A Recursive Filter For Linear Systems on Riemannian Manifolds

Ambrish Tyagi James W. Davis
Dept. of Computer Science and Engineering
The Ohio State University
Columbus OH 43210
{tyagia,jwdavis}@cse.ohio-state.edu

Abstract

We present an online, recursive filtering technique to model linear dynamical systems that operate on the state space of symmetric positive definite matrices (tensors) that lie on a Riemannian manifold. The proposed approach describes a predict-and-update computational paradigm, similar to a vector Kalman filter, to estimate the optimal tensor state. We adapt the original Kalman filtering algorithm to appropriately propagate the state over time and assimilate observations, while conforming to the geometry of the manifold. We validate our algorithm with synthetic data experiments and demonstrate its application to visual object tracking using covariance features.

1. Introduction

Symmetric positive definite matrices (or tensors), in the form of covariance matrices, are becoming a popular feature representation in various computer vision applications. These feature spaces can be used to encode the principal diffusion directions in Diffusion Tensor Imaging (DTI) [3, 7] or model the appearance of objects for the task of visual tracking [8, 9]. Outside the domain of computer vision, symmetric positive definite tensors play an important role in various branches of continuum physics (*e.g.*, stress-strain tensor) [6].

The space of symmetric positive definite (SPD) matrices is not a vector space. Instead, it lies on a Riemannian manifold that constitutes a convex half-cone in the vector space of matrices. The lack of usual vector operations for the SPD matrices (*e.g.*, subtraction, mean) pose a challenge for them to be used in many practical algorithms. Likewise, it is also desirable in many applications to estimate the true value of the SPD tensors from noisy measurements or to model linear dynamical systems that operate on tensor state spaces. The popular vector space algorithm, Kalman filter (KF), is an efficient recursive technique to estimate the vector state of a system from noisy measurements. The typical approach of applying the standard vector [5] or matrix [2] Kalman filter to a SPD state will fail due to the geometric structure of the underlying space. Applying regular vector space operations will result in invalid state configurations that do not lie on the SPD manifold (*e.g.*, matrix difference of two SPD tensors may not be SPD).

In this paper we present an optimal filtering strategy, analogous to the vector space KF, which is applicable to linear dynamical systems whose state is represented by SPD tensors. The proposed algorithm provides an optimal estimate of the system state in the least mean squared sense. At each time instance, the state distribution is modeled as a multivariate Gaussian on the Riemannian manifold of SPD matrices. The algorithm predicts the new state of the system according to the given model and appropriately propagates the necessary information in time, conforming to the differential-geometric properties of the manifold. Subsequently, the filter assimilates a new observation into the system and updates the state estimate. This predict-and-update cycle is repeated to propagate the system state over time and assimilate observations during the estimation process.

The proposed filtering algorithm can be applied to a wide range of problems that deal with estimation of SPD state states like measurement of physical tensors, online model update of covariance appearance features, *etc*. To demonstrate the efficacy of our framework, we first apply it to the problem of estimating the value of a constant tensor from noisy measurements on a synthetically generated dataset. Next, we describe an approach for object tracking using covariance matrix of appearance features. We employ an augmented SPD state space that simultaneously encodes the position, velocity, and appearance of the object. The state is propagated in time using a linear system that smoothes the location and updates the appearance model in a unified manner. Comparative tracking results with the baseline covariance tracking algorithm with mean update strategy [8] are presented.

The remainder of the paper is organized as follows. We

978-1-4244-2243-2/08/$25.00 ©2008 IEEE

present the related work in Sect. 2. The proposed algorithm is derived in Sect. 3 followed by the description of the object tracking model in Sect. 4. Section 5 presents an experimental evaluation of the proposed framework. Finally, we summarize and give concluding remarks in Sect. 6.

2. Related Work

Kalman filtering, developed by Rudolf Kalman in 1960 [5], is an efficient recursive technique to estimate the state of a dynamic system from a series of incomplete and noisy measurements. In computer vision, KFs have been widely used for object tracking. They assume that the underlying process driving an object is linear and Gaussian distributed. Typically, KF operates on vector spaces and assume that the state is represented as either a vector or a matrix [2]. Extended Kalman filters (EKFs) have also been used, which allow the underlying process to be nonlinear.

Also, recent research in computer vision has focused on algorithms that exploit the underlying structure (*e.g.*, manifolds) of the state spaces used for object detection, tracking, learning, *etc*. The SPD covariance matrices are popular in the area of medical imaging [3, 7] and object tracking [8, 9]. Several vector space algorithms such as averaging, interpolation, principal component analysis, *etc*., have been successfully extended to work on the SPD manifolds [3, 6, 7].

3. Theory

We begin with a brief overview of Riemannian geometry on the space of SPD matrices. Interested readers are referred to [1, 7] for more information. The details of modeling a linear dynamical system with a SPD state space followed by the derivation of the optimal filtering strategy including the state and error propagation mechanism on the Riemannian manifold are presented. Our proof is inspired by the optimal derivation of the vector KF algorithm [5]. Henceforth, we will denote the elements of the vector space by small boldface letters (*e.g.*, $\mathbf{v}$) and the elements of the SPD manifold by capital boldface letters (*e.g.*, $\mathbf{X}$).

3.1. Riemannian Manifolds of SPD Matrices

Let $\mathbf{X} \in Sym_n^+$ denote the $2-$mode $n \times n$ SPD tensor. Elements of Sym_n^+ lie on a connected Riemannian manifold, $\mathcal{M} \in \mathbb{R}^{n \times n}$. The derivative at a point $\mathbf{X}$ on this manifold lies in the tangent space, $\mathcal{T}_{\mathbf{X}}\mathcal{M}$, which is a vector space formed by symmetric matrices, Sym_n. The Sym_n^+ manifold is endowed with an invariant Riemannian metric $<\mathbf{y}, \mathbf{z}>_{\mathbf{X}} = \text{tr}\left(\mathbf{X}^{-1/2}\mathbf{y}\mathbf{X}^{-1}\mathbf{z}\mathbf{X}^{-1/2}\right)$, $\forall \mathbf{y}, \mathbf{z} \in \mathcal{T}_{\mathbf{X}}\mathcal{M}$, that varies smoothly from point to point. The metric induces a norm for the tangent vectors on the tangent space, such that, $||\mathbf{y}||_{\mathbf{X}} = <\mathbf{y}, \mathbf{y}>_{\mathbf{X}}$.

