SIAM J. IMAGING SCIENCES
Vol. 0, No. 0, pp. 000–000


# A Novel Dynamic System in the Space of SPD Matrices with Applications to Appearance Tracking*

Guang Cheng† and Baba C. Vemuri†

**Abstract.** In this paper, we address the problem of video tracking using covariance descriptors constructed from simple features extracted from the given image sequence. Theoretically, this can be posed as a tracking problem in the space of $(n, n)$ symmetric positive definite (SPD) matrices denoted by $P_n$. A novel probabilistic dynamic model in $P_n$ based on Riemannian geometry and probability theory is presented in conjunction with a geometric (intrinsic) recursive filter for tracking a time sequence of SPD matrix measurements in a Bayesian framework. This newly developed filtering method can be used for the covariance descriptor updating problem in covariance tracking, leading to new and efficient video tracking algorithms. To show the accuracy and efficiency of our tracker in comparison to the state-of-the-art, we present synthetic experiments on $P_n$ and several real data experiments for tracking in video sequences.

**Key words.** intrinsic recursive filter, Riemannian geometry, space of symmetric positive definite matrices, covariance descriptor

**AMS subject classifications.** AUTHOR MUST PROVIDE

**DOI.** 10.1137/110853376

## 1. Introduction.

The space of symmetric positive definite (SPD) matrices denoted as $P_n = \{\mathbf{X} = (x_{ij})_{1\leq i,j\leq n} | \mathbf{X} = \mathbf{X}^T \ \forall \mathbf{v} \in R^n, \mathbf{v} \neq 0, \mathbf{v}^T\mathbf{X}\mathbf{v} > 0\}$ is very important as a feature space in the area of computer vision and its applications. Several kinds of features lie in $P_n$, such as covariance matrices, diffusion tensors in medical imaging, Cauchy deformation tensors in mechanics, etc. Unlike Euclidean space, $P_n$ is a Riemannian manifold but not a vector space. Many operations and algorithms in Euclidean space cannot be applied directly to $P_n$, and this has led to a flurry of research activity in the recent past. Several operations in Euclidean space have been extended to Riemannian manifolds. For example, the extension of the arithmetic mean to a Riemannian manifold is called the Karcher mean [14]; the extension of principal component analysis (PCA) is called the principle geodesic analysis [10, 11]; the mean-shift algorithm [8] has also been extended to Riemannian manifolds [31]. However, for filtering operations in dynamic scenes such as the popular Kalman filter [29], an intrinsic extension does not exist in the literature to date.

Recursive filtering is a technique to reduce the noise in the measurements by using the theory of recursion applied to filtering. It is often used in time sequence data analysis, especially in the tracking problem, where the model of the target needs to be updated based on the measurement and previous tracking results. Many recursive filtering techniques have been de-

*Received by the editors October 31, 2011; accepted for publication (in revised form) December 28, 2012; published electronically DATE.
http://www.siam.org/journals/siims/x-x/85337.html
†Department of Computer & Information Science & Engineering, University of Florida, Gainesville, FL 32611 (gcheng@cise@ufl.edu, vemuri@cise.ufl.edu). The second author's research was supported by NIH grant NS066340.

veloped in Euclidean space, such as the Kalman filter, the extended Kalman filter, etc., where the inputs and outputs of the filter are all vectors [29]. However, several tracking problems are naturally set in $P_n$, a Riemannian symmetric space [13]. Recent work reported in [26] on covariance tracking uses a covariance matrix (constructed from pixelwise features inside the object region) that belongs to $P_n$ in order to describe the appearance of the target being tracked. This covariance descriptor has proved to be robust in both video detection [35, 33] and tracking [26, 24, 39, 18, 36, 15, 19, 5]. The covariance descriptor is a compact feature representation of the object with relatively low dimension compared to other appearance models, such as the histogram model in [9]. In [34] an efficient algorithm for generating covariance descriptors from feature vectors is reported based on the integral image technique, which makes it possible to use covariance descriptors in real time video tracking and surveillance.

One major challenge in covariance tracking is how to recursively estimate the covariance template (a covariance descriptor that serves as the target appearance template) based on the input video frames. In [26] and also in [24, 19] the Karcher mean of sample covariance descriptors from a fixed number of video frames is used as the covariance template. This method is based on the natural Riemannian distance—the $GL$-invariant distance (section 2.1) in $P_n$. Currently, this Karcher mean can not be computed in closed form, and the computation is achieved using a gradient based optimization technique which is inefficient especially when the input contains a large number of samples. To overcome this efficiency problem, a Log-Euclidean metric was used in [18, 15], an arithmetic mean like method was used in [39], and a recursive filter for linear systems in $P_n$ was developed in [36]. However, none of these is intrinsic because they adopt methods which are extrinsic to $P_n$.

Recently, some methods were reported addressing the recursive filtering problem on Riemannian manifolds other than $P_n$. For example, the geometric particle filter for handling two-dimensional affine motions (2-by-2 nonsingular matrix) was reported in [25, 17, 15], and an extension to Riemannian manifolds was developed in [28]. However, since the covariance descriptor is usually a high-dimensional descriptor, e.g., the degrees of freedom of a $5 \times 5$ covariance matrix are 15, the number of samples required for the particle filter would be quite large in this case. Additionally, computing the intrinsic (Karcher) mean on $P_n$ is computationally expensive for large sample sizes. Thus, using an intrinsic particle filter to update covariance descriptor would be computationally expensive for the tracking problem. There are also existing tracking methods on Grassmann manifolds [30, 7]. However, it is nontrivial to extend these to $P_n$, since Grassmann manifolds and $P_n$ have very different geometric properties; e.g., Grassmann manifolds are compact and have a nonnegative sectional curvature when using an invariant Riemannian metric [38], while $P_n$ is noncompact and has nonpositive sectional curvature when using an invariant (to the general linear group ($GL$)) Riemannian metric [13].

In this paper, we focus on the problem of developing an intrinsic recursive filter—abbreviated IRF for the rest of this paper—on $P_n$. A novel probabilistic dynamic model on $P_n$ based on Riemannian geometry and probability theory is presented. Here, the noisy state and observations are described by matrix-variate random variables whose distribution is a generalized normal distribution on $P_n$ based on the $GL$-invariant measure. In [23, 16] the authors provide a linear approximation of this distribution for cases when the variance of the distribution is very small. In contrast, in this paper, we explore several properties of this distribution for

the arbitrary variance case. We then develop the IRF based on this novel dynamic model and the Bayesian framework with a moving window approximation presented in [7] which tracks modes of the distribution (for details, see section 3). By applying this recursive filter—to achieve covariance tracking—in conjunction with an existing particle position tracker [2], we obtain a new efficient real time video tracking algorithm described in section 3.2. We present experiments with comparisons to existing state-of-the-art methods and quantitative analysis that support the effectiveness and efficiency of the proposed algorithm.

The remainder of this paper is organized as follows. In section 2, we introduce the probabilistic dynamic model on $P_n$ after presenting some background Riemannian geometry and an invariant probability measure. Then the IRF and the tracking algorithms are presented in section 3, followed by the experiments in section 4. Finally we draw conclusions in section 5.

## 2. IRF: A new dynamic tracking model on $P_n$.

**2.1. Riemannian geometry on $P_n$.** In this section, we briefly introduce the basic tools of Riemannian geometry for $P_n$ and then motivate the use of the $GL$-invariant metric on $P_n$ for developing our new dynamic model. We refer the reader to [21, 13, 32] for details. Following this, we contrast the popularly used Log-Euclidean framework against the intrinsic framework for developing the dynamic recursive filter proposed in this paper. This provides the necessary motivation for an IRF.

$P_n$ is the space of $n \times n$ SPD matrices, which is a Riemannian manifold. It can be identified with the quotient space $GL(n)/O(n)$ [32], where $GL(n)$ denotes the general linear group, the group of $(n \times n)$ nonsingular matrices, and $O(n)$ is the orthogonal group, the group of $(n \times n)$ orthogonal matrices. This makes $P_n$ a homogeneous space with $GL(n)$ as the group that acts on it and the group action defined for any $\mathbf{X} \in P_n$ by $\mathbf{X}[\mathbf{g}] = \mathbf{g}\mathbf{X}\mathbf{g}^t$. One can now define $GL$-invariant quantities such as the $GL$-invariant inner product based on the group action defined above. We will now begin with inner product in the tangent space of $P_n$. For tangent vectors $\mathbf{U}$ and $\mathbf{V} \in T_{\mathbf{X}}P_n$ (the tangent space at point $\mathbf{X}$, which is the space of symmetric matrices of dimension $(n+1)n/2$ and a vector space) the $GL$-invariant inner product is defined as $\forall \mathbf{g} \in GL(n)$, $\langle \mathbf{U}, \mathbf{V} \rangle_{\mathbf{X}} = \langle \mathbf{g}\mathbf{U}\mathbf{g}^t, \mathbf{g}\mathbf{V}\mathbf{g}^t \rangle_{\mathbf{g}\mathbf{X}\mathbf{g}^t}$. On $P_n$ this $GL$-invariant inner product takes the form

$$\langle \mathbf{U}, \mathbf{V} \rangle_X = \mathrm{tr}(\mathbf{X}^{-1/2}\mathbf{U}\mathbf{X}^{-1}\mathbf{V}\mathbf{X}^{-1/2}). \tag{2.1}$$

With metric/inner product defined on the manifold, the length of any curve in $P_n$, $\gamma : [0,1] \to P_n$ is defined as $length(\gamma)^2 = \int_0^1 \langle \dot{\gamma}, \dot{\gamma} \rangle_{\gamma(t)} dt$. The distance between any $\mathbf{X}, \mathbf{Y} \in P_n$ is defined as the length of the shortest curve between $\mathbf{X}$ and $\mathbf{Y}$ (geodesic distance). With the $GL$-invariant metric, the distance between $\mathbf{X}, \mathbf{Y} \in P_n$ is given by (see [32])

$$dist(\mathbf{X}, \mathbf{Y})^2 = \mathrm{tr}(\log^2(\mathbf{X}^{-1}\mathbf{Y})), \tag{2.2}$$

where log is the matrix log operator. Since this distance is induced from the $GL$-invariant metric in (2.1), it is naturally $GL$ invariant, i.e., $dist^2(\mathbf{X}, \mathbf{Y}) = dist^2(\mathbf{g}\mathbf{X}\mathbf{g}^t, \mathbf{g}\mathbf{Y}\mathbf{g}^t)$.

With $GL$-invariant metric defined on $P_n$, the intrinsic or Karcher mean of a set of elements $\mathbf{X}_i \in P_n$ can be computed by performing the following minimization:

$$\mu* = \underset{\mu}{\mathrm{argmin}} \sum_i dist^2(\mathbf{X}_i, \mu) \tag{2.3}$$

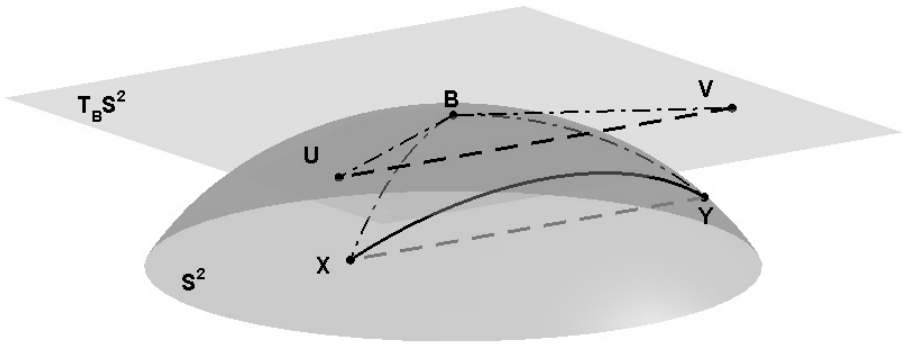


**Figure 1.** *An example of different distances on $S^2$.*

using a gradient based technique.

