Regression on Lie Groups and Its Application to Affine Motion Tracking

Fatih Porikli

Abstract In this chapter, we present how to learn regression models on Lie groups and apply our formulation to visual object tracking tasks. Many transformations used in computer vision, for example orthogonal group and rotations, have matrix Lie group structure. Unlike conventional methods that proceed by directly linearizing these transformations, thus, making an implicit Euclidean space assumption, we formulate a regression model on the corresponding Lie algebra that minimizes a first order approximation to the geodesic error. We demonstrate our method on affine motions, however, it generalizes to any matrix Lie group transformations.

1 Introduction

Suppose we are given with a set of pairs $\{(\mathrm{M}_i, f_i)\}$ where M_i's are on an n-dimensional Lie group G and f_i's are their associated field vectors in $\mathbb{R}^d$. Our goal is to derive a regression function $\boldsymbol{\beta} : \mathbb{R}^d \mapsto G$ that approximates the corresponding point M on the Lie group for a vector f

$$\mathrm{M} = \boldsymbol{\beta}(f). \tag{1}$$

We take advantage of the Lie algebra $\mathfrak{g}$ and solve the corresponding linear regression problem instead

$$\log \mathrm{M} = f^T \Omega. \tag{2}$$

After a brief overview of Lie groups in Section 2, we define an approximate solution of (1) for matrix Lie groups in Section 3, and apply it to 2D affine motion tracking in Section 4. Part of the discussion can also be found in [24, 30].

Fatih Porikli
Australian National Univeristy and Data61/CSIRO, e-mail: fatih.porikli@anu.edu.au

2 Lie Group

A Lie group is a set G that is a group with the topology of an n-dimensional smooth differentiable manifold, in which the group operations multiplication $G \times G \mapsto G : (\mathrm{X}, \mathrm{Y}) \mapsto \mathrm{XY}$ and inversion $G \mapsto G : \mathrm{X} \mapsto \mathrm{X}^{-1}$ are smooth maps. In other words, the mapping $(\mathrm{X}, \mathrm{Y}) \mapsto \mathrm{X}^{-1}\mathrm{Y}$ is a smooth mapping of the product manifold $G \times G$ into G.

Some simple examples of Lie groups are the non-zero real numbers, the circle, the torus, the set of rotations of 3-dimensional space, the 3-sphere, and the set of square matrices that have nonzero determinant. Consider the sphere $S^2 \subset \mathbb{R}^3$ under rotations. The group property means that any two consecutive rotations of the sphere can also be done by rotating it over a single angle, and any rotation has an inverse, i.e rotating the sphere over an opposite angle. This shows the sphere has rotational symmetries. Since these rotations can be arbitrarily small and many small rotations adds up for a big rotation, these operations are smooth maps (it is indistinguishable from ordinary Euclidean space at small scales), therefore the rotation group $SO(3)$ acting on S^2 is a Lie group. This can be observed for the set of square matrices that have non-zero determinant. Such a matrix corresponds to a transformation of the space. The set of such transformations for a group: the matrices can be multiplied, each has an inverse, the multiplication is associative, and the identity transformation fixes each point of space. From these examples, we abstract the concept of a Lie group as a set of transformations or symmetries that has the structure of a smooth manifold, i.e. continuous symmetries.

Any Lie group gives rise to a Lie algebra. There is a corresponding connected Lie group unique up to covering to any finite-dimensional Lie algebra over real numbers. This correspondence between Lie groups and Lie algebras allows one to study Lie groups in terms of Lie algebras then transfer results from algebras back to groups. The tangent space to the identity element I of the group forms a Lie algebra $\mathfrak{g}$, which is a vector space together with a non-associative multiplication called Lie bracket $[\mathrm{x}, \mathrm{y}]$. Lie bracket is a binary operator over $\mathfrak{g} \times \mathfrak{g} \mapsto \mathfrak{g}$ defined as $[\mathrm{x}, \mathrm{y}] := \mathrm{xy} - \mathrm{yx}$ and satisfies the bilinearity, alternativity, and Jacobi identity axioms, i.e. $[a\mathrm{x} + b\mathrm{y}, \mathrm{z}] = a[\mathrm{x}, \mathrm{z}] + b[\mathrm{y}, \mathrm{z}]$, $[\mathrm{x}, \mathrm{x}] = 0$, $[\mathrm{x}, [\mathrm{y}, \mathrm{z}]] + [\mathrm{z}, [\mathrm{x}, \mathrm{y}]] + [\mathrm{y}, [\mathrm{z}, \mathrm{x}]] = 0$ for all scalars a, b and all elements $\mathrm{x}, \mathrm{y}, \mathrm{z} \in \mathfrak{g}$. We can reinterpret most of the properties of a Lie group into properties of the bracket on the Lie algebra.