The minimum length curve joining two points on $\mathcal{M}$ is called the *geodesic*. For any $\mathbf{X} \in \mathcal{M}$ there is a unique geodesic starting with the tangent vector $\mathbf{v} \in \mathcal{T}_{\mathbf{X}}\mathcal{M}$. The exponential and logarithmic maps associated with the Riemannian metric are respectively given by,

$$\exp_{\mathbf{X}}(\mathbf{v}) = \mathbf{X}^{1/2} \exp(\mathbf{X}^{-1/2}\mathbf{v}\mathbf{X}^{-1/2})\mathbf{X}^{1/2} \quad (1)$$

$$\log_{\mathbf{X}}(\mathbf{Y}) = \mathbf{X}^{1/2} \log(\mathbf{X}^{-1/2}\mathbf{Y}\mathbf{X}^{-1/2})\mathbf{X}^{1/2} \quad (2)$$

where, manifold exponential operator, $\exp_{\mathbf{X}} : \mathcal{T}_{\mathbf{X}}\mathcal{M} \mapsto \mathcal{M}$, (Eqn. 1) maps the tangent vector $\mathbf{v}$ to the location on the manifold reached in a unit time by the geodesic starting at $\mathbf{X}$ in the tangent direction. Its inverse, the logarithmic operator, $\log_{\mathbf{X}} : \mathcal{M} \mapsto \mathcal{T}_{\mathbf{X}}\mathcal{M}$, (Eqn. 2) gives the vector in $\mathcal{T}_{\mathbf{X}}\mathcal{M}$ (with the smallest norm) corresponding to the geodesic from $\mathbf{X}$ to $\mathbf{Y}$. The geodesic distance between $\mathbf{X}$ and $\mathbf{Y}$ is given by

$$d^2(\mathbf{X}, \mathbf{Y}) = ||\log_{\mathbf{X}}(\mathbf{Y})||_{\mathbf{X}} = \text{tr}[\log^2(\mathbf{X}^{-1/2}\mathbf{Y}\mathbf{X}^{-1/2})] \quad (3)$$

Since the exponential map is a diffeomorphism (one-to-one, onto, and continuously differentiable mapping in both directions), the logarithm is uniquely defined at all the points on the manifold of Sym_n^+ [7]. However, it should be noted that the manifold operators are different from the standard *matrix* exp and log operators, which are defined as

$$\exp(\mathbf{A}) = \mathbf{U}diag(\exp(diag(\mathbf{D})))\mathbf{U}^T \quad (4)$$

$$\log(\mathbf{B}) = \mathbf{V}diag(\log(diag(\mathbf{S})))\mathbf{V}^T \quad (5)$$

where the eigen-decompositions of $\mathbf{A} \in Sym_n$ and $\mathbf{B} \in Sym_n^+$ are given by $\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{U}^T$ and $\mathbf{B} = \mathbf{V}\mathbf{S}\mathbf{V}^T$, respectively.

Most of the usual vector space operations using addition and subtraction can be reinterpreted on a Riemannian manifold via the exponential and logarithm maps (Eqns. 1,2). For instance the vector addition operation $\mathbf{Y} = \mathbf{X} + \overrightarrow{\mathbf{XY}}$ is equivalent to $\mathbf{Y} = \exp_{\mathbf{X}}(\overrightarrow{\mathbf{XY}})$ when the tangent vector $\overrightarrow{\mathbf{XY}} \in \mathcal{T}_{\mathbf{X}}\mathcal{M}$ identifies the geodesic starting at $\mathbf{X}$ and ending at $\mathbf{Y}$. Similarly, the vector subtraction operation $\overrightarrow{\mathbf{XY}} = \mathbf{Y} - \mathbf{X}$ can be realized by $\overrightarrow{\mathbf{XY}} = \log_{\mathbf{X}}(\mathbf{Y})$.

3.2. Linear Dynamical Systems on Sym_n^+

Let $\mathbf{X}(t) \in Sym_n^+$ represent the true state of a system at time t. $\mathbf{X}(t) = X_{ij}(t)$ is subscripted by two *covariant* indices $i, j \in [1 \ldots n]$. Furthermore, let $\mathbf{G}(t)$ be the $4-$mode state transition tensor of size $n \times n \times n \times n$ that characterizes the process which transforms the current system state $\mathbf{X}(t)$ to $\mathbf{X}(t+1)$. $\mathbf{G}(t) = G_{pq}^{ij}(t)$ is denoted by two *covariant* indices p, q, and two *contravariant* indices i, j. The $2-$mode process noise is represented by $\mathbf{w} \in Sym_n$. The **state update** equation of the linear dynamical system on the Sym_n^+ manifold is given by

$$\mathbf{X}(t+1) = \exp_{(\mathbf{G}(t)\mathbf{X}(t))}(\mathbf{w}) \quad (6)$$

Similarly, the observation model $\mathbf{H}(t) = H^{ij}_{pq}(t)$ is a $4-$mode tensor indexed by the two covariant and two contravariant subscripts, similar to $G^{ij}_{pq}(t)$. The $2-$mode observation noise is denoted by $\mathbf{v} \in Sym_n$. Therefore, at time t a **measurement** $\mathbf{Z}(t)$ of the true state $\mathbf{X}(t)$ is made according to

$$\mathbf{Z}(t) = \exp_{(\mathbf{H}(t)\mathbf{X}(t))}(\mathbf{v}) \tag{7}$$

Both the state and system noise tensors are assumed to be drawn from zero mean multivariate normal distributions, $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Omega})$ and $\mathbf{v} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Psi})$, with the $4-$mode covariance tensors, $\mathbf{\Omega} = E[\mathbf{w}\mathbf{w}^T]$ and $\mathbf{\Psi} = E[\mathbf{v}\mathbf{v}^T]$, respectively.

The reader is reminded of the tensor **contraction** operation used in the aforementioned system description and rest of the document. A tensor contraction is performed when one or more contravariant indices are same as the covariant indices. For example,

$$\mathbf{G}(t)\mathbf{X}(t) = G^{ij}_{pq}(t)x_{ij}(t) = \sum_{i=1}^{n}\sum_{j=1}^{n} G^{ij}_{pq}(t)x_{ij}(t) \tag{8}$$

3.3. Optimal Filter for Sym^+_n Manifold

As discussed in Sect. 3.1, $\log_{\mathbf{X}}$ can be used to realize the vector space subtraction operation on the Riemannian manifold. Moreover, since $\log_{\mathbf{X}} : \mathcal{M} \mapsto \mathcal{T}_{\mathbf{X}}\mathcal{M}$, and $\mathcal{T}_{\mathbf{X}}\mathcal{M}$ is a vector space, we can also interpret $\log_{\mathbf{X}}$ as a function that assigns unique vector space coordinates (w.r.t. $\mathbf{X}$) to any point on $\mathcal{M}$. A vector space is closed under finite vector addition and scalar multiplication, along with the existence of an additive inverse. This allows operations such as addition and subtraction of vectors in $\mathcal{T}_{\mathbf{X}}\mathcal{M}$. However, an important point to remember is that all vector coordinates (in $\mathcal{T}_{\mathbf{X}}\mathcal{M}$) for points on $\mathcal{M}$ are dependent on the base point $\mathbf{X}$, and hence do not remain constant as the base point is changed.