The Log and Exponential maps [13] are very useful tools on the Riemannian manifold. The Exponential map denoted as $\mathrm{Exp}_{\mathbf{X}}(\cdot)$, where $\mathbf{X} \in P_n$, maps a vector rooted at the origin of the tangent space $T_{\mathbf{X}}P_n$ to a geodesic emanating from $\mathbf{X}$. The Log map ($\mathrm{Log}_{\mathbf{X}}(\cdot)$) is the inverse of the Exponential map. The Exponential and Log map on $P_n$ are given by

$$\begin{aligned} \mathrm{Exp}_{\mathbf{X}}(\mathbf{V}) &= \mathbf{X}^{1/2}\exp(\mathbf{X}^{-1/2}\mathbf{V}\mathbf{X}^{-1/2})\mathbf{X}^{1/2}, \\ \mathrm{Log}_{\mathbf{X}}(\mathbf{Y}) &= \mathbf{X}^{1/2}\log(\mathbf{X}^{-1/2}\mathbf{Y}\mathbf{X}^{-1/2})\mathbf{X}^{1/2}, \end{aligned} \tag{2.4}$$

where $\mathbf{X}, \mathbf{Y} \in P_n$, $\mathbf{V} \in T_{\mathbf{X}}P_n$, and log and exp denote the matrix exp and log operators.

**2.1.1. Geodesic, Euclidean, and Log-Euclidean distances.** To illustrate the differences between geodesic, Euclidean, and Log-Euclidean distances on a Riemannian manifold, we have a simple example on $S^2$—the unit 2-sphere depicted in Figure 1. Given two points $\mathbf{X}, \mathbf{Y} \in S^2$, the geodesic distance is the length of the shorter arc of the great circle between $\mathbf{X}$ and $\mathbf{Y}$—the solid black line. The Euclidean distance between them is the length of the straight line between $\mathbf{X}$ and $\mathbf{Y}$ in the embedded three-dimensional Euclidean space—the dashed grey line. Given another point $\mathbf{B} \in S^2$, we could project $\mathbf{X}, \mathbf{Y}$ to the tangent space at $\mathbf{B}$, $T_{\mathbf{B}}S^2$ by the Log map. In Figure 1, the points after projection are $\mathbf{U}, \mathbf{V}$. The length of the line $\mathbf{UV}$ (the dashed black line) can then be used as the distance between $\mathbf{X}, \mathbf{Y}$, which is called Log-Euclidean distance. Note that if $\mathbf{B} = \mathbf{X}$ or $\mathbf{B} = \mathbf{Y}$, the Log-Euclidean and geodesic distances are the same. Otherwise, they are different in Riemannian manifolds except for certain manifolds like Euclidean space.

The Euclidean and Log-Euclidean distances are extrinsic to the manifold, i.e., depend on the embedding Euclidean space and a predefined base point $\mathbf{B}$ for the Log-Euclidean distance. Therefore, they are often called extrinsic distances, while the geodesic distance which depends only on the manifold is called intrinsic distance. In [1], the base point $\mathbf{B}$ is fixed at the identity element of the space. However, so long as there is a predefined base point $\mathbf{B} \neq \mathbf{X}$ or $\mathbf{Y}$ used to compute the distance between $\mathbf{X}$ and $\mathbf{Y}$ as shown in Figure 1, the framework can still be viewed as Log-Euclidean or extrinsic, especially when this distance serves the purpose of a

cost function being optimized. In [36], the estimation error is measured using a arbitrarily chosen base point $\mathbf{B}$, which makes it a Log-Euclidean method, even though the base point is not fixed.

In $P_n$, the intrinsic/geodesic distance is based on the $GL$-invariant metric, as shown in section 2.1. Euclidean and Log-Euclidean distances can also be viewed as being based on corresponding metrics which are invariant to $O(n)$. So, which metric should we choose for the covariance tracking problem? There are two primary reasons for the choice of a $GL$-invariant metric over the conventional Euclidean metric.

First of all, $P_n$ is an open subset of the corresponding Euclidean space $R^{(n+1)n/2}$, which implies that $P_n$ would be incomplete with a Euclidean metric, since it is possible to find a Cauchy sequence which might not converge for this case. This implies that for some of the optimization problems set in $P_n$, the optimum cannot be achieved inside $P_n$. This in turn means that the covariance updates could lead to matrices that are not covariance matrices, an unacceptable situation in practice. This problem will not arise when using the $GL$-invariant metric, since the space of $P_n$ is geodesically complete with a $GL$-invariant metric [32].

Second, the feature vectors in general might contain components of different scales and from disparate sources, e.g., object position, color, etc. In this case, a normalization of the (in general) unknown scales of different components would be necessary when using Euclidean distance, which is nontrivial and may lead to use of ad hoc methods. However, with a $GL$-invariant metric, this scaling issue does not arise, since the presence of different scales for the elements of a feature vector from which the covariance matrix is constructed is equivalent to multiplication of the covariance matrix with a positive definite diagonal matrix. This operation is a $GL$ group operation, and since $GL$ invariance implies invariance to $GL$ group operations, this scaling issue is a nonissue when using a $GL$-invariant metric.

The Log-Euclidean metric can be viewed as a linear approximation of the $GL$-invariant metric. This approximation has lower error when in a small neighborhood of the predefined base point. But when computed in a large region around the base/anchor point, it will suffer from high approximation error, as shown in our variance computation in section 2.2.3 and also the result in the synthetic experiment in section 4.1, which would accumulate with each frame and affect the tracking result, especially for noisy data. Also, the Log-Euclidean metric as an approximation does not preserve some of the good properties of the $GL$-invariant metric, such as the scale invariance, as mentioned above.

The $GL$-invariant metric, on the other hand, is more difficult to compute. Many $GL$-invariant computations do not have closed form solutions and thus are less efficient to compute. However, the proposed filter in this paper could be computed with a computational effort similar to that of the filter in [36] but is more accurate for larger amounts of noise.

### 2.2. Invariant measure and generalized normal distribution on $P_n$.

**2.2.1. Probability measures on $P_n$.** To define a probability distribution on a manifold, first we need to define a measure on the manifold. In this paper, we use the $GL$-invariant measure $[d\mathbf{X}]$ on $P_n$. $GL$ invariance here implies $\forall \mathbf{g} \in GL(n)$ and $\forall \mathbf{X} \in P_n$, $[d\mathbf{g}\mathbf{X}\mathbf{g}^t] = [d\mathbf{X}]$. From [32], we know that $[d\mathbf{X}] = |\mathbf{X}|^{-(n+1)/2} \prod_{1 \le i \le j \le n} dx_{ij}$, where $x_{ij}$ is the element in the $i$th row and $j$th column of the SPD matrix $\mathbf{X}$. Also, this measure is consistent with the $GL$-invariant metric defined on $P_n$ defined earlier and also presented in [23].

Similar to the Karcher mean, the Karcher expectation for the random variable $\mathbf{X}$ on any Riemannian manifold $M$ can be defined as the result of the following minimization problem:

$$\bar{\mathbf{X}} = E(\mathbf{X}) = \underset{\mathbf{Y}\in M}{\operatorname{argmin}} \int_M dist^2(\mathbf{X}, \mathbf{Y}) d\mu(\mathbf{X}), \tag{2.5}$$

where $\bar{\mathbf{X}}$ denotes the expectation of random variable $\mathbf{X}$ and $\mu(\mathbf{X})$ is the probability measure defined on $M$. Similarly, the variance can be defined based on this expectation by

$$Var(\mathbf{X}) = \int_M dist^2(\mathbf{X}, \bar{\mathbf{X}}) d\mu(\mathbf{X}). \tag{2.6}$$

Note that, in Euclidean space $R^m$, which is also a Riemannian manifold, the Karcher expectation is equivalent to the traditional definition of expectation, and the variance in (2.6) is the trace of the covariance matrix. In $P_n$, by taking the gradient of the energy function in (2.5) and setting it to zero, we find that the expectation of the random variable will satisfy the following equation:

$$\int_{P_n} \log(\bar{\mathbf{X}}^{-1/2}\mathbf{X}\bar{\mathbf{X}}^{-1/2}) p(\mathbf{X})[d\mathbf{X}] = 0. \tag{2.7}$$

**2.2.2. Generalized normal distribution on $P_n$.** The generalization of the normal distribution to $P_n$ used in this paper is defined as follows:

$$dP(\mathbf{X}; \mathbf{M}, \omega^2) = p(\mathbf{X}; \mathbf{M}, \omega^2)[d\mathbf{X}] = \frac{1}{Z} \exp\left( - \frac{dist(\mathbf{X}, \mathbf{M})^2}{2\omega^2} \right) [d\mathbf{X}], \tag{2.8}$$

where $P(\cdot)$ and $p(\cdot)$ are the probability distribution and density, respectively, of the matrix-variate random variable $\mathbf{X} \in P_n$, with two parameters $\mathbf{M} \in P_n$ and $\omega^2 \in R^+$, and $Z$ is the scalar normalization factor. $dist(\cdot)$ is defined in (2.2). As shown in [23], this distribution has minimum information given the Karcher mean and variance. That is, in the absence of any information this distribution would be the best possible assumption from an information theoretic viewpoint. Also, this distribution is different from the Log-normal distribution which was used in [36, 27]. Actually, the two distributions have very similar densities, but the density used in this paper is based on $GL$-invariant measure while Log-normal density is based on the Lebesgue measure in Euclidean space.

A very important property of the above generalized normal distribution is summarized in the following theorem, whose proof is given in Appendix A.

Theorem 2.1. *The normalization factor $Z$ in (2.8) is a finite constant with respect to parameter $\mathbf{M} \in P_n$.*

The consequence of Theorem 2.1 is that if the prior and the likelihood are both based on the generalized normal distribution defined using the $GL$-invariant measure, computing the mode of the posterior density can be achieved by minimizing the sum of squared $GL$-invariant distances from the unknown expectation of the given samples.

One direct consequence of Theorem 2.1 is the following corollary, whose proof is given in Appendix A.