The distances on a manifold are measured by the lengths of the curves connecting the points, and the minimum length curve between two points is called the geodesic. There exists a unique geodesic starting with vector $\mathrm{m} \in \mathfrak{g}$ at the group element I. The exponential map $\exp : \mathfrak{g} \to G$ maps the vector m to the point reached by this geodesic. Let $\exp(\mathrm{m}) = \mathrm{M}$, then the length of the geodesic is given by $\rho(\mathrm{I}, \mathrm{M}) = \|\mathrm{m}\|$. In general, the exponential map is onto but not one-to-one. Therefore, the inverse mapping $\log : G \to \mathfrak{g}$ is uniquely defined only around the neighborhood of I. If for any $\mathrm{M} \in G$, there exist several $\mathrm{m} \in \mathfrak{g}$ such that $\mathrm{M} = \exp(\mathrm{m})$, then $\log(\mathrm{M})$ is selected as the vector with the smallest norm. Left multiplication by the inverse of a group element $\mathrm{M}^{-1} : G \to G$ gives way to map the point M to I. The tangent

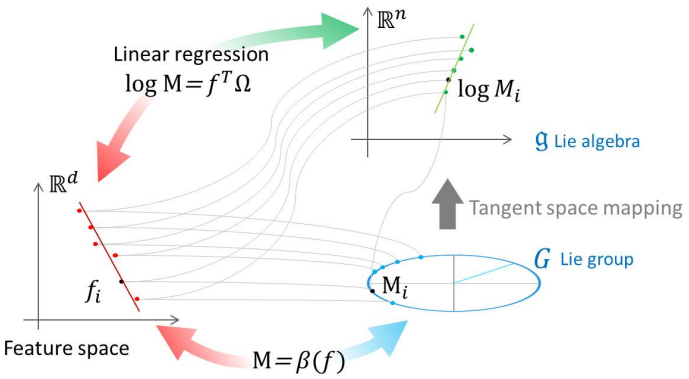

Fig. 1 Conceptual illustration of linear regression on Lie group.

space at I is the Lie algebra. The action of M^{-1} on the tangent space is through the adjoint action map. See [18] for more explanation.

Using the logarithm map and the group operation, the geodesic distance between two group elements is measured by

$$\rho(\mathrm{M}_1, \mathrm{M}_2) = \| \log(\mathrm{M}_1^{-1}\mathrm{M}_2)\|. \tag{3}$$

The norm above for the Euclidean space $\mathbb{R}^d$ with ordinary vector addition as the group operation is the Euclidean norm. How basis elements in the Lie algebra map to natural basis elements in R^d is not unique, which amounts to a choice of weighting, as explained in [8].

The exponential and logarithm maps for matrix Lie groups are given by the matrix exponential and logarithm operators

$$\exp(\mathrm{m}) = \sum_{k=0}^{\infty} \frac{1}{k!}\mathrm{m}^k \quad , \quad \log(\mathrm{M}) = \sum_{k=1}^{\infty} \frac{(-1)^{k-1}}{k}(\mathrm{M}-\mathrm{I})^k. \tag{4}$$

A comprehensive discussion on matrix manifolds and higher-order optimization methods on manifolds can be found in [1].

3 Linear Regression on Matrix Lie Groups

The regression function $\boldsymbol{\beta} : \mathbb{R}^d \mapsto G$ estimates the element M on the matrix Lie group G for a given d-dimensional feature vector f as $\mathrm{M} = \boldsymbol{\beta}(f)$. This concepts is illustrated in Figure 1.