Let $\mathbf{X}(t)$, $\tilde{\mathbf{X}}(t)$, and, $\hat{\mathbf{X}}(t)$, denote the true, predicted, and corrected states of a linear system at time t, respectively. The predicted state $\tilde{\mathbf{X}}(t)$ is the *a priori* estimate of the state at t, given the knowledge of the process before t, and the corrected state $\hat{\mathbf{X}}(t)$ is the *a posteriori* state estimate at t, given measurement $\mathbf{Z}(t)$. In the following sections we will drop the temporal index (t) on all elements whenever it is possible without ambiguity. Also, we will proceed with the following derivation by centering all quantities in the tangent plane at a known location, $\mathbf{X_b} \in \mathcal{M}$, on the manifold. At a given time instance, the predicted and the corrected state errors (w.r.t. the true state) are given by

$$\tilde{\mathbf{e}} = \log_{\mathbf{X_b}}(\mathbf{X}) - \log_{\mathbf{X_b}}(\tilde{\mathbf{X}}) \tag{9}$$

$$\hat{\mathbf{e}} = \log_{\mathbf{X_b}}(\mathbf{X}) - \log_{\mathbf{X_b}}(\hat{\mathbf{X}}) \tag{10}$$

and the corresponding estimates of the error covariances (before and after observation assimilation) are given by

$$\tilde{\mathbf{\Gamma}} = E[\tilde{\mathbf{e}}\tilde{\mathbf{e}}^T] \tag{11}$$

$$\hat{\mathbf{\Gamma}} = E[\hat{\mathbf{e}}\hat{\mathbf{e}}^T] \tag{12}$$

Analogous to the derivation of the vector KF, we assume that the *a posteriori* state estimate $\hat{\mathbf{X}}$ is a linear combination of the *a priori* state estimate $\tilde{\mathbf{X}}$ and the weighted difference between an actual measurement $\mathbf{Z}$ and the predicted measurement $\mathbf{H}\tilde{\mathbf{X}}$. We can define $\hat{\mathbf{X}}$ in the vector space of the tangent plane at $\mathbf{X_b}$, and finally bring it back to the manifold (using $\exp_{\mathbf{X_b}}$), as shown below,

$$\hat{\mathbf{X}} = \exp_{\mathbf{X_b}}\Big[\log_{\mathbf{X_b}}(\tilde{\mathbf{X}}) + \mathbf{K}(\mathbf{\Delta Z})\Big] \tag{13}$$

$$\mathbf{\Delta Z} = \log_{\mathbf{X_b}}(\mathbf{Z}) - \log_{\mathbf{X_b}}(\mathbf{H}\tilde{\mathbf{X}}) \tag{14}$$

$$= \log_{\mathbf{X_b}}(\mathbf{H}\mathbf{X}) + \mathbf{v} - \log_{\mathbf{X_b}}(\mathbf{H}\tilde{\mathbf{X}}) \tag{15}$$

where $\mathbf{K}$ is the $4-$mode weighting tensor (equivalent to the Kalman gain matrix) and $\mathbf{\Delta Z}$ is the innovation. Intuitively, Eqn. 13 realizes the aforementioned linear combination by adding the scaled (by $\mathbf{K}$) innovation vector $\mathbf{\Delta Z}$ to the vector coordinates of $\tilde{\mathbf{X}}$ in the coordinate system centered at $\mathbf{X_b}$. Since the vector operations are performed in $\mathcal{T}_{\mathbf{X_b}}\mathcal{M}$, all quantities should be first converted to the appropriate coordinate system via the $\log_{\mathbf{X_b}}$ operator. Similarly, Eqn. 15 interprets the vector coordinates of the noisy observation $\mathbf{Z}$ (in $\mathcal{T}_{\mathbf{X_b}}\mathcal{M}$) as the vector addition operation of the vector coordinates of the ideal observation $\mathbf{H}\mathbf{X}$ at $\mathcal{T}_{\mathbf{X_b}}\mathcal{M}$ and the measurement noise vector $\mathbf{v}$. Importantly, Eqns. 13-15 assume that the observation space ($\mathbf{Z}$) is same as the state space ($\mathbf{X}$), and, we further restrict $\mathbf{H}$ to be an identity tensor, thus assuming that we can measure all components of the state space. This simplifies eqn. 13 to become

$$\hat{\mathbf{X}} = \exp_{\mathbf{X_b}}[(\mathbf{I}-\mathbf{K})\log_{\mathbf{X_b}}(\tilde{\mathbf{X}}) + \mathbf{K}\log_{\mathbf{X_b}}(\mathbf{X}) + \mathbf{K}\mathbf{v}] \tag{16}$$

Upon substituting Eqn. 16 into Eqns. 10 and 12, and performing some trivial manipulations, we obtain the *a posteriori* state estimation error and the error covariance as,

$$\hat{\mathbf{e}} = (\mathbf{I} - \mathbf{K})\tilde{\mathbf{e}} - \mathbf{K}\mathbf{v} \tag{17}$$

$$\hat{\mathbf{\Gamma}} = (\mathbf{I} - \mathbf{K})\tilde{\mathbf{\Gamma}}(\mathbf{I} - \mathbf{K})^T + \mathbf{K}\mathbf{\Psi}\mathbf{K}^T \tag{18}$$

To obtain a filter that is a least mean-square error estimator, we seek to minimize the expected value of the *a posteriori* state estimate (Eqns. 10, 17). This is equivalent to minimizing the trace of the *a posterior* error covariance matrix $\hat{\mathbf{\Gamma}}$ (Eqns. 12, 18). We can find the optimal value of $\mathbf{K}$ by minimizing the error function $\mathbf{J} = \text{tr}[\hat{\mathbf{\Gamma}}]$. The minimization is done by taking the partial derivative of $\mathbf{J}$ with respect to $\mathbf{K}$, and equating it to zero, *i.e.*,

$$\frac{\partial \mathbf{J}}{\partial \mathbf{K}} = \frac{\partial}{\partial \mathbf{K}}\text{tr}\Big[(\mathbf{I} - \mathbf{K})\tilde{\mathbf{\Gamma}}(\mathbf{I} - \mathbf{K})^T + \mathbf{K}\mathbf{\Psi}\mathbf{K}^T\Big] \tag{19}$$

$$= -2(\mathbf{I} - \mathbf{K})\tilde{\mathbf{\Gamma}} + 2\mathbf{K}\mathbf{\Psi} = \mathbf{0} \tag{20}$$

therefore, solving for the optimal value of $\mathbf{K}$ to be

$$\mathbf{K} = \tilde{\mathbf{\Gamma}}(\mathbf{\Psi} + \tilde{\mathbf{\Gamma}})^{-1} \tag{21}$$

and further simplifying the expression for error covariance by substituting Eqn. 21 in Eqn. 18 to get

$$\hat{\mathbf{\Gamma}} = (\mathbf{I} - \mathbf{K})\tilde{\mathbf{\Gamma}} \tag{22}$$