Corollary 2.2. *Given a set of independent and identically distributed (i.i.d) samples* $\{\mathbf{X_i}\}$ *drawn from the distribution* $dP(\mathbf{X};\mathbf{M},\omega^2)$*, the MLE of the parameter* $\mathbf{M}$ *is the Karcher mean of all samples.*

From Theorem 2.1 we know that the normalization factor $Z$ in (2.8) is a function of $\omega$. The integral in (A.2) is nontrivial, and currently no exact solution is available for arbitrary $n$. For $n = 2$ we have

$$\begin{aligned} Z_2(\omega) &= 2c_2 \int_{-\infty}^{\infty}\int_{-\infty}^{y_1} \exp\Big(-\sum_{i=1}^{2}\Big(\frac{1}{2\omega^2}y_i^2+\frac{1}{2}y_i\Big)\Big)(\exp(y_1)-\exp(y_2))dy_2dy_1 \\ &= \sqrt{2\pi}c_2\omega\exp\Big(\frac{1}{4}\omega^2\Big)\int_{-\infty}^{\infty}\exp\Big(-\frac{(y_1-0.5\omega^2)^2}{2\omega^2}\Big)\Big(1+erf\Big(\frac{y_1+0.5\omega^2}{\sqrt{2\omega^2}}\Big)\Big) \\ &\quad -\exp\Big(-\frac{(y_1+0.5\omega^2)^2}{2\omega^2}\Big)\Big(1+erf\Big(\frac{y_1-0.5\omega^2}{\sqrt{2\omega^2}}\Big)\Big)dy_1 \\ &= 4\pi c_2\omega^2\exp\Big(\frac{1}{4}\omega^2\Big)erf\Big(\frac{\omega}{2}\Big), \end{aligned} \tag{2.9}$$

where $erf(x) = \frac{2}{\sqrt{\pi}}\int_0^x \exp(-t^2)dt$ is the error function.

**2.2.3. Mean and the variance of the generalized normal distribution.** Similar to the normal distribution in Euclidean space, the mean and variance of the generalized normal distribution on $P_n$ in (2.8) are controlled by the parameters $\mathbf{M}$ and $\omega^2$, respectively. The relation between $\mathbf{M}$ and $dP(\mathbf{X};\mathbf{M},\omega^2)$ is given by the following theorem, whose proof is given in Appendix B.

Theorem 2.3. *Parameter* $\mathbf{M}$ *is the Karcher expectation of the generalized normal distribution* $dP(\mathbf{X};\mathbf{M},\omega^2)$.

The variance of $dP(\mathbf{X};\mathbf{M},\omega^2)$ is controlled by the parameter $\omega^2$. Unlike the multivariate normal distribution in Euclidean space, where the Karcher variance (see (2.6)) is equal to $n\omega^2$, the relation between the variance and $\omega^2$ of the generalized normal distribution is much more complex. Without loss of generality we assume $\mathbf{X} \in P_n$ is a matrix variate random variable from $dP(\mathbf{X};\mathbf{I},\omega^2)$. The variance $Var(\mathbf{X}) = \frac{1}{Z}\int_{P_n}||\log(\mathbf{X})||^2\exp(-\frac{||\log(\mathbf{X})||^2}{2\omega^2})[d\mathbf{X}]$. As in (A.2), by using the Polar coordinates and taking the log of the eigenvalues, we can get

$$Var(\mathbf{X}) = \omega^2 Var_q(\mathbf{y}), \tag{2.10}$$

where $\mathbf{y}$ is a random vector in $R^n$ having a distribution with density function,

$$q(\mathbf{y}) = \frac{1}{z(\omega)}\exp\Big(-\frac{1}{2}\sum_i y_i^2\Big)\prod_{1\le i<j\le n} 2\Big|\sinh\Big(\frac{\omega(y_i-y_j)}{2}\Big)\Big|, \tag{2.11}$$

where $z(\omega)$ is the normalization factor. Currently, there are no analytic solutions for $Var_q(\mathbf{y})$ for arbitrary $n$. When $n = 2$ we can compute $Var(\mathbf{y})$ using a technique similar to that in (2.9):

$$Var_q(\mathbf{y}) = \frac{\omega}{\sqrt{\pi}\exp(\frac{1}{4}\omega^2)erf(\frac{\omega}{2})} + 2\Big(1+\frac{\omega^2}{4}\Big). \tag{2.12}$$

From (2.12) we can find that in $P_2$ when $\omega$ is close to zero, $Var(\mathbf{X}) \approx 3\omega^2$, and when $\omega$ is large, $Var(\mathbf{X}) \approx \frac{\omega^4}{2}$. This is because $P_n$ can be locally approximated by a Euclidean space. When $\omega$ is close to zero, the major portion of the distribution would be in a small region in $P_n$, where Euclidean approximation is relatively accurate. Hence, $Var(\mathbf{X})$ is proportional to $\omega^2$, which is similar to the normal distribution in Euclidean space. When $\omega^2$ is not close to zero, Euclidean approximation is no longer accurate, and the $Var(\mathbf{X})$ becomes a complicated function of $\omega^2$. This property has been used to get the approximation of the generalized normal distribution with small $\omega^2$ in [23, 16].

The following two theorems show that the above stated approximations will still be satisfied for $n > 2$, whose proofs are given in Appendices C and D, respectively.

Theorem 2.4.

$$\lim_{\omega \to 0} \frac{Var(\mathbf{X})}{\omega^2} = \frac{n(n+1)}{2}. \tag{2.13}$$

Note that this theorem can also be obtained using the approximation of the generalized normal distribution with small $\omega^2$ in [23, 16]. Furthermore, from the proof in Appendix C, we can deduce that since the Log-normal is a projection of a normal distribution from the tangent space (can be identified with $Sym(n)$) to $P_n$, and here the random vector $\mathbf{y}$ is the normalized log of the eigenvalues of $\mathbf{X}$, we can see that *when $\omega$ is close to zero, the generalized normal distribution can be approximated by a Log-normal distribution.*

Theorem 2.5.

$$\lim_{\omega \to \infty} \frac{Var(\mathbf{X})}{\omega^4} = \frac{(n^3 - n)}{12}. \tag{2.14}$$

**2.3. The probabilistic dynamic model on $P_n$.** To perform tracking on $P_n$, obviously the observation $\mathbf{Y}_k$ at frame $k$ lies on $P_n$. Also, we can define the state to lie on $P_n$, i.e., $\mathbf{X}_k \in P_n$. The state transition model and the observation model can then be defined as

$$p(\mathbf{X}_k|\mathbf{X}_{k-1}) = \frac{1}{Z_s}\exp\left( - \frac{dist^2(\mathbf{X}_k, \mathbf{g}\mathbf{X}_{k-1}\mathbf{g}^t)}{2\omega^2}\right), \tag{2.15}$$

$$p(\mathbf{Y}_k|\mathbf{X}_k) = \frac{1}{Z_o}\exp\left( - \frac{dist^2(\mathbf{Y}_k, \mathbf{h}\mathbf{X}_k\mathbf{h}^t)}{2\phi^2}\right), \tag{2.16}$$

where $\mathbf{g}, \mathbf{h} \in GL(n)$. $\omega^2, \phi^2 > 0$ are the parameters that control the variance of the state transition and the observation noise. The above two densities are both based on the $GL$-invariant measure on $P_n$, unlike in [36, 27], where they are based on the Lebesgue measure. The key implication of this is that the normalization factor in the densities is a constant for the $GL$-invariant measure and not so for the Lebesgue measure case. If the normalization factor is not a constant, one does not have a valid density.

## 3. IRF-based tracking algorithm on $P_n$.

**3.1. The Bayesian tracking framework.** For simplicity, we use the Bayesian tracking framework described in [7] here. The tracking problem can be viewed as, given a time sequence of observations $\underline{\mathbf{Y}}_s = \{\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_s\}$ from time 1 to time $s$, how can one compute the state

$\mathbf{X}_s$ at time $s$? To solve this problem, first we make two assumptions: (1) The state transition is Markovian, i.e., the state $\mathbf{X}_s$ depends only on $\mathbf{X}_{s-1}$, or say

$$p(\mathbf{X}_s|\underline{\mathbf{X}}_{s-1}, \underline{\mathbf{Y}}_{s-1}) = p(\mathbf{X}_s|\mathbf{X}_{s-1}). \tag{3.1}$$

(2) The observation $\mathbf{Y}_s$ is dependent only on the state $\mathbf{X}_s$ at the current time point $s$, i.e.,

$$p(\mathbf{Y}_s|\underline{\mathbf{X}}_s, \underline{\mathbf{Y}}_{s-1}) = p(\mathbf{Y}_s|\mathbf{X}_s). \tag{3.2}$$

Thus, $p(\mathbf{X}_s|\mathbf{X}_{s-1})$ is called the state transition model and $p(\mathbf{Y}_s|\mathbf{X}_s)$ is called the observation model.

The goal of tracking can thus be viewed as computing the posterior $p(\underline{\mathbf{X}}_s|\underline{\mathbf{Y}}_s)$. First we have

$$p(\underline{\mathbf{X}}_s|\underline{\mathbf{Y}}_s) = p(\underline{\mathbf{X}}_s, \underline{\mathbf{Y}}_s)/p(\underline{\mathbf{Y}}_s) \propto p(\underline{\mathbf{X}}_s, \underline{\mathbf{Y}}_s) \tag{3.3}$$

and

$$\begin{aligned} p(\underline{\mathbf{X}}_s, \underline{\mathbf{Y}}_s) &= p(\mathbf{Y}_s|\mathbf{X}_s)p(\mathbf{X}_s|\mathbf{X}_{s-1})p(\underline{\mathbf{X}}_{s-1}, \underline{\mathbf{Y}}_{s-1}) \\ &= \prod_{k=1}^{s} p(\mathbf{Y}_k|\mathbf{X}_k)p(\mathbf{X}_k|\mathbf{X}_{k-1}). \end{aligned}$$

By using a moving window approximation and setting the window width to be 2, we can then compute $\hat{\mathbf{X}}_k, \hat{\mathbf{X}}_{k-1}$ by solving the following optimization problem:

$$\begin{aligned} \hat{\mathbf{X}}_k, \hat{\mathbf{X}}_{k-1} &= \underset{\mathbf{X}_k, \mathbf{X}_{k-1}}{\operatorname{argmax}} \prod_{j=k-1}^{k} p(\mathbf{Y}_j|\mathbf{X}_j)p(\mathbf{X}_j|\mathbf{X}_{j-1}) \\ &= \underset{\mathbf{X}_k, \mathbf{X}_{k-1}}{\operatorname{argmin}} E_k(\mathbf{X}_k, \mathbf{X}_{k-1}), \end{aligned}$$

where

$$\begin{aligned} E_k(\mathbf{X}_k, \mathbf{X}_{k-1}) = \phi^{-2} dist^2(\mathbf{h^{-1}Y_k h^{-t}}, \mathbf{X_k}) + \omega^{-2} dist^2(\mathbf{gX_{k-1}g^t}, \mathbf{X_k}) \\ + \phi^{-2} dist^2(\mathbf{h^{-1}}\mathbf{Y}_{k-1}\mathbf{h}^{-t}, \mathbf{X}_{k-1}) + \omega^{-2} dist^2(\mathbf{X}_{k-1}, \mathbf{gX}_{k-2}\mathbf{g^t}). \end{aligned} \tag{3.4}$$

Thus $E_k$ is the energy function we would like to optimize at each frame $k$. Upon a closer look, we get the following theorem on the geodesic convexity of $E_k$ and whose proof is given in Appendix E.