The parameters of the regression function are learned from a set of N training pairs $\{(\mathrm{M}_i, f_i)\}$. Since these matrices are on a differentiable manifold, the sum of

the squared geodesic distances between the estimations $\boldsymbol{\beta}(f_i)$ and the given matrices M_i can be used as the loss function

$$L = \sum_{i=1}^{N} \rho^2 \left(\boldsymbol{\beta}(f_i), \mathrm{M}_i\right). \tag{5}$$

In general, the exponential map does not satisfy the identity $\exp(\mathrm{m}_1)\exp(\mathrm{m}_2) = \exp(\mathrm{m}_1 + \mathrm{m}_2)$. The Baker-Campbell-Hausdorff (BCH) formula [22] expresses the logarithm $\log(\exp(\mathrm{M}_1)\exp(\mathrm{M}_2))$ of the product of two Lie group elements as a Lie algebra element using only Lie algebraic operations for noncommutative Lie groups. A first order approximation to the BCH is

$$\log(\exp(\mathrm{M}_1)\exp(\mathrm{M}_2)) = \mathrm{M}_1 + \mathrm{M}_2 + \frac{1}{2}[\mathrm{M}_1, \mathrm{M}_2] + O(\mathrm{M}_1^2, \mathrm{M}_2^2) \tag{6}$$

using the Lie bracelet operator. Since the corresponding Lie algebra elements for M_1 and M_2 are $\mathrm{m}_1 = \log(\mathrm{M}_1)$ and $\mathrm{m}_2 = \log(\mathrm{M}_2)$, the geodesic distance can be approximated by

$$\begin{aligned} \rho(\mathrm{M}_1, \mathrm{M}_2) &= \|\log(\mathrm{M}_1^{-1}\mathrm{M}_2)\| \\ &= \|\log\left[\exp(-\mathrm{m}_1)\exp(\mathrm{m}_2)\right]\| \\ &= \left\|\mathrm{m}_2 - \mathrm{m}_1 + 0.5[-\mathrm{m}_1, \mathrm{m}_2] + O(\mathrm{m}_1^2, \mathrm{m}_2^2)\right\| \\ &\approx \|\mathrm{m}_2 - \mathrm{m}_1\|. \end{aligned} \tag{7}$$

Using (7), the loss function (5) can be approximated as

$$L \approx \sum_{i=1}^{N} \left\|\log\left(\mathrm{M}_i\right) - \log\left(\boldsymbol{\beta}(f_i)\right)\right\|^2. \tag{8}$$

up to the first-order terms. The approximation is good enough as long as the training samples are in a small neighborhood of the identity.

To formulate the the loss function in terms of a linear regression in a vector space, a matrix $\Omega : \mathbb{R}^d \mapsto \mathbb{R}^n$ that estimates the tangent vectors $\log\left(\mathrm{M}_i\right)$ on Lie algebra is defined

$$\boldsymbol{\beta}(f) = \exp\left(f^T \Omega\right) \tag{9}$$

where Ω is a $d \times n$ matrix of linear regression coefficients. Selecting d orthonormal bases on the Lie algebra, the matrix norm can be computed as the Euclidean distance between two vectors.

By taking the advantage of the Lie algebra, the tangent vectors at the identity $\log\left(\mathrm{M}_i\right)$ can be rearranged from matrix to n-dimensional vector form. Let $\mathbf{X}$ be a $N \times d$ matrix of row-wise arranged feature vectors, and $\mathbf{Y}$ be the corresponding $N \times n$ matrix of vector form mappings of the tangent vectors

$$\mathbf{X} = \begin{bmatrix} f_1 \\ \vdots \\ f_N \end{bmatrix} \quad \mathbf{Y} = \begin{bmatrix} \log(\mathrm{M}_1) \\ \vdots \\ \log(\mathrm{M}_N) \end{bmatrix}. \tag{10}$$

Then, the loss function (8) can be written as

$$L \approx \mathrm{tr}\big((\mathbf{Y} - \mathbf{X}\Omega)^T(\mathbf{Y} - \mathbf{X}\Omega)\big) \tag{11}$$

where the trace tr replaces the summation in (8). Differentiating the loss function with respect to Ω, the minimum is achieved at

$$\Omega = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}. \tag{12}$$

For rank deficient cases where the number of training samples is smaller than the dimension of the feature space $N < d$, the least squares estimate becomes inaccurate since $\mathbf{X}^T\mathbf{X}$ has determinant zero. To avoid overfitting, a penalty on the magnitude of the regression coefficients in the loss function is introduced