The weighting or the *blending* factor $\mathbf{K}$ can be interpreted as follows. As the measurement error covariance $\mathbf{\Psi} \rightarrow \mathbf{0}$, the blending factor $\mathbf{K} \rightarrow \mathbf{I}$, therefore giving a higher weight to the innovation, or in other words, having a stronger belief in the observation $\mathbf{Z}$ as opposed to the predicted measurement $\mathbf{H}\tilde{\mathbf{X}}$. On the other hand, if the *a priori* estimate of error covariance $\tilde{\mathbf{\Gamma}} \rightarrow \mathbf{0}$, the weight $\mathbf{K} \rightarrow \mathbf{0}$ and the actual measurement is weighted less than the predicted measurement $\mathbf{H}\tilde{\mathbf{X}}$, implying a higher belief in the process.

3.4. State and Error Propagation

Once the current estimate of the state gets updated by assimilating the observation, we need to provide a mechanism to propagate the estimate of the current state and the error covariance to obtain the *a priori* estimates for the next time step. The equivalent step is trivial in case of the vector KF since the vectors are free to translate in a Euclidean space, unlike on a manifold where the tangent vectors are tied to a base point. On a Riemannian manifold, we can still compare things locally, by using parallel transport, but not anymore globally. Since in our case, the Riemannian manifold of SPD matrices is afforded with an invariant metric, we can use some of its properties to obtain the desired mechanism for state/error propagation over time.

The *a priori* estimate of the state at $t+1$ can be obtained as $\tilde{\mathbf{X}}(t+1) = \mathbf{G}(t)\hat{\mathbf{X}}(t)$. In order to obtain a recursive solution for the error propagation to the next time step, we make some design choices in our linear dynamical model. Firstly, we restrict our state transfer tensor $\mathbf{G}$ to a special kind such that it can be decomposed as $\mathbf{G} = \mathbf{A} \otimes \mathbf{A}$, where $\otimes$ denotes the Kronecker tensor product and $\mathbf{A} \in GL_n(\mathbb{R})$ is any real invertible $n \times n$ matrix. This allows us to rewrite the expression $\mathbf{GX} = \mathbf{AXA}^T$. Such a restriction on $\mathbf{G}$ still allows us to model a wide range of linear systems, an example of which is shown in Sect. 4. Secondly, at each time we also propagate the base point $\mathbf{X_b}(t)$ according to the system transfer function, such that, $\mathbf{X_b}(t+1) = \mathbf{G}(t)\mathbf{X_b}(t)$. Therefore, the *a priori* estimate of the state error and the error covariance at $t+1$ is given by (see Appendix A),

$$\begin{aligned}
\tilde{\mathbf{e}}(t+1) &= \log_{\mathbf{X_b}(t+1)}(\mathbf{X}(t+1)) - \log_{\mathbf{X_b}(t+1)}(\tilde{\mathbf{X}}(t+1)) \\
&= \log_{\mathbf{A}(t)\mathbf{X_b}(t)\mathbf{A}(t)^T}(\mathbf{A}(t)\mathbf{X}(t)\mathbf{A}(t)^T) + \mathbf{w} - \\
&\quad \log_{\mathbf{A}(t)\mathbf{X_b}(t)\mathbf{A}(t)^T}(\mathbf{A}(t)\hat{\mathbf{X}}(t)\mathbf{A}(t)^T) \qquad (23) \\
\tilde{\mathbf{e}}(t+1) &= \mathbf{A}(t)\hat{\mathbf{e}}(t)\mathbf{A}(t)^T + \mathbf{w} \qquad (24) \\
\tilde{\mathbf{\Gamma}}(t+1) &= \mathbf{G}(t)\hat{\mathbf{\Gamma}}(t)\mathbf{G}(t)^T + \mathbf{\Omega} \qquad (25)
\end{aligned}$$

where Eqn. 23 represents the new true state $\mathbf{X}(t+1)$ as the previous state $\mathbf{X}(t)$ transformed by $\mathbf{G}(t)$ and corrupted by the noise vector $\mathbf{w}$.

Input: *a posteriori* estimates $\hat{\mathbf{X}}(t)$ and $\hat{\mathbf{\Gamma}}(t)$, observation $\mathbf{Z}(t)$, base point $\mathbf{X_b}(t)$
Output: Filtered estimates $\hat{\mathbf{X}}(t+1)$ and $\hat{\mathbf{\Gamma}}(t+1)$
STEP 1: Prediction (for next time step)

$$\begin{aligned}
\tilde{\mathbf{X}}(t+1) &= \mathbf{G}(t)\hat{\mathbf{X}}(t) \\
\mathbf{X_b}(t+1) &= \mathbf{G}(t)\mathbf{X_b}(t) \\
\tilde{\mathbf{\Gamma}}(t+1) &= \mathbf{G}(t)\hat{\mathbf{\Gamma}}(t)\mathbf{G}^T(t) + \mathbf{\Omega}
\end{aligned}$$

STEP 2: Update time-step ($t = t+1$)
STEP 3: Assimilate observation and update

$$\begin{aligned}
\mathbf{K} &= \tilde{\mathbf{\Gamma}}(\mathbf{\Psi} + \tilde{\mathbf{\Gamma}})^{-1} \\
\hat{\mathbf{X}} &= \exp_{\mathbf{X_b}}[(\mathbf{I} - \mathbf{K}) \log_{\mathbf{X_b}}(\tilde{\mathbf{X}}) + \mathbf{K} \log_{\mathbf{X_b}}(\mathbf{Z})] \\
\hat{\mathbf{\Gamma}} &= (\mathbf{I} - \mathbf{K})\tilde{\mathbf{\Gamma}}
\end{aligned}$$

Algorithm 1: Recursive filtering on Sym_n^+ Manifolds

3.5. Implementation Details

The prediction and update steps for one iteration of the filtering algorithm on the Riemannian manifold of SPD matrices are given in Algorithm 1. The initial values of the base point $\mathbf{X_b}(0)$, state estimate $\hat{\mathbf{X}}(0)$, error covariance $\hat{\mathbf{\Gamma}}(0)$, and noise covariances $\mathbf{\Omega}$ and $\mathbf{\Psi}$, are provided by the user.