Theorem 3.1. *The energy function $E_k$ in (3.4) is geodesically convex on the Riemannian manifold $P_n \times P_n$, where $\times$ denotes the Cartesian product.*

A function $f : P_n \mapsto R$ being geodesically convex implies, for all geodesics on $\gamma : [0, 1] \mapsto P_n$, the composition $f \circ \gamma$ is convex [37]. Geodesic convexity is an extension of the standard notion of convexity to the Riemannian manifold. It is not hard to show that both convexities share most of the properties, such as the local minimum being equivalent to the global minimum, etc. More details on the properties can be found in [37].

By taking the gradient of $E_k$ with respect to $\mathbf{X}_k$, we can find that at the local minimum (3.5) will be satisfied:

$$\phi^{-2}\mathrm{Log}_{\mathbf{X}_k}(\mathbf{h^{-1}Y_k h^{-t}}) + \omega^{-2}\mathrm{Log}_{\mathbf{X_k}}(\mathbf{gX}_{k-1}\mathbf{g}^t) = 0. \tag{3.5}$$

This means that when reaching the optimum, $\mathbf{X}_k$ will be on the geodesic of $\mathbf{h^{-1}Y}_k\mathbf{h^{-t}}$ and $\mathbf{gX}_{k-1}\mathbf{g}^t$, and $\frac{dist(\mathbf{h^{-1}Y}_k\mathbf{h^{-t}},\mathbf{X}_k)}{dist(\mathbf{gX}_{k-1}\mathbf{g}^t,\mathbf{X}_k)} = \frac{\phi^2}{\omega^2}$. Thus at the optimum point we will have

$$\begin{aligned}\phi^{-2}dist^2(\mathbf{h^{-1}Y_k h^{-t}}, \mathbf{X_k}) + \omega^{-2}dist^2(\mathbf{gX_{k-1}g^t}, \mathbf{X_k})\\ = \frac{1}{\phi^2+\omega^2}dist^2(\mathbf{h^{-1}Y_k h^{-t}}, \mathbf{gX_{k-1}g^t}).\end{aligned} \tag{3.6}$$

Combining (3.6) and (3.4), we can get

$$\begin{aligned}E'_k(\mathbf{X}_{k-1}) = \frac{1}{\phi^2+\omega^2}dist^2(\mathbf{h^{-1}Y_k h^{-t}}, \mathbf{gX_{k-1}g^t})\\ + \phi^{-2}dist^2(\mathbf{h^{-1}Y}_{k-1}\mathbf{h}^{-t}, \mathbf{X}_{k-1}) + \omega^{-2}dist^2(\mathbf{X}_{k-1}, \mathbf{gX}_{k-2}\mathbf{g}^t).\end{aligned} \tag{3.7}$$

It is obvious that $E'_k$ and $E_k$ have the same optimum. Instead of minimizing $E_k$, we can minimize $E'_k$, which is a weighted Karcher mean of three points on $P_n$. The classical gradient decent algorithm on $P_n$ [21] can efficiently solve this problem.

After getting the optimal $\mathbf{X}^*_{k-1}$, $\mathbf{X}^*_k$ can be computed in a closed form:

$$\mathbf{X}^*_k = (\mathbf{h^{-1}Y_k h^{-t}})^{1/2}[(\mathbf{h^{-1}Y_k h^{-t}})^{-1/2}\mathbf{gX^*_{k-1}g^t}(\mathbf{h^{-1}Y_k h^{-t}})^{-1/2}]^{\frac{\phi^2}{\phi^2+\omega^2}}(\mathbf{h^{-1}Y_k h^{-t}})^{1/2}. \tag{3.8}$$

It is easy to show that the state update here is an estimation of the mode of the posterior $p(\underline{\mathbf{X}}_s|\underline{\mathbf{Y}}_s)$, which is different from the usual Kalman filter and particle filter methods, where the state update is the mean of the posterior $p(\mathbf{X}_s|\underline{\mathbf{Y}}_s)$. In the proposed update process, the covariance of the posterior is not necessary for updating the state. We do not provide an update of the covariance here, partly because the covariance update is hard to compute for this distribution on $P_n$. Actually, there is no existing closed form solution for the covariance matrices even for the distribution $p(\mathbf{X}_k|\mathbf{X}_{k-1}) = \frac{1}{Z_s}\exp(-\frac{dist^2(\mathbf{X}_k,\mathbf{gX}_{k-1}\mathbf{g}^t)}{2\omega^2})$. In our future work, we will focus on developing an efficient and convergent covariance updating mechanism in this framework.

**3.2. The tracking algorithm.** The intrinsic recursive filter (IRF) for covariance matrices (descriptors) on $P_n$ presented above can be used in combination with many existing tracking techniques. Many algorithms based on covariance descriptors like those in [26, 36] can use our IRF as the model updating method for covariance descriptors. In this paper we combine the IRF with an existing particle position tracker [2] and get a real-time video tracking algorithm.

**3.2.1. Feature extraction.** Assume we have an rectangular region $R$ with width $W$ and height $H$ which represents the target object in a certain image $I$ in the video sequence. The feature vector $f(x, y)$, where $(x, y) \in R$, can be extracted to include the information of appearance, position, etc., to describe information at the point $(x, y)$. In [26], the feature vector was chosen to be $\mathbf{f} = [x, y, I(x, y), |I_x(x, y)|, |I_y(x, y)|]$, where $I_x$ and $I_y$ are the components

of the gradient $\nabla I$. For color images, $I(x, y) = [R, G, B]$ is a vector. With the feature vectors at each point in the region of the object, the covariance matrix can be computed as $C_R = \frac{1}{WH}\sum_{k\in R}(\mathbf{f}_k - \mu_R)(\mathbf{f}_k - \mu_R)^t$. This covariance matrix can be computed in constant time with respect to the size of the region $R$ by using the technique called the integral image, as was done in [34]. We can also add the mean $\mu_R$ into the covariance descriptor and still obtain an SPD matrix in the following manner:

$$\hat{C_R} = \begin{bmatrix} C_R + \lambda^2\mu\mu^t & \lambda\mu \\ \lambda\mu^t & 1 \end{bmatrix}, \tag{3.9}$$

where $\lambda$ is a parameter used to balance the effect of the mean and variance in the descriptor (in this paper $\lambda = 0.001$).

As in [34], we use several covariance descriptors for each object in the scene. Very briefly, each region enclosing an object is divided into five regions, and in each of these, a covariance descriptor is computed and tracked individually. A matching score (likelihood) is computed using four of them with relatively small distance to the corresponding template in the template matching stage described below. This approach is used in order to increase the robustness of our algorithm.

**3.2.2. Tracking and template matching.** We use a sampling importance resampling (SIR) particle filter [2] as a position and velocity tracker. The state vector of the particle filter at the $k$th frame is now given by $\mathbf{u}_k = (x_k, y_k, v_{x,k}, v_{y,k}, \log(s_k))^t$, where $x_k, y_k, v_{x,k}.v_{y,k}$ denote the position and velocity of the object in the two-dimensional image, and $\log(s_k)$ is the log of the scale. The state transition is defined by the equation below:

$$\mathbf{u}_k = \mathbf{F}\mathbf{u}_{k-1} + \mathbf{n}, \tag{3.10}$$

where the state transition matrix $\mathbf{F}$ is defined based on Newton's first law:

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & 0.01 & 0 & 0 \\ 0 & 1 & 0 & 0.01 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.11}$$

The additive noise $\mathbf{n}$ is normally distributed with the covariance matrix assumed to be a diagonal matrix, and in our work reported here, it was set to $(4^2, 4^2, 20^2, 20^2, 0.015^2)$. These state transition parameters are dependent on the videos being tracked. They could also be learned from the manually labelled training sets.

At the $k$th frame, the likelihood for the particle filter is based on the generalized normal distribution, as discussed in section 2.2:

$$p(I_k|\mathbf{u}_k) = \frac{1}{Z}\exp\left(-\frac{dist(\mathbf{Y}_k, \mathbf{h}\mathbf{X}_k\mathbf{h}^t)^2}{2\phi^2}\right), \tag{3.12}$$

where $\mathbf{Y}_k$ is the covariance descriptor extracted from the image $I_k$ based on the state vector $\mathbf{u}_k$, as described in section 3.2.1. $\mathbf{X}_k$ is the covariance template updated using our IRF.

The classical particle filter updates the weight for the sample using the following equation:

$$w_k^i \propto w_{k-1}^i \frac{p(I_k|\mathbf{u}_k^i)p(\mathbf{u}_k^i|\mathbf{u}_{k-1}^i)}{q(\mathbf{u}_k^i|\mathbf{u}_{k-1}^i, I_k)}, \tag{3.13}$$

where $w_k^i$ is the weight for the $i$th sample at frame $k$ and $q$ is the importance sampling density. Since we are using SIR, $q(\mathbf{u}_k|\mathbf{u}_{k-1}, I_k) = p(\mathbf{u}_k|\mathbf{u}_{k-1})$, and the weight update equation becomes

$$w_k^i \propto w_{k-1}^i p(I_k|\mathbf{u}_k^i). \tag{3.14}$$

At step $k$, we first compute the prediction of the object covariance template using $\hat{\mathbf{Y}}_k = \mathbf{h}\mathbf{g}\mathbf{X}_{k-1}\mathbf{g}^t\mathbf{h}^t$ and then the prediction of the position and scale of the object represented in the set of particles based on the state transition matrix (3.11). Covariance descriptors then are computed for each of the predicted particle states at the corresponding object regions. The likelihood for each covariance descriptor is computed based on the generalized normal distribution centered at the predicted corresponding covariance template. And the likelihood for each particle's state is computed as multiplication of the likelihoods of covariance descriptors that are closer to their corresponding template, as mentioned above. After multiplying the likelihood with the weight of each particle, the mean of the sample set is computed. This is followed by computation of covariance descriptors at the location of the mean of the particle set. This covariance descriptor then forms the input to our IRF. The algorithm for each frame is given in the text box below.

**Tracking Algorithm**

Step 1. Resample and draw samples following (3.10).
Step 2. Predict the covariance template at the current step using (2.15).
Step 3. Extract covariance descriptor using methods in section 3.2.1 for each particle and then update each particle weight using (3.14).
Step 4. Compute the weighted mean $\mathbf{u}_k$ of all the particles.
Step 5. Extract the covariance descriptor at $\mathbf{u}_k$ and use as observation $\mathbf{Y}_k$ to optimize the energy function in (3.7), and then update the covariance template$\mathbf{X}_k$ using (3.8).

In our paper, we use 300 particles for the particle set. Our tracking algorithm runs at around 15Hz for videos with a frame size of $352 \times 288$ on a desktop with a 2.8GHz CPU.

## 4. Experiments.

In this section, we present the results of applying our intrinsic recursive filter to both synthetic and real data sets. The real data sets were taken from standard video sequences used in the computer vision community for testing tracking algorithms. First we present the synthetic data experiments and then the real data.