$$L \approx \mathrm{tr}\big((\mathbf{Y} - \mathbf{X}\Omega)^T(\mathbf{Y} - \mathbf{X}\Omega)\big) + \lambda\|\Omega\|^2 \tag{13}$$

which is also known as the *ridge regression* [12]. The minimizer of the loss function is given by

$$\Omega = (\mathbf{X}^T\mathbf{X} + \lambda\mathrm{I})^{-1}\mathbf{X}^T\mathbf{Y} \tag{14}$$

where I is an $d \times d$ identity matrix. The regularization coefficient λ determines the degree of shrinkage on the regression coefficients.

4 Application to Affine Motion Tracking

Locating an image region that undergoes 2D affine transformations is an essential task for camera motion estimation, pose invariant object recognition, and object tracking. In addition to challenging problems such as appearance changes, lighting variations, background clutters, and temporary occlusions, affine motion tracking confronts with computational issues due to the high dimensionality of the motion parameter space that induces an intractable number of hypotheses to be tested.

4.1 Related Work

Conventional methods often attempt to solve affine motion tracking in a vector space by state-space estimation [25, 14, 7, 2], template alignment [9, 5, 19] and feature correspondence [20, 13] approaches. State-space estimators assume affine tracking as a Markovian process and construct a probability density function of object param-

eters, which is supposed to be a normal distribution in case of Kalman filtering [7]. Due to this assumption, Kalman filters fail to describe multi-modal distributions, thus, Monte Carlo integration methods such as particle filters [14] are utilized. In theory, particle filter can track any parametric variation including affine motion. However, its dependency to random sampling induces degenerate likelihood estimations especially for the higher dimensional parameter spaces. Moreover, its computational requirements exponentially grow with the number of the state variables. In template alignment, the parametrized motion models -often more complex than affine motion- is estimated using appearance and shape models that are usually fitted by nonlinear optimization, e.g. iteratively solving for incremental additive updates to the shape parameters [9] or compositional updates to the warped models [4]. Alternatively, affine tracking can be formulated as a minimization on a cost function that consists of the sum of squares differences between the model instance obtained with a linear transformation and input image. However, rarely the relationship between the image intensity values and the model variation can be expressed in a linear form. To accommodate nonlinear transformations, stochastic gradient descent [25], relevance vector machine [28], Tikhonov regularization[2] are employed. One shortcoming of these algorithms is that they require computation of partial derivatives, Jacobian, and Hessian for each iteration, which makes them impractical. Several methods utilize feature point correspondences. Feature point based methods mainly differ in the type of features and descriptors, e.g. using SIFT [26], SURF [13], a combination of primitive features like simple differences between intensity values at randomly chosen locations [20], used for matching the object model to the current frame. The feature-point based trackers are highly sensitive to the available texture information on the object.

Tracking in general can also be regarded as a detection and model fitting problem. A typical tracking-by-detection framework is composed mainly of motion model, observation model and model updater [29, 23, 27]. Motion model generates a set of candidates which might contain the target in the current frame based on the estimation from the previous frame. Observation model judges whether a candidate is the target based on the features extracted from it. Model updater online updates the observation model to adapt the change of the object appearance. Conventional models range from histograms, templates, classifier ensembles, to more intricate appearance models such as region covariance matrices [21] where the matrix is updated on a manifold. Model fitting is considered as a classification problem in [3] by training an ensemble of classifiers with object and background pixels and integrating classifiers over time. More recently, [11] proposed directly predicting the change in object location between frames by an online structured output support vector machine. This method uniformly samples the state space to generate positive and negative support vectors. Such a brute force approach on a larger search window, however, is computationally intractable.