Two important things to keep in mind are (a) the conversion between the $n \times n$ matrix representation and $m = n(n+1)/2-$dimensional vector representation of the Sym_n^+ and Sym_n matrices, and (b) the conversion between the full and compact representation of the $4-$mode covariance matrices such as $\hat{\mathbf{\Gamma}}$, $\tilde{\mathbf{\Gamma}}$, $\mathbf{\Omega}$, $\mathbf{\Psi}$. For conversion (a) we form a vector from the lower triangular part of the symmetric matrix, and vice versa. The conversion mentioned in (b) is required due to the fact that the $n \times n \times n \times n-$dimensional $4-$mode covariance tensor of a $2-$mode $n \times n$ symmetric matrix will be singular due to the linear dependency of the symmetric elements of the $2-$mode tensor. Therefore an equivalent, compact, non-singular, $m \times m-$dimensional representation of the $4-$mode tensor can be obtained by removing the linearly dependent coefficients due to the symmetric elements of the $2-$mode tensor. A conversion in the other direction is also possible by repeating the redundant components from the compact representation. We do not present the details of the aforementioned conversion in interest of space, but the reader is referred to literature on cartesian tensors [4] for similar transformations. These conversions have been used to derive Eqns. 18, 21, 25, *etc*.

4. Object Tracking With Covariance Features

Recent research has advocated the use of a covariance matrix of image features for tracking objects [8, 9] instead of the conventional histogram object representation models used in popular algorithms. The covariance features enable a compact representation of both the spatial and statistical

properties of an image patch. The covariance tracker in [8] finds the object location by comparing (using Eqn. 3) the covariance matrix at different image locations to the known object model. The appearance model is updated by finding the mean of the covariance matrices on the Riemannian manifold over a fixed *time window*.

Even though [8] provides a mechanism for tracking objects and updating the appearance model, and one can additionally use a standard vector KF to smooth the tracking outputs, a better approach, instead, would be to have a framework that can track objects, smooth the obtained trajectories, and update the appearance features in a unified manner. Moreover, the strategy of averaging over a fixed window is susceptible to failure during drastic changes of appearance and the quality of results depend on the windows size used for model update.

In this paper we present an application of our proposed filtering approach (Sect. 3) to the task of object tracking using covariance features. We will construct an augmented SPD state space that contains information regarding the object position, velocity, and the covariance matrix of its appearance features. The online filtering process on the Riemannian manifold will be used to recursively obtain the smoothed object location and the updated appearance model simultaneously. The recursive procedure will not require the system to maintain an explicit history of appearance features over a time window, and hence is expected to be robust to drastic appearance changes.

The state matrix characterizing the joint spatial-covariance state space used for tracking is given by

$$\mathbf{S}(t) = \begin{bmatrix} x(t) & 0 & 0 & 0 & \mathbf{0}_n^T \\ 0 & y(t) & 0 & 0 & \mathbf{0}_n^T \\ 0 & 0 & \delta x(t) & 0 & \mathbf{0}_n^T \\ 0 & 0 & 0 & \delta y(t) & \mathbf{0}_n^T \\ \mathbf{0}_n & \mathbf{0}_n & \mathbf{0}_n & \mathbf{0}_n & \mathbf{C}(t) \end{bmatrix} \tag{26}$$

where, $\mathbf{C}$ is the $n \times n$ covariance matrix of features corresponding to the image patch centered at $\{p_x, p_y\}$ (see Appendix B), having v_x and v_y as the $x-$ and $y-$components of object velocity, respectively. Elements $x, y, \delta x, \delta y \in (0, 1]$, are the normalized position and velocity coordinates of the object, respectively. The coordinates are normalized by the number of rows, r, and columns, c, in the image to get $x = p_x/c$, $y = p_y/r$, $\delta x = 0.5(v_x + c)/c$, and $\delta y = 0.5(v_y + r)/r$. Since $\mathbf{C} \in Sym_n^+$, and the diagonal elements $x, y, \delta x, \delta y > 0$, the state matrix $\mathbf{S}(t) \in Sym_{n+4}^+$ is assured to be SPD.

We consider a time varying state transition function $\mathbf{G}(t) = \mathbf{A}(t) \otimes \mathbf{A}(t)$ and a fixed observation model $\mathbf{H} = \mathbf{I}$. The $2-$mode tensor $\mathbf{A}$ has the following form

$$\mathbf{A}(t) = \begin{bmatrix} \lambda_x(t) & 0 & \mathbf{0}_{n+2}^T \\ 0 & \lambda_y(t) & \mathbf{0}_{n+2}^T \\ \mathbf{0}_{n+2} & \mathbf{0}_{n+2} & \mathbf{I}_{n+2} \end{bmatrix} \tag{27}$$

where, $\mathbf{I}_k$ is a $k \times k$ identity matrix. After each iteration, t, we calculate $\lambda_x(t+1) = \sqrt{(p_x(t) + v_x(t))/p_x(t)}$ and $\lambda_y(t+1) = \sqrt{(p_y(t) + v_y(t))/p_y(t)}$. The state transition function described above propagates the system state from t to $t+1$ and predicts the new object location based on the object dynamics (velocity). The state matrix coefficients corresponding to the object velocity and the appearance features are assumed to be stationary and only get updated due to observation assimilation and the model noise.

Lastly, we generate the measurement $\mathbf{Z}(t+1)$ for the filtering process at each frame as follows. We find the object location in the new frame by scanning the neighborhood of the previous object location $\{p_x(t), p_y(t)\}$ and selecting the location at which the covariance feature matrix has the smallest distance (Eqn. 3) to the current (filtered) object model $\mathbf{C}(t)$. Object velocity is calculated as a vector difference between the current and the previous object location. Both, the object position and velocity, are normalized between 0 and 1 (analogous to the coefficients in $\mathbf{S}(t)$). The covariance feature matrix corresponding to the new location becomes the right-lower sub-matrix of the $2-$mode observation tensor $\mathbf{Z}(t+1)$.

5. Experiments

In this section, we first present results on a synthetic dataset to estimate a constant tensor valued random variable by filtering noisy measurements. Next, we show the results of the tracking framework developed in Sect. 4.