### 4.1. Synthetic data experiments.

To validate the proposed filtering technique, we first performed synthetic data experiments on $P_3$, the space of $3\times3$ SPD matrices. A time sequence of i.i.d samples of SPD matrices was randomly drawn from the Log-normal distribution [27] centered at the identity matrix. This was done by first drawing samples $\{\mathbf{v}_i\}$ in $R^6$ (isomorphic to $Sym(3)$) from the normal distribution $N(0, \sigma^2\mathbf{I}_6)$. Then, these samples are projected to

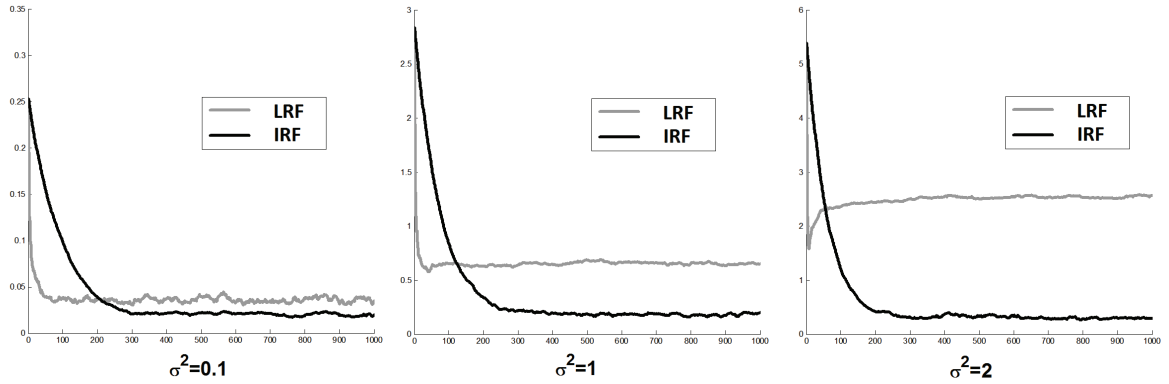


**Figure 2.** *Mean estimation error from 20 trials for the synthetic data experiment. The x-axis denotes the time step. The y-axis denotes the estimation error measured using the Riemannian distance between the estimates and the ground truth. In all three subfigures the black curves denote the estimation error for our IRF and the grey curves for LRF with* $\mathbf{X}_b$ *set as the observation in the first step.*

$P_3$ (denoted by $\{\mathbf{X}_i\}$) using the exponential map at the point $\mathbf{I}_3$ (identity matrix). Thus, $\{\mathbf{X}_i\}$ can be viewed as a time sequence of random measurements of the identity matrix. Our recursive filter can then be used as an estimator of this random process. The estimation error at time point $k$ can be computed as the Riemannian distance by (2.2) between the estimate $\hat{\mathbf{X}}_k$ and the ground truth (the identity matrix).

We tested our IRF and evaluated it by comparing its performance with the recursive filter for linear systems on $P_n$ (denoted by LRF) reported in [36]. The parameters of LRF were set to be exactly the same as was presented in [36] except for the initial base point $\mathbf{X}_b$, where all the samples are projected to the tangent space $T_{X_b}P_n$ and then processed with LRF. We set $\mathbf{X}_b$ to be the observation in the first step. In this problem, setting $\mathbf{X}_b$ to be the ground truth would give the best result for LRF, because in this case LRF would reduce to the Kalman filter on the tangent space. Since in the practical case the ground truth is unknown, here we set $\mathbf{X}_b$ as the observation at the first step, which is the best information we know about the data sequence before tracking. We also tried to randomly set $\mathbf{X}_b$, and this did not lead to any observable differences. For the proposed method, the $GL$ actions $\mathbf{g}, \mathbf{h}$ were both set to be the identity, and $\phi^2/\omega^2 = 200$. We performed experiments with three different noise levels, $\sigma^2 = 0.1, 1$, and 2. At each noise level we executed the whole process 20 times and computed the mean error for the corresponding time step.

The results are summarized in Figure 2. From the figure, we can see that LRF performs better when $\sigma^2 = 0.1$, and our method (IRF) performs better when the data is more noisy ($\sigma^2 = 1, 2$). The reason is that LRF uses several Log-Euclidean operations, which is in fact an approximation. For low noise level data, the data points are in a relatively small region around the ground truth (identity), in which case the Log-Euclidean approximation is quite accurate. But for higher noise levels in the data, the region becomes larger and the approximation becomes inaccurate, which leads to large estimation errors. In contrast, our filtering method is fully based on the Riemannian geometry without any Log-Euclidean approximation, so it performs consistently and correctly converges for all three noise levels. In conclusion, although

our recursive filter might converge a little bit slower than LRF, it is more robust to larger amounts of noise, which is common in real tracking situations.

In the synthetic experiments, IRF takes on average 0.86 seconds for each sequence (1000 samples), which is slower than LRF (0.52 seconds per sequence). This is not significant, since, in the video tracking program, the most time consuming part is the likelihood computation, which usually takes 10 to 100 times more than the state update time.

**Table 1**
*Tracking result for the real data experiment.*

| Seq. | Obj. | Start | End | Err(IRF) | Err(LRF) | Err(KM) |
|---|---|---|---|---|---|---|
| C3ps1 | 1 | 200 | 700 | 4.3213 | 10.8467 | 10.7666 |
| C3ps1 | 7 | 200 | 700 | 4.1998 | 7.3524 | 4.7929 |
| C3ps1 | 8 | 200 | 700 | 2.6258 | 5.5844 | 6.2928 |
| C3ps2 | 7 | 500 | 800 | 2.7605 | 10.6478 | 12.4289 |
| C3ps2 | 8 | 500 | 800 | 3.4451 | 7.2113 | 11.6432 |
| Cosow2 | 1 | 500 | 900 | 3.7612 | 4.613 | 6.0196 |
| Cosow2 | 3 | 500 | 900 | 4.9871 | 5.8552 | 7.9788 |

**4.2. Real data experiments.** For the real data experiment, we applied our IRF to more than 3000 frames in different video sequences. Two other covariance descriptor updating methods were also applied to these sequences for comparison, namely, (1) the LRF reported in [36] and (2) the updating method using the Karcher mean (KM) of tracking results in previous frames reported in [26]. The image feature vectors for the target region were computed as reported in [26]. The buffer size $T$ in the KM method were set to 20, which means the KM of covariance descriptors in 20 previous frames were used for the prediction of the covariance descriptor in the current frame. The parameters for LRF were set to values given in [36]. The parameters controlling the state transition and observation noise in our IRF are set to $\omega^2 = 0.0001$ and $\phi^2 = 0.01$. Since our IRF is combined with a particle filter as a position tracker, for the purpose of comparisons, the KM and LRF are also combined with exactly the same particle filter–based position tracker.

First, we used three video sequences from the dataset CAVIAR [4]: 1. ThreePastShop1cor(C3ps1); 2. ThreePastShop2cor(C3ps2); 3. OneShopOneWait2cor(Cosow2). All three sequences are from a fixed camera and a frame size of $384 \times 288$. Seven "objects" were tracked separately. The given ground truth was used to quantitatively evaluate the tracking results. To measure the error for the tracking results, we used the distance between the center of the estimated region and the ground truth. With all three methods having the same initialization, the tracking results are shown in the Table 1, where all the errors shown are the average errors over all the tracked frames. From the table we can see that LRF is more accurate than KM-based methods in most of the results, and our IRF outperforms both these methods. The KM drifts away from the target, because it is based on a sliding window approach. If the number of consecutive noisy frames is close to the window size, the tracker will tend to track the noisy features. For LRF, since it is a nonintrinsic approach, the approximation of the $GL$-invariant metric would introduce errors that accumulate over time across the frames, causing it to drift away. Since IRF is an intrinsic recursive filter, which uses the

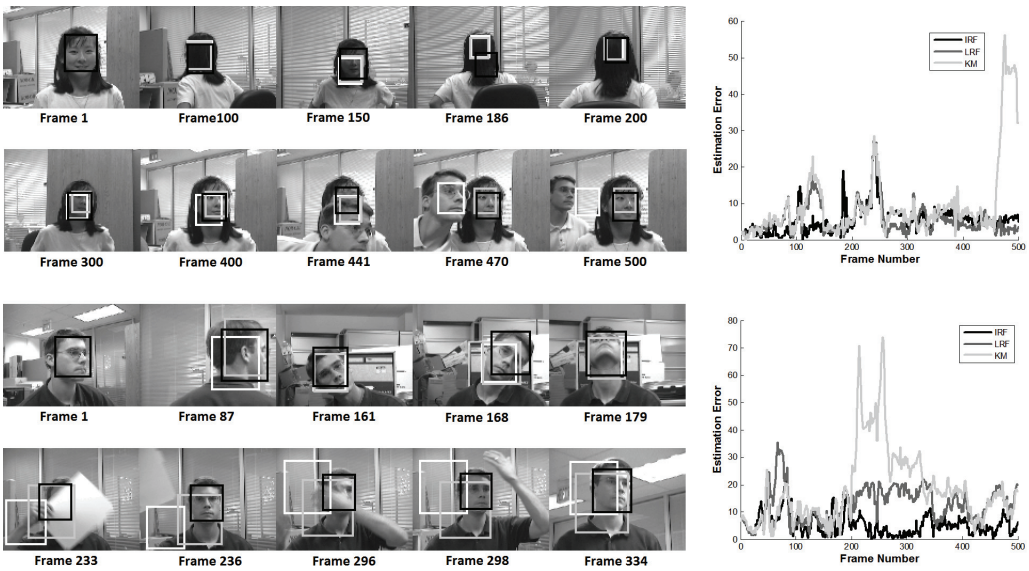


**Figure 3.** *Head tracking result for video sequences with a moving camera. The top and bottom rows depict snapshots and quantitative evaluations of the results from the Seq_mb and Seq_sb, respectively (http://www.ces.clemson.edu/∼stb/research/headtracker/seq/). The tracking error is measured by the distance between the estimated object center and the ground truth. Tracking results from the three methods are shown by using different colored boxes superposed on the images and different colored lines in the plots. Results from our method (IRF) are in black, LRF in dark grey, and KM in white (box) or light grey (error curve).*

$GL$-invariant metric, there is less error introduced in the covariance tracker updates. This in turn leads to higher accuracy in the experiments above.

In the second experiment, we performed head tracking in video sequences with a moving camera. Two video sequences were used: (i) Seq_mb sequence (tracking face) and (ii) Seq_sb. Each of the sequences has 500 frames with frame size $96 \times 128$. Both sequences are challenging because of complex background, fast appearance changes, and occlusions. The results are summarized in Figure 3.

In Seq_mb, KM fails at frame 450 where the occlusion occurs, while LRF and IRF do not lose track. Both KM and LRF produce relatively large errors in capturing the position of the girl's face after the girl turns around the first time between frames 100 to 150 due to the complete change in appearance of the target (girl's face). LRF produces a relatively larger error in estimating the scale (compared to the initialization) of the face between frames 400 to 500, which can be found in the snapshots included in Figure 3. The result of our method (IRF) has a relatively larger error at around frames 100 and 180, because at this time, the camera is tracking the hair of the girl where no feature can be used to locate the position of the face. However, for other frames, IRF tracks the face quite accurately.

In Seq_sb both KM and LRF fail at frame 200, but IRF, however, successfully tracks the whole sequence with relatively high accuracy even with fast appearance changes and occlusions, as shown in the quantitative analysis in Figure 3. These experiments thus demonstrate the accuracy of our method in both moving camera and fixed camera cases.