4.2 Tracking as a Regression Problem on Lie Group

We interpret object tracking task as a supervised learning problem and solve it using a regression function on the Lie algebra. We focus on 2D region motions that establish a matrix Lie group structure. The transformations that we are interested (affine motion $\mathrm{Aff}(2,\mathbb{R})$, similarity transform $S(2)$, Euclidean motion $SE(2)$, etc.) are closed subgroups of general linear group $GL(3,\mathbb{R})$, which is the group of 3×3 nonsingular square matrices. We develop formulation for 2D affine motion group, however the tracking method is applicable to any matrix Lie group structured motion transformation. A two-dimensional affine transformation $\mathrm{Aff}(2,\mathbb{R})$ is given by a 3×3 matrix M as

$$\mathrm{M} = \begin{pmatrix} \theta & \mathbf{t} \\ 0 & 1 \end{pmatrix} \tag{15}$$

where θ is a nonsingular 2×2 rotation matrix and $\mathbf{t} \in \mathbb{R}^2$ is a translation vector. The set of all affine transformations forms a matrix Lie group. The structure of affine matrices in (15) is a $d = 6$ dimensional manifold. The associated Lie algebra is the set of matrices

$$\mathrm{m} = \begin{pmatrix} \mathrm{U} & \mathbf{v} \\ 0 & 0 \end{pmatrix} \tag{16}$$

where, U is a 2×2 matrix and $\mathbf{v} \in \mathbb{R}^2$. The matrix m can be formed into a $d = 6$ dimensional vector by selecting the entries of U and $\mathbf{v}$ as an orthonormal basis.

The set of 2D affine transformations $\mathrm{Aff}(2,\mathbb{R})$ do not constitute a vector space, but rather a manifold that has the structure of a Lie group. Existing methods for the most part disregard this manifold structure and flatten the topology in a vector space. Vector forms cannot globally parameterize the intrinsic topology on the manifold in a homogeneous fashion, thus fail to accurately evaluate the distance between affine motion matrices causing unreliable tracking performance. There are only a few relevant work for parameter estimation on Lie groups, e.g. [10] for tracking an affine snake and [6, 24, 16] for tracking a template. However, [6] fails to account for the noncommutativity of the matrix multiplications thus the estimations are valid only around the initial transformation. [24] learned the correlation between affine motions and the observed descriptors using a regression model on Lie algebra. Inherent topology is considered by [16] where a conventional particle filter based tracker where the state dynamics are defined on a manifold using a log-Euclidean metric. However, none of these methods incorporate an efficient mechanism to incorporate object appearance changes.

Our formulation has several advantages. After learning the regression function, the tracking reduces to evaluating the function at the previous location, therefore it can be performed very fast. In addition, the framework gives flexibility to use any region descriptor.

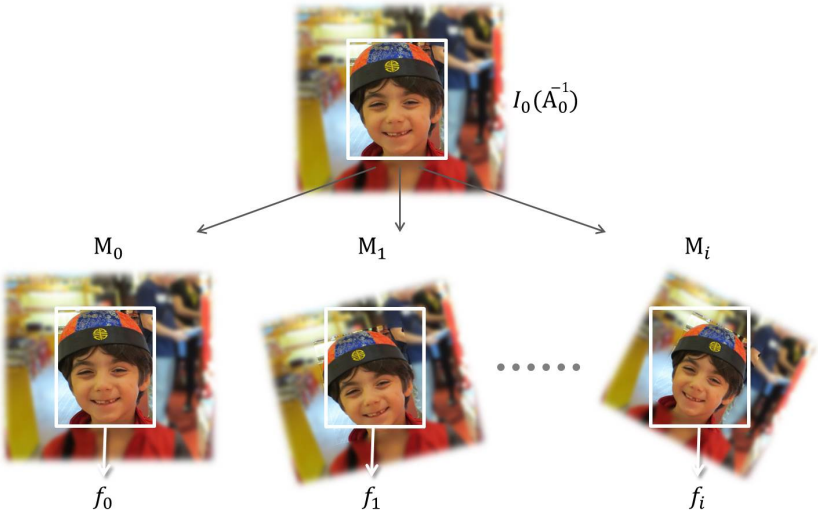

Fig. 2 Training samples are generated by applying N affine motions M_i at the object coordinates.

Learning Regression Function:

During the initialization of the tracking at frame I_0, we generate a training set of N random affine transformation matrices $\{\mathrm{M}_i\}$ around the identity matrix and compute their corresponding observed descriptors f_i within the initial object region to obtain the training set samples $\{f_i, \mathrm{M}_i\}$. The process is illustrated in Figure 2.

Specifically, the object coordinates are transformed by multiplying on the right with M_i^{-1} and the corresponding descriptor $f_i = f(I_0(\mathrm{A}_0^{-1}.\mathrm{M}_i^{-1}))$ is computed Using the initial location of