5.1. Estimating a Constant Tensor

We conducted experiments with synthetic data on the space of Sym_3^+ manifold. A set of noisy tensor measurements was generated by adding white Gaussian noise to a constant tensor, $\mathbf{M}$. A random Gaussian tensor can be generated as follows: First, a $m = 3(3+1)/2 = 6-$dimensional vector of independent and normalized real Gaussian samples is obtained. This vector is multiplied by the square root of the desired (compact, $m \times m-$dimensional) covariance matrix. The Gaussian sample is converted back to the equivalent 3×3 matrix in the tangent space Sym_3, and brought back to the manifold using the $\exp_{\mathbf{M}}$ operator.

For this experiment, we set the system transfer function to $\mathbf{G} = \mathbf{I}_3 \otimes \mathbf{I}_3$, base point $\mathbf{X_b} = \mathbf{I}_3$, and initialized the error matrices to $\mathbf{\Omega} = \omega \mathbf{I}_6$, $\mathbf{\Psi} = \psi \mathbf{I}_6$, $\hat{\mathbf{\Gamma}}(0) = \gamma \mathbf{I}_6$. The parameters $\omega = 0.0001$ and $\gamma = 1$ were fixed in all cases, whereas, ψ was assigned the values $\{0.01, 0.1, 1.0\}$ for different experiments. To evaluate the performance of the proposed filtering approach, we compared the distance of the true state, $\mathbf{M}$, to the smoothed estimate, $\mathbf{M}_N$, of the state matrix after assimilating N observations. The distance, $d^2(\mathbf{M}, \mathbf{M}_N)$, was calculated using the Riemannian distance metric in Eqn. 3. The entire experiment was repeated mul-

	N	5	10	15	20	25	50	100	250	500
$\psi = 0.01$	$d^2(\mathbf{M}, \mathbf{M}_N)$	0.0357	0.0157	0.0127	0.0133	0.0051	0.0072	0.0074	0.0068	0.0076
	$\text{tr}(\hat{\mathbf{\Gamma}}(N))$	0.0155	0.0081	0.0065	0.0060	0.0058	0.0057	0.0057	0.0057	0.0057
$\psi = 0.1$	$d^2(\mathbf{M}, \mathbf{M}_N)$	0.3525	0.1706	0.1130	0.1025	0.0308	0.0407	0.0261	0.0223	0.0274
	$\text{tr}(\hat{\mathbf{\Gamma}}(N))$	0.1502	0.0681	0.0453	0.0350	0.0293	0.0205	0.0187	0.0187	0.0187
$\psi = 1.0$	$d^2(\mathbf{M}, \mathbf{M}_N)$	3.4498	1.9052	1.1225	1.1890	0.3600	0.3945	0.2468	0.1383	0.1819
	$\text{tr}(\hat{\mathbf{\Gamma}}(N))$	1.4640	0.6609	0.4281	0.3177	0.2535	0.1316	0.0789	0.0605	0.0597

Table 1. Distance between the true state and the filtered output, $d^2(\mathbf{M}, \mathbf{M}_N)$, and the trace of estimated error covariance, $\text{tr}(\hat{\mathbf{\Gamma}}(N))$, after filtering N observations. For this experiment the initial conditions were set to $\omega = 0.0001$ and $\gamma = 1$.

tiple times and the averaged results are presented.

Table 1 shows the geodesic distance between the smoothed state estimate and the true state for different values of N. We present the results for different values of the measurement noise ψ. The table also shows the trace of the corresponding error covariance obtained at the end of N^{th} iteration. The results show that the distance between the filtered output and the true state converges to a small value (*e.g.*, ~ 0.007 for the case of $\psi = 0.01$) after around 25 iterations. The trace of the error covariance matrix also stabilizes around the same time. Therefore, our algorithm converges quickly in terms of both the state and the error covariance estimation, resulting in an accurate estimation of the underlying constant SPD matrix. Furthermore, we computed the distances between all noisy measurements and the true state for the case of $\psi = 0.01$. The average distance of the noisy measurements from the true state was 0.1534. Most observations (498 out of 500) were at a distance greater than the convergence distance of 0.0076 in the given experiment. This further demonstrates the ability of the proposed system to filter out the noise and obtain an accurate estimate of the true SPD state.

Moreover, we can also see the effect of system and measurement noise on the accuracy of convergence. A higher measurement to system noise ratio (*e.g.*, ψ=1.0, ω=0.0001) takes longer time to converge and is slower to respond to the measurements as compared to the case where the ratio of the two noise values is small (*e.g.*, ψ=0.01, ω=0.0001). Also, the trace of the error covariance matrix converges to a higher value in the former case, implying a higher prediction uncertainty due to large measurement noise.

Lastly, we experimented with different values of the base point $\mathbf{X_b}$ and obtained similar results, independent of its choice. This further ascertains the correctness of our derivation, especially, the formulation of the error terms in Eqns. 9 and 10 (that accurately captures the difference between $\mathbf{X}$, $\tilde{\mathbf{X}}$, and $\hat{\mathbf{X}}$, on the manifold), and specification of the *a posteriori* state estimate $\hat{\mathbf{X}}$ centered at $\mathcal{T}_{\mathbf{X_b}}\mathcal{M}$ (Eqn. 16).

5.2. Tracking Objects With Appearance Update

We compare the performance of the proposed tracking approach (Sect. 4) with four other trackers formed by combination of some existing approaches. We refer to the baseline approach of [8] (without model update) as the "Cov" tracker. The baseline tracker along with the model update strategy based on averaging on Riemannian manifolds [8] is called "Cov+Mean" tracker. Algorithms obtained after smoothing the results of the aforementioned trackers by a standard vector KF are denoted as the "Cov+KF" and "Cov+Mean+KF" trackers, respectively.

An important advantage of the proposed tracking algorithm is its ability to simultaneously filter the position estimate and the appearance model of the object being tracked. The online filtering approach does not require one to keep a history of appearance models over a fixed time window. When averaging is done over a fixed temporal window, in the competing methods, the quality of the updated model can be affected by an improper choice of the window size. Instead, our algorithm dynamically weights the observations at each frame (using the weighting function $\mathbf{K}$) and hence can dynamically adapt to both *gradual* and *drastic* appearance changes. In other cases where the appearance changes are not drastic and other model update strategies perform equally well, our algorithm in addition to providing a smooth estimate of the state space can handle situations of occlusion and data corruption (in either position or appearance) without requiring additional modules.

Figure 1 shows tracking results of the various algorithms on four different sequences (one in each row). Images in each row correspond to different snapshots of the tracking results from each sequence. The red tracks correspond to the proposed approach. The green, yellow, blue, and cyan colored tracks correspond to the results of the Cov, Cov+KF, Cov+Mean, and Cov+Mean+KF algorithms, respectively. Appearance models for all algorithms were identically initialized at locations where the target was completely visible. The noise parameters for filtering techniques were identical in all sequences.