For the speed of the tracking algorithm, since we extract features only from a window around the target object, the resolution of the video does not really affect the tracking speed while the size of the target object does. Here we compute the average processing time per frame for all sequences. IRF on average takes 0.027 seconds to process each frame, which is faster than KM (0.035 seconds), and LRF is the fastest of all three methods, which takes 0.0067 seconds. One main reason is that LRF is not a particle filter, so it needs much less time in computing the likelihood. The state update takes only 0.0012 seconds in IRF. Also, IRF is still a real time filter (on average more than 30 fps) and yields more accurate results, especially in high noise cases.

**5. Discussion and conclusion.** In this paper, we presented a novel intrinsic recursive filter (IRF) for covariance tracking, which proved to be more robust to noise than existing methods reported in the literature. IRF is based on the intrinsic geometry of the space of covariance matrices and a $GL$-invariant metric that are used in developing the dynamic model and the recursive filter.

We presented a generalization of the normal distribution to $P_n$ and used it to model the system and the observation noise in the IRF. Several properties of this distribution in $P_n$ were also presented in this paper, which to the best of our knowledge have never been addressed in the literature. Note that our generalization of the normal distribution to $P_n$ is rotationally invariant, and the variance of the distribution is controlled by a scalar ($\omega^2$ in (2.8)) rather than a variance control tensor, which is a more general form. One main reason for using this specific form is that the scalar variance control parameter is $GL$ invariant, while the variance control tensor is not, as shown through the following simple calculation. Suppose $\mathbf{V} \in T_{\mathbf{X}}P_n$ is a tangent vector (which is a symmetric matrix) at point $\mathbf{X} \in P_n$, and $\Sigma$ is a variance control tensor. The value of the density function on $\exp_{\mathbf{X}}(\mathbf{V})$ would depend upon the quadratic form $vec(\mathbf{V})^t\Sigma^{-1}vec(\mathbf{V})$, where $vec(\cdot)$ is the vectorization operation on the matrix argument and $\Sigma$ is a second-order tensor. In practice, $\mathbf{X}$ would be the Karcher expectation of the aforementioned distribution and $\mathbf{V}$ would be the tangent vector corresponding to the geodesic from $\mathbf{X}$ to a sample point from the distribution. If we change the coordinates by using a $GL$ operation $\mathbf{g}$, the Karcher expectation becomes $\mathbf{g}\mathbf{X}\mathbf{g}^t$, the vector becomes $\mathbf{g}\mathbf{V}\mathbf{g}^t$, and the quadratic form becomes $vec(\mathbf{g}\mathbf{V}\mathbf{g}^t)^t\Sigma^{-1}vec(\mathbf{g}\mathbf{V}\mathbf{g}^\mathbf{t})$. If we want to keep the value of the density unchanged, we need to change $\Sigma$ according to $\mathbf{g}$, which means that $\Sigma$ is not $GL$ invariant. However, in contrast, it is easy to show that $\omega^2$ in (2.8) is $GL$ invariant.

Further, the IRF is quite different from the Kalman filter, which is known to be an optimal linear filter (in a vector space) based on an additive Gaussian noise assumption. One reason for the Kalman filter to be optimal is that it actually tracks the distribution of the object state (posterior) based on a Bayesian tracking framework. If a filter does not track the whole distribution, usually it would explicitly or implicitly approximate the posterior based on the state variables it has tracked. However, the approximation error might accumulate in the system. From a geometric point of view, the Kalman filter is highly dependent on geometric properties of the Euclidean space. This is because the Kalman filter is based on the fact that the convolution of two Gaussians is a Gaussian. And this property of the Gaussian stems from the fact that the Gaussian is the limit distribution in the central limit theorem. One key problem in extending the Kalman filter intrinsically to $P_n$ is finding two densities $p_A(\mathbf{X};\theta_A)$,

$p_B(\mathbf{X}|\mathbf{Y})$ with the following properties:

$$p_A(\mathbf{X};\theta'_A) = \int_{P_n} p_B(\mathbf{X}|\mathbf{Y}) p_A(\mathbf{Y};\theta)[d\mathbf{Y}], \tag{5.1}$$

where $\theta_A$ is the parameter of density $p_A$. $p_A$ here is usually the posterior and $p_B$ is the state transition noise distribution. The equation above means that after the state transition the form of the posterior remains the same. Without this property, even if the whole distribution is tracked, the filter is implicitly approximating the true distribution after the state transition by using the same form as the posterior from the last step, which still would lead to errors being accumulated in the system. However, it is nontrivial to find such distributions on $P_n$. In [32, 12], a central limit theorem was presented in $P_n$ for rotationally invariant probability measures based on the Helgason–Fourier transform [13]. However, currently the probability measure in the limit does not have a closed form in the space domain. Thus, intrinsically extending the Kalman filter to $P_n$ is still an open problem. IRF instead tracks only the mode of the distribution. It is not an optimal filter, but is *intrinsic and mathematically consistent* with respect to the noise model used, unlike the LRF in [36]. We also presented a real-time covariance tracking algorithm based on this filter which is combined with an existing particle position tracker from the literature [2]. Finally, experiments on synthetic and real data favourably demonstrated the accuracy of our method over rival methods.

## Appendix A. Proofs of Theorem 2.1 and Corollary 2.2.

Theorem 2.1. The normalization factor $Z$ in (2.8) is a finite constant with respect to parameter $\mathbf{M} \in P_n$.

To prove this theorem, we need to first prove the following lemma.

Lemma A.1.

$$W = \int_{P_n} \exp\Big( - \frac{\mathrm{Tr}(\log \mathbf{X} \log \mathbf{X}^{\mathbf{t}}\Big)}{2\omega^2})[d\mathbf{X}] < \infty.$$

*Proof.* This lemma indicates that the normalization factor $Z$ is constant, and hence $p(\mathbf{X};\mathbf{M},\omega^2)$ is a probability density function on $P_n$. To prove this lemma, we first represent $\mathbf{X}$ in polar coordinates $\{\lambda_i\}, \mathbf{R}$ based on the eigendecomposition, $\mathbf{X} = \mathbf{R}\mathbf{\Lambda}\mathbf{R}^t$, where $\mathbf{\Lambda} = diag(\lambda_1, \ldots, \lambda_n), \mathbf{R}\mathbf{R}^t = I_{n\times n}$. From [32] we know that

$$[dX] = c_n \prod_{j=1}^{n} \lambda_j^{-(n-1)/2} \prod_{1\le i<j\le n} |\lambda_i - \lambda_j| \prod_{i=1}^{n} \frac{d\lambda_i}{\lambda_i} d\mathbf{R}, \tag{A.1}$$

where $d\mathbf{R}$ is the invariant measure on the orthogonal group $O(n)$ [6] with $\int_{O(n)} d\mathbf{R} = 1$ (since the orthogonal group is compact, we can easily normalize the measure), $c_n$ is a constant depending on $n$, and $d\lambda_i$ is the Lebesgue measure in $R$. With the following change of variables,

$y_i = \log(\lambda_i)$, we get

$$W = c_n \int_{R^n} \exp\left(-\sum_{i=1}^{n}\left(\frac{1}{2\omega^2}y_i^2 + \frac{n-1}{2}y_i\right)\right) \prod_{1\le i<j\le n} |\exp(y_i) - \exp(y_j)| d\mathbf{y} \tag{A.2}$$

$$= c_n \int_{R^n} \left| \sum_{\gamma\in S_n} sgn(\gamma) \exp\left(-\frac{1}{2}\sum_{i=1}^{n}(y_i^2/\omega^2 + (n-1-2\gamma(i))y_i)\right)\right| d\mathbf{y} \tag{A.3}$$

$$\le c_n (2\pi\omega^2)^{\frac{n}{2}} \sum_{\gamma\in S_n} \exp\left(\frac{\omega^2\sum_{i=1}^{n}\gamma(i)^2 - \omega^2 n(n-1)^2/4}{2}\right) < \infty, \tag{A.4}$$

where $\gamma$ is an element of $S_n$ which is the set of all permutations of $0, 1, \ldots, n-1$, and $sgn(\gamma)$ denotes the signature of $\gamma$ which is 1 or $-1$, depending on the permutation. The derivation from (A.2) to (A.3) is based on the fact that $\prod_{1\le i<j\le n}(\exp(y_i) - \exp(y_j))$ is actually a Vandermonde determinant. By expansion, using the Leibniz formula and putting in the Gaussian term, we can directly get (A.3). The inequality in (A.4) is based on the convexity of the absolute-value function. ■

We are now ready to present the proof of Theorem 2.1.

*Proof.* Assume $Z$ is a function of $\mathbf{M}$ $\forall \mathbf{M} \in P_n$ denoted by $Z(\mathbf{M})$:

$$Z(\mathbf{M}) = \int_{P_n} \exp\left(-\frac{dist^2(\mathbf{X}, \mathbf{M})}{2\omega^2}\right)[d\mathbf{X}]. \tag{A.5}$$

Since the $GL$ group action is transitive on $P_n$, $\forall \mathbf{N} \in P_n$, $\exists \mathbf{g} \in GL(n)$ such that $\mathbf{N} = \mathbf{g}\mathbf{M}\mathbf{g}^t$. Thus,

$$Z(\mathbf{N}) = \int_{P_n} \exp\left(-\frac{dist^2(\mathbf{X}, g\mathbf{M}g^t)}{2\omega^2}\right)[d\mathbf{X}] = \int_{P_n} \exp\left(-\frac{dist^2(g^{-1}\mathbf{X}g^{-t}, \mathbf{M})}{2\omega^2}\right)[d\mathbf{X}].$$

Let $\mathbf{Y} = g^{-1}\mathbf{X}g^{-t}$ so that $\mathbf{X} = g\mathbf{Y}g^t$. Substituting this into the above equation we get

$$Z(\mathbf{N}) = \int_{P_n} \exp\left(-\frac{dist^2(\mathbf{Y}, \mathbf{M})}{2\omega^2}\right)[d\mathbf{g}\mathbf{Y}\mathbf{g}^\mathbf{t}] = \int_{P_n} \exp\left(-\frac{dist^2(\mathbf{Y}, \mathbf{M})}{2\omega^2}\right)[d\mathbf{Y}] = Z(\mathbf{M}).$$

Thus, $\forall \mathbf{M}, \mathbf{N} \in P_n$, $Z(\mathbf{M}) = Z(\mathbf{N})$. From (A.1) we know that $Z(\mathbf{I}) < \infty$; by substitution as in the above, we obtain the result that $Z$ is finite and constant with respect to $\mathbf{M}$. ■

What follows is the proof of Corollary 2.2.

Corollary 2.2. Given a set of i.i.d samples $\{\mathbf{X_i}\}$ drawn from the distribution $dP(\mathbf{X}; \mathbf{M}, \omega^2)$, the MLE of the parameter $\mathbf{M}$ is the KM of all samples.

