, P. Michor, J. Shah, and D. Mumford. A metric on shape space with explicit geodesics. *Rendiconti Lincei - Matematica e Applicazioni*, 9(1):25–57, 2008.