In the first two sequences there are drastic changes in the appearance of the object, either due to changes in orientation (*e.g.*, target turns around) or physical characteristics (*e.g.*, target removes the jacket). The second sequence is also complicated by the presence of another person that distracts the trackers due to similar appearance, additionally, occluding the target (partially) for a few frames. All algorithms, except the proposed filtering approach, fail to update

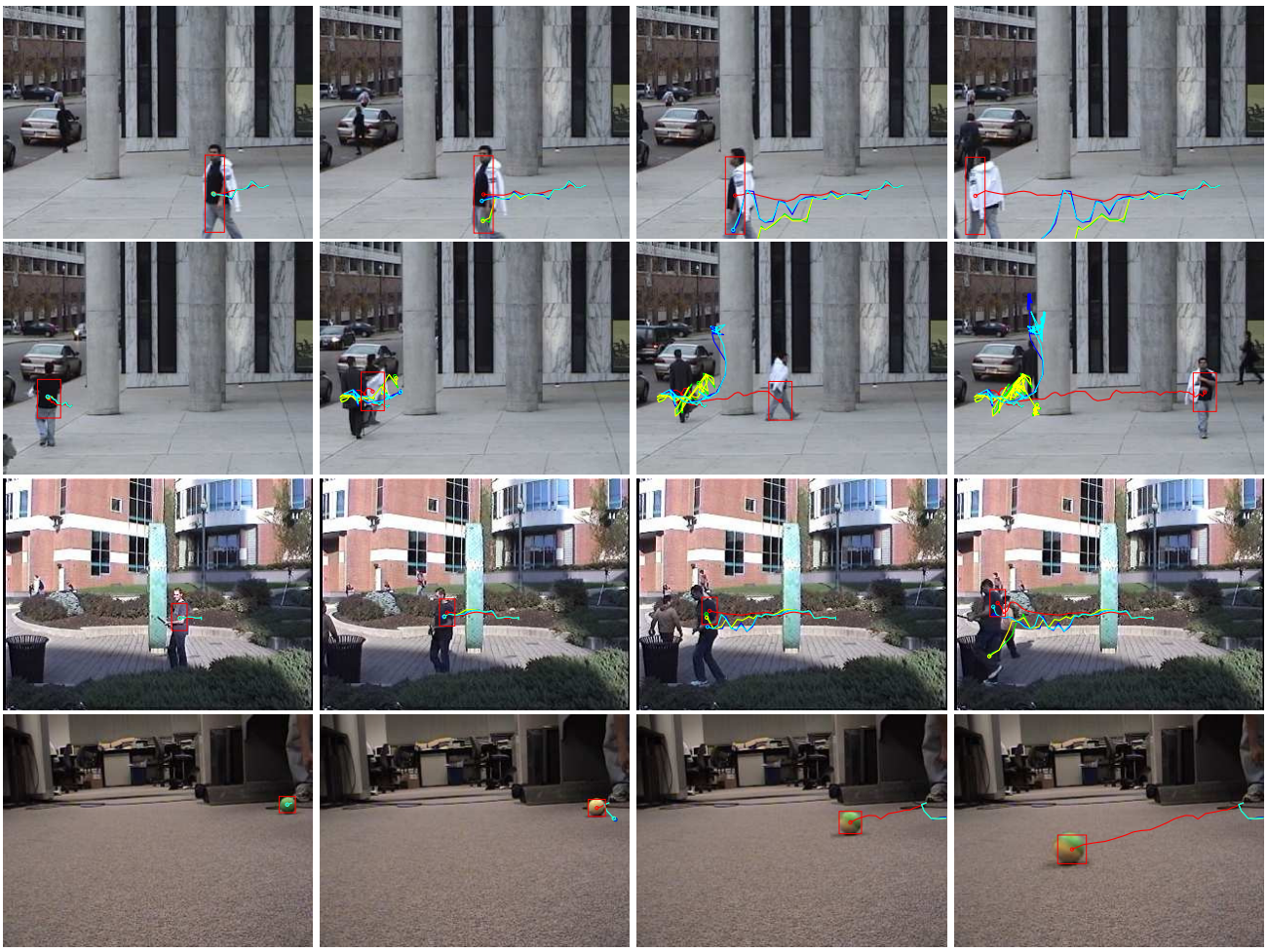

Figure 1. Tracking results of various algorithms on different sequences (one in each row). Trajectories are shown in color.

the covariance feature model in presence of drastic changes in the object's appearance, and therefore are unable to track the target for the entire duration. The addition of vector KF to the covariance algorithms only smoothes the trajectory generated by the corresponding algorithms, but is unable to improve the tracking performance.

Furthermore, even in cases where averaging based model update algorithm (Cov+Mean) is able keep up with the objects appearance (gradual changes), the tracking output produced by our algorithm is much smoother as compared to other approaches (*e.g.*, the third sequence in Fig. 1). Lastly, the fourth sequence shown in Fig. 1 is an extremely challenging example of the rapidly changing appearance model. All the competing algorithms fail to track the target since either they do not update the appearance model or they need to average over a time window for model update. The appearance of the target changes too quickly for these methods to update their models. Whereas, the proposed tracker adapts to the bimodal appearance of the target since the beginning of the sequence, tracking it successfully over time.

We further compare the model update strategies used by various algorithms. Figure 2 presents the geodesic distances (Eqn. 3) of the object models used by different algorithms to the ground truth object model at each frame for the first sequence used in Fig. 1. We do not show the results for the Cov+KF tracker as they are identical to that of the Cov tracker (since both algorithms do not update the appearance models). In the original sequence the object appearance changes from black to white between frames $60-100$. As shown in the plot, the appearance models employed by the different trackers start to become unreliable during this transition. The Cov (and Cov+KF) tracker is unable to follow the target object after ~ 80 frames due to the lack of model update mechanism. The Cov+Mean and the Cov+Mean+KF algorithms attempt to track the object for few more frames after which they are unable to handle the drastic changes in the object's appearance and fail to track the object. On the other hand, the proposed tracking algorithm, based on the online filtering approach for SPD manifolds, is able to dynamically adapt to the appearance changes by recursively assimilating observations (that include the covariance features) generated by the system.