*Proof.*

$$-\log(p(\mathbf{X_1}, \mathbf{X_2}, \ldots, \mathbf{X_m}; \mathbf{M}, \omega^2)) = -\sum_i \log(p(\mathbf{X_i}; \mathbf{M}, \omega^2))$$

$$= n\log Z + \frac{\sum_i dist^2(\mathbf{X_i}, \mathbf{M})}{2\omega^2}.$$

Since $Z$ is constant with respect to $\mathbf{M}$, as proved the Theorem 2.1, we have

$$\underset{\mathbf{M}}{\operatorname{argmax}}\, p(\mathbf{X_1}, \mathbf{X_2}, \ldots, \mathbf{X_m}; \mathbf{M}, \omega^2) = \underset{\mathbf{M}}{\operatorname{argmin}} \sum_i dist^2(\mathbf{X}_i, \mathbf{M})$$

Thus, the MLE of the parameter $\mathbf{M}$ of the distribution $dP(\mathbf{X}; \mathbf{M})$ equals the KM of samples. ∎

## Appendix B. Proof of Theorem 2.3.

Theorem 2.3. Parameter $\mathbf{M}$ is the Karcher expectation of the generalized normal distribution $dP(\mathbf{X}; \mathbf{M}, \omega^2)$.

*Proof.* To prove this, we need to show that $dP(\mathbf{X}; \mathbf{M}, \omega^2)$ satisfies (2.7). Let

$$\Psi = \int_{P_n} \log(\mathbf{M}^{-1/2}\mathbf{X}\mathbf{M}^{-1/2}) dP(\mathbf{X}; \mathbf{M}, \omega^2);$$

then in the integral, using a change of variables, $\mathbf{X}$ to $\mathbf{Y} = \mathbf{M}\mathbf{X}^{-1}\mathbf{M}$ ($\mathbf{X} = \mathbf{M}\mathbf{Y}^{-1}\mathbf{M}$). Since $P_n$ is a symmetric space and the metric/measure is $GL$ invariant, we know that $[d\mathbf{X}] = [d\mathbf{Y}]$, and $dist(\mathbf{X}, \mathbf{M}) = dist(\mathbf{Y}, \mathbf{M})$. Thus we have

$$\begin{aligned}\Psi &= \int_{P_n} \log(\mathbf{M}^{-1/2}\mathbf{X}\mathbf{M}^{-1/2}) \frac{1}{Z} \exp\left( - \frac{dist^2(\mathbf{X}, \mathbf{M})}{2\omega^2} \right) [d\mathbf{X}] \\ &= \int_{P_n} \log(\mathbf{M}^{1/2}\mathbf{Y}^{-1}\mathbf{M}^{1/2}) \frac{1}{Z} \exp\left( - \frac{dist^2(\mathbf{Y}, \mathbf{M})}{2\omega^2} \right) [d\mathbf{Y}] = -\Psi = 0.\end{aligned}$$

Since $P_n$ has nonpositive curvature, the solution of (2.5) is unique [14]. Thus $\mathbf{M}$ is the Karcher expectation of $dP(\mathbf{X}; \mathbf{M}, \omega^2)$. ∎

## Appendix C. Proof of Theorem 2.4.

Theorem 2.4.

$$\lim_{\omega \to 0} \frac{Var(\mathbf{X})}{\omega^2} = \frac{n(n+1)}{2}.$$

*Proof.* Let

$$v(\mathbf{y}, \omega) = \sum_{\gamma \in S_n} sgn(\gamma) \exp\left( - \frac{1}{2} \sum_{i=1}^{n} (y_i^2 + \omega(n - 1 - 2\gamma(i)) y_i) \right), \tag{C.1}$$

where $\gamma$, $S_n$, and $sgn(\gamma)$ are related to the permutation of $0, 1, \ldots, n-1$, which is defined to be the same as in (A.3). Also we can find that $q(\mathbf{y}) = \frac{|v(\mathbf{y},\omega)|}{z(\omega)}$ and $z(\omega) = \int_{R^n} |v(\mathbf{y}, \omega)| d\mathbf{y}$. The Taylor expansion of $v(\mathbf{y}, \omega)$ up to $\frac{n(n-1)}{2}$th order with respect to $\omega$ around zero is

(C.2)

$$\begin{aligned}v(\mathbf{y}, \omega) = \sum_{\gamma \in S_n} sgn(\gamma) \sum_{k=0}^{\frac{n(n-1)}{2}} \frac{(-\omega)^k}{k!} \exp\left( - \frac{\sum_{i=1}^n y_i^2}{2} \right) \left( \sum_{i=1}^{n} \left( \frac{n-1}{2} - \gamma(i) \right) y_i \right)^k + O(\omega^{\frac{n^2-n+2}{2}}) \\ = C(-\omega)^{\frac{n(n-1)}{2}} \exp\left( - \frac{\sum_{i=1}^n y_i^2}{2} \right) \prod_{1 \le i < j \le n} (y_i - y_j) + O(\omega^{\frac{n^2-n+2}{2}}),\end{aligned}$$

where $C$ is a constant. Equation (C.2) used the fact that given $n$ nonnegative integers $\kappa_i$ and $\sum_{i=1}^{n} \kappa_i \leq \frac{n(n-1)}{2}$,

$$\sum_{\gamma} sgn(\gamma) \prod_{i=1}^{n} \gamma(i)^{\kappa_i} = \ 0 \quad \text{if } \{\kappa_i\} \notin S_n. \tag{C.3}$$

So, in the Taylor expansion all the terms with degree less than $\frac{n(n-1)}{2}$ are zeros. In the $\frac{n(n-1)}{2}$th order terms, only terms with powers in $S_n$ will be nonzero.

Let the density $\hat{q}(\mathbf{y}) = \frac{1}{\hat{z}} \exp(-\frac{\sum_{i=1}^{n} y_i^2}{2}) \prod_{1 \leq i < j \leq n} |y_i - y_j|$, which is exactly the joint distribution of the eigenvalues of a Gaussian orthogonal ensemble [20], which is a symmetric random matrix with each of its elements being independent random variables drawn from a zero mean Gaussian. In this case, the variance of the diagonal elements in the random matrix is 1 and that of the off diagonal elements is $\frac{1}{2}$. Recall that we are now in polar coordinates. By transforming $\hat{q}$ to the Cartesian coordinates of the space of symmetric matrices we get

$$\begin{aligned} Var_{\hat{q}}(\mathbf{y}) &= \frac{1}{\hat{z}} \int_{R^n} \mathbf{y}^t \mathbf{y} \hat{q}(\mathbf{y}) d\mathbf{y} \\ &= (2\pi)^{-\frac{n(n-1)}{4}} \int_{\mathbf{V} \in Sym(n)} \mathrm{tr}(\mathbf{V}^2) \exp\left( -\frac{\mathrm{tr}(\mathbf{V}^2)}{2} \right) d\mathbf{V} = \frac{n(n+1)}{2}, \end{aligned} \tag{C.4}$$

where $Sym(n)$ is the space of $n \times n$ symmetric matrices, and $d\mathbf{V}$ is the Lebesgue measure in $Sym(n)$.

From above we know that

$$\lim_{\omega \to 0} \frac{Var(\mathbf{X})}{\omega^2} = \lim_{\omega \to 0} Var_q(\mathbf{y}) = Var_{\hat{q}}(\mathbf{y}) = \frac{n(n+1)}{2}. \tag{C.5}$$

■

## Appendix D. Proof of Theorem 2.5.

Theorem 2.5.

$$\lim_{\omega \to \infty} \frac{Var(\mathbf{X})}{\omega^4} = \frac{(n^3 - n)}{12}.$$

*Proof.* We first define the upper bound and lower bound on $q(\mathbf{y})$:

$$\begin{aligned} q_u(\mathbf{y}) &= \frac{1}{z_u(\omega)} \sum_{\gamma} \exp\left( -\frac{1}{2} \sum_{i=1}^{n} (y_i^2 + \omega(n - 1 - 2\gamma(i)) y_i) \right), \\ q_\iota(\mathbf{y}) &= \frac{1}{z_\iota(\omega)} \exp\left( -\frac{1}{2} \sum_{i} y_i^2 \right) \prod_{1 \leq i < j \leq n} 2\left( \cosh\left( \frac{\omega(y_i - y_j)}{2} \right) - 1 \right) \end{aligned} \tag{D.1}$$

with $z_u$ and $z_\iota$ being the normalization factors, respectively. Note that both $q_u$ and $q_\iota$ are Gaussian mixtures. In $q_u$ all mixing weights are positive, while in $q_\iota$ there are negative weights. After expansion we have

$$\begin{aligned} q_\iota(\mathbf{y}) &= \frac{1}{z_\iota(\omega)} \sum_{\beta \in B_n} w_\beta \exp\left( -\frac{1}{2} \sum_{i=1}^{n} (y_i + \omega(n - 1 - 2\beta(i))/2)^2 \right), \\ w_\beta &= \alpha_\beta \exp\left( \frac{\omega^2 \sum_{i=1}^{n} \beta(i)^2 - \omega^2 (n(n-1)^2/4)}{2} \right), \end{aligned} \tag{D.2}$$

where $B_n$ is the set of all possible power combinations of polynomial

$$\sum_{\beta\in B_n}\alpha_\beta\prod_{i=1}^{n}x_i^{\beta(i)}=\prod_{1\le i<j\le n}(x_i+x_j-2\sqrt{x_ix_j}) \tag{D.3}$$

and $\alpha_\beta$ are the coefficients. We can prove that

$$\max_{\beta\in B_n}\sum_{i=1}^{n}\beta(i)^2=\sum_{i=1}^{n}\gamma(i)^2=\frac{(2n-1)(n^2-n)}{6}. \tag{D.4}$$

The maximum can be achieved only at $\beta\in S_n$, and $\alpha_\beta=1\,\forall\beta\in S_n$.

From the definition we can compute the normalization constants and the variances of $q_\iota$ and $q_u$ in a closed form:

$$\begin{aligned}
z_u&=(2\pi)^{n/2}\sum_{\gamma}\exp\left(\frac{\omega^2\sum_{i=1}^{n}\gamma(i)^2-\omega^2n(n-1)^2/4}{2}\right)\\
&=(2\pi)^{n/2}n!\exp\left(\frac{\omega^2n(n^2-1)}{24}\right),\\
z_\iota&=(2\pi)^{n/2}\sum_{\beta\in B_n}\alpha_\beta\exp\left(\frac{\omega^2\sum_{i=1}^{n}\beta(i)^2-\omega^2n(n-1)^2/4}{2}\right)\\
&=z_u+(2\pi)^{n/2}\sum_{\beta\in B_n\setminus S_n}\alpha_\beta\exp\left(\frac{\omega^2\sum_{i=1}^{n}\beta(i)^2-\omega^2n(n-1)^2/4}{2}\right),\\
Var_{q_u}(\mathbf{y})&=n+\frac{n^3-n}{12}\omega^2,\\
Var_{q_\iota}(\mathbf{y})&=n-\frac{n(n-1)^2}{4}\omega^2\\
&+\frac{(2\pi)^{n/2}}{z_\iota}\omega^2\sum_{\beta\in B_n}\exp\left(\frac{\omega^2\sum_{i=1}^{n}\beta(i)^2-\omega^2n(n-1)^2/4}{2}\right)\left(\sum_{i=1}^{n}\beta(i)^2\right).
\end{aligned} \tag{D.5}$$