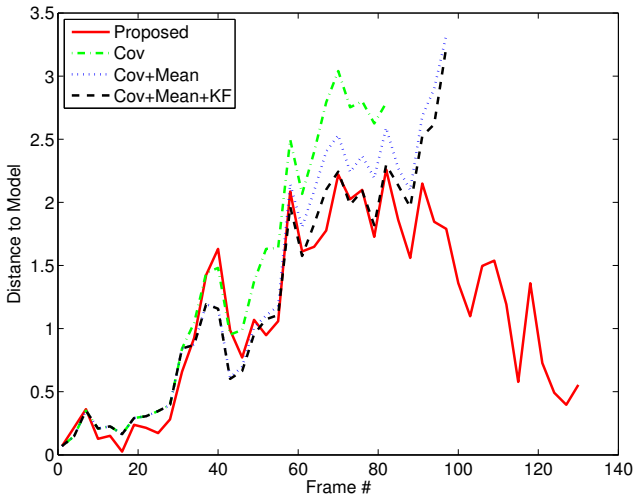

Figure 2. Distances between the appearance models used by different tracking algorithms to the ground truth appearance model of the object at each frame. Results correspond to the first sequence shown in Fig. 1.

6. Summary and Conclusion

We presented a recursive solution for optimal filtering of a symmetric positive definite state space, modeled as a linear dynamical process on Riemannian manifolds. The proposed framework is applicable to various domains such as control theory, continuum mechanics, medical imaging, and computer vision, that employ SPD matrices to represent information. The filter equations are derived in accordance with the differential-geometric properties of the Riemannian manifold of SPD matrices, and an iterative, predict-and-update style computational framework is described to obtain the smooth estimate of the state. Finally, we presented experimental results to demonstrate two applications of the proposed framework, namely the task of state estimation and visual tracking using covariance features.

7. Acknowledgements

This research was supported in part by the National Science Foundation under grant No. 0236653. We also wish to thank Mark Keck and Vinay Sharma for their comments and feedback during the production of this manuscript.

References

[1] W. M. Boothby. *An Introduction to Differentiable Manifolds and Riemannian Geometry*. Academic Press, 2002.

[2] D. Choukroun et al. State matrix kalman filter. In *Proc. IEEE Conf. Decision and Control*, pages 393–398, Dec 2003.

[3] P. T. Fletcher and S. Joshi. Riemannian geometry for the statistical analysis of diffusion tensor data. *IEEE Trans. Signal Proc.*, 87(2):250–262, 2007.

[4] H. Jeffreys. *Cartesian Tensors*. Cambridge, The University Press, 1931.

[5] R. Kalman. A new approach to linear filtering and prediction problems. *Trans. of the ASME-Journal of Basic Engineering*, 82 (Series D):35–45, 1960.

[6] M. Moakher. On the averaging of symmetric positive-definite tensors. *J. of Elasticity*, 82(3):237–296, March 2006.

[7] X. Pennec, P. Fillard, and N. Ayache. A Riemannian framework for tensor computing. *Int. J. of Comp. Vis.*, 66(1):41–66, 2006.

[8] F. Porikli, O. Tuzel, and P. Meer. Covariance tracking using model update based on lie algebra. In *Proc. Comp. Vis. and Pattern Rec.*, pages 728–735, 2006.

[9] A. Tyagi, J. Davis, and G. Potamianos. Steepest descent for efficient covariance tracking. In *Proc. Wkshp. Motion and Video Computing*, 2008.

A. Tangent Vector Transformation Under Action of Linear Groups

Consider the tangent vector $\gamma = \log_{\mathbf{X}}(\mathbf{Y})$ at $\mathbf{X}$, corresponding to the geodesic ending at $\mathbf{Y}$. Also recall the property of matrix logarithm $\log(\mathbf{C}^{-1}\mathbf{B}\mathbf{C}) = \mathbf{C}^{-1}\log(\mathbf{B})\mathbf{C}$ for any invertible matrix $\mathbf{C}$. By rearranging the terms of Eqn. 2 and using the above property we get $\gamma = \mathbf{X}\log(\mathbf{X}^{-1}\mathbf{Y})$.

Consider the action of the general linear group $\mathbf{A} \in GL$ that transforms the points $\mathbf{X}$ and $\mathbf{Y}$ to $\mathbf{A}\mathbf{X}\mathbf{A}^T$ and $\mathbf{A}\mathbf{Y}\mathbf{A}^T$, respectively. The tangent vector at $\mathbf{A}\mathbf{X}\mathbf{A}^T$ corresponding to the geodesic from $\mathbf{A}\mathbf{X}\mathbf{A}^T$ to $\mathbf{A}\mathbf{Y}\mathbf{A}^T$ is given by

$$\begin{aligned} \eta &= \log_{\mathbf{A}\mathbf{X}\mathbf{A}^T}(\mathbf{A}\mathbf{Y}\mathbf{A}^T) && (28)\\ &= (\mathbf{A}\mathbf{X}\mathbf{A}^T)\log\left((\mathbf{A}\mathbf{X}\mathbf{A}^T)^{-1}\mathbf{A}\mathbf{Y}\mathbf{A}^T\right) && (29)\\ &= \mathbf{A}\mathbf{X}\mathbf{A}^T\mathbf{A}^{-T}\log(\mathbf{X}^{-1}\mathbf{A}^{-1}\mathbf{A}\mathbf{Y})\mathbf{A}^T && (30)\\ &= \mathbf{A}\gamma\mathbf{A}^T && (31) \end{aligned}$$

Therefore, Eqn. 31 describes the transformation of the tangent vector due to the aforementioned group action.

B. Covariance Appearance Features

Let the feature vector $\mathbf{f}_k$ at pixel location (x, y), having the color triple (r, g, b), be denoted as $\mathbf{f}_k = [x\ y\ r\ g\ b]$. For a $W \times H$ patch centered at pixel $\mathbf{r} = (x_0, y_0)$, the covariance matrix of its features $\mathbf{f}_k$, $k = 1 \ldots WH$, is given by

$$\mathbf{F}_{\mathbf{r}} = \frac{1}{WH}\sum_{i=1}^{WH}(\mathbf{f}_i - \mu_{\mathbf{r}})(\mathbf{f}_i - \mu_{\mathbf{r}})^T \qquad (32)$$

where $\mu_{\mathbf{r}}$ is the mean feature vector for the pixels in the given image patch. We can normalize the $(i, j)^{th}$ element of $\mathbf{F}_{\mathbf{r}}$ by the product of the standard deviations of the i^{th} and the j^{th} feature dimension to obtain $\mathbf{C}_{\mathbf{r}}$ (*i.e.*, the matrix of correlation coefficients). Matrix $\mathbf{C}_{\mathbf{r}}$ still lies on the Sym_n^+ manifold and has the additional advantage that all values in the matrix $\in [-1, 1]$. In this paper, we employ normalized covariance matrices for tracking objects.