Since $0\le\cosh(x)-1\le|\sinh(x)|$ and $|\sum_i x_i|\le\sum_i|x_i|$, we have $\forall\mathbf{y}\in R^n$, $z_\iota q_\iota(\mathbf{y})\le zq(\mathbf{y})<z_uq_u(\mathbf{y})$ and also $z_\iota\le z<z_u$. We can then get the following bounds for $Var_q(\mathbf{y})$:

$$\frac{z_\iota}{z_u}Var_{q_\iota}(\mathbf{y})\le Var_q(\mathbf{y})\le\frac{z_u}{z_\iota}Var_{q_u}(\mathbf{y}). \tag{D.6}$$

From (D.5) we can show that

$$\lim_{\omega\to\infty}\frac{z_\iota}{z_u}=\lim_{\omega\to\infty}\left(1+\sum_{\beta\in B_n\setminus S_n}\alpha_\beta\exp\left(\frac{\omega^2}{2}\left(\sum_{i=1}^{n}\beta(i)^2-\frac{(2n-1)(n^2-n)}{6}\right)\right)\right)=1, \tag{D.7}$$

because $\forall \beta \in B_n \setminus S_n$, $\sum_i \beta(i)^2 < \frac{(2n-1)(n^2-n)}{6}$. Similarly,

$$\begin{aligned} &\lim_{\omega\to\infty} \frac{Var_{q_l}(\mathbf{y})}{\omega^2} = -\frac{n(n-1)^2}{4} \\ &+ \lim_{\omega\to\infty} \sum_{\beta\in B_n} \alpha_\beta \exp\left(\frac{\omega^2}{2}\left(\sum_{i=1}^{n}\beta(i)^2 - \frac{(2n-1)(n^2-n)}{6}\right)\right)\left(\sum_{i=1}^{n}\beta(i)^2\right) \\ &= \frac{n^3-n}{12} = \lim_{\omega\to\infty}\frac{Var_{q_u}(\mathbf{y})}{\omega^2} = \lim_{\omega\to\infty}\frac{Var_q(\mathbf{y})}{\omega^2} = \lim_{\omega\to\infty}\frac{Var(\mathbf{X})}{\omega^4}. \end{aligned} \tag{D.8}$$

■

## Appendix E. Proof of Theorem 3.1.

Theorem 3.1. The energy function $E_k$ in (3.4) is geodesically convex on the Riemannian manifold $P_n \times P_n$, where $\times$ denotes the Cartesian product.

*Proof.* Recall that $E_k$ in (3.4) is a sum of squared distance function with positive weights. Here the strategy is to prove that each item inside the sum is geodesically convex.

It is known that $\forall \mathbf{C} \in P_n$, function $d_{\mathbf{C}}(\mathbf{X}) = dist(\mathbf{X}, \mathbf{C})$ is geodesically convex [22]. Also, it is known that if $\gamma, \beta : [0,1] \mapsto P_n$ are two geodesics on $P_n$, then $\alpha(t) = (\gamma(t), \beta(t))$ is a geodesic on $P_n \times P_n$ [40]. Combining these two facts with the nonnegativity of the distance function, we can see that in $P_n \times P_n$, the function $f_{\mathbf{C}}((\mathbf{X}_1, \mathbf{X}_2)) = dist(\mathbf{X}_1, \mathbf{C})^2$ is geodesically convex.

Now we want to prove that the function $g((\mathbf{X}_1, \mathbf{X}_2)) = dist(\mathbf{X}_1, \mathbf{X}_2)^2$ is geodesically convex. As shown in [3], for any two geodesics $\gamma_1, \gamma_2 : [0,1] \mapsto P_n$,

$$dist\left(\gamma_1\left(\frac{1}{2}\right), \gamma_2\left(\frac{1}{2}\right)\right) \leq \frac{1}{2}dist(\gamma_1(0), \gamma_2(0)) + \frac{1}{2}dist(\gamma_1(1), \gamma_2(1)). \tag{E.1}$$

With the definition of the geodesic convexity, we can see that $\sqrt{(}g)$ and $g$ are both geodesically convex.

Since the distance function is $GL$ invariant, $E_k$ can be written as a weighted sum of $f$ and $g$ with positive weights, and then we know that $E_k$ is also geodesically convex. ■

## REFERENCES


[1] V. Arsigny, P. Fillard, X. Pennec, and N. Ayache, *Fast and simple calculus on tensors in the log-Euclidean framework*, in Proceedings of the 8th International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI), Vol. 3749, 2005, pp. 115–122.

[2] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, *A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking*, IEEE Trans. Signal Process., 50 (2002), pp. 174–188.

[3] W. Ballmann, *Lectures on Spaces of Nonpositive Curvature*, DMV Sem. 25 of Oberwolfach Seminars,, Birkhäuser, Basel, 1995.

[4] *CAVIAR: Context Aware Vision Using Image-Based Active Recognition*, http://homepages.inf.ed.ac.uk/rbf/CAVIAR/.

[5] A. Cherian, S. Sra, A. Banerjee, and N. Papanikolopoulos, *Efficient similarity search for covariance matrices via the Jensen-Bregman logdet divergence*, in Proceedings of the 13th International Conference on Computer Vision (ICCV), 2011.

[6] Y. Chikuse, *Statistics On Special Manifolds*, Springer, New York, 2002.

[7] Y. Chikuse, *State space models on special manifolds*, J. Multivariate Anal., 97 (2006), pp. 1284–1294.

[8] D. Comaniciu and P. Meer, *Mean shift: A robust approach toward feature space analysis*, IEEE Trans. Pattern Anal. Mach. Intell., 24 (2002), pp. 603–619.
[9] D. Comaniciu, V. Ramesh, and P. Meer, *Real-time tracking of non-rigid objects using mean shift*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2000.
[10] P. Fletcher and S. Joshi, *Principal geodesic analysis on symmetric spaces: Statistics of diffusion tensors*, in Computer Vision and Mathematical Methods in Medical and Biomedical Image Analysis, Lecture Notes in Comput. Sci. 3117, M. Sonka, I. Kakadiaris, and J. Kybic, eds.,Springer-Verlag, Berlin, 2004, pp. 87–98.
[11] P.T. Fletcher, C. Lu, S.M. Pizer, and S. Joshi, *Principal geodesic analysis for the study of nonlinear statistics of shape*, IEEE Trans. Med. Imag., 23 (2004), pp. 995–1005.
[12] P. Graczyk, *A central limit theorem on the space of positive definite symmetric matrices*, Ann. Inst. Fourier (Grenoble), 42 (1992), pp. 857–874.
[13] S. Helgason, *Differential Geometry, Lie Groups, and Symmetric Spaces*, Academic Press, New York, 2001.
[14] H. Karcher, *Riemannian center of mass and mollifier smoothing*, Comm. Pure Appl. Math, 30 (1977), pp. 509–541.
[15] J. Kwon and F. C. Park, *Visual tracking via particle filtering on the affine group*, Int. J. Robot. Res., 29 (2010), pp. 198–217.
[16] C. Lenglet, M. Rousson, R. Deriche, and O. Faugeras, *Statistics on the manifold of multivariate normal distributions: Theory and application to diffusion tensor MRI processing*, J. Math. Imaging Vision, 25 (2006), pp. 423–444.
[17] M. Li, W. Chen, K. Huang, and T. Tan, *Visual tracking via incremental self-tuning particle filtering on the affine group*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010.
[18] X. Li, W. Hu, Z. Zhang, X. Zhang, M. Zhu, and J. Cheng, *Visual tracking via incremental log-Euclidean Riemannian subspace learning*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2008, pp. 1–8.
[19] Y. Liu, G. Li, and Z. Shi, *Covariance tracking via geometric particle filtering*, EURASIP J. Adv. Signal Process., 2010 (2010), pp. 22:1–22:9.
[20] M. L. Metha, *Random Matrices and the Statistics Theory of Energy Levels*, Academic Press, New York, London, 1967.
[21] M. Moakher, *A differential geometric approach to the geometric mean of symmetric positive-definite matrices*, SIAM J. Matrix Anal. Appl., 26 (2005), pp. 735–747.
[22] G. D. Mostow, *Strong Rigidity of Locally Symmetric Spaces*, Ann. of Math. Stud. 78, Princeton University Press, Princeton, NJ, 1973.
[23] X. Pennec, *Intrinsic statistics on Riemannian manifolds basic tools for geometric measurements*, J. Math. Imaging Vision, 25 (2006), pp. 127–154.
[24] F. Porikli, *Learning on manifolds*, in Proceedings of the 2010 Joint IAPR International Conference on Structural, Syntactic, and Statistical Pattern Recognition, Springer-Verlag, Berlin, Heidelberg, 2010, pp. 20–39.
[25] F. Porikli and P. Pan, *Regressed importance sampling on manifolds for efficient object tracking*, in Proceedings of the IEEE International Conference on Advanced Video and Signal Based Surveillance, 2009.
[26] F. Porikli, O. Tuzel, and P. Meer, *Covariance tracking using model update based on lie algebra*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Vol. 1, 2006, pp. 728–735.
[27] A. Schwartzman, *Random Ellipsoids and False Discovery Rates: Statistics for Diffusion Tensor Imaging Data*, Ph.D. thesis, Stanford University, Stanford, CA, 2006.
[28] H. Snoussi and C. Richard, *Monte Carlo tracking on the Riemannian manifold of multivariate normal distributions*, in Proceedings of the Digital Signal Processing Workshop and 5th IEEE Signal Processing Education Workshop (DSP/SPE), 2009.
[29] H. W. Sorenson, ed., *Kalman Filtering: Theory and Application*, IEEE Press, Los Alamitos, CA, 1985.
[30] A. Srivastava and E. Klassen, *Bayesian and geometric subspace tracking*, Adv. in Appl. Probab., 36 (2004), pp. 43–56.

[31] R. Subbarao and P. Meer, *Nonlinear mean shift over Riemannian manifolds*, Int. J. Comput. Vis., 84 (2009), pp. 1–20.
[32] A. Terras, *Harmonic Analysis on Symmetric Spaces and Applications*, Springer-Verlag, Berlin, 1988.
[33] D. Tosato, M. Farenzena, M. Cistani, and V. Murino, *A re-evaluation of pedestrian detection on Riemannian manifolds*, in Proceedings of the 20th International Conference on Pattern Recognition (ICPR), 2010, pp. 3308–3311.
[34] O. Tuzel, F. Porikli, and P. Meer, *Region covariance: A fast descriptor for detection and classification*, in Proceedings of the 9th European Conference on Computer Vision (ECCV), 2006.
[35] O. Tuzel, F. Porikli, and P. Meer, *Pedestrian detection via classification on Riemannian manifolds*, IEEE Trans. Pattern Anal. Mach. Intell., 30 (2008), pp. 1713–1727.
[36] A. Tyagi and J. W. Davis, *A recursive filter for linear systems on Riemannian manifolds*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2008, pp. 1–8.
[37] C. Udriste, *Convex Functions and Optimization Methods on Riemannian Manifolds*, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1994.
[38] Y. Wong, *Sectional curvatures of Grassmann manifolds*, Proc. Nat. Acad. Sci. U.S.A., 60 (1968), pp. 75–79.
[39] Y. Wu, J. Cheng, J. Wang, and H. Lu, *Real-time visual tracking via incremental covariance tensor learning*, in Proceedings of the 12th International Conference on Computer Vision (ICCV), 2009.
[40] Y. Xie, B. Vemuri, and J. Ho, *Statistical analysis of tensor fields*, in Proceedings of the 13th International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI), 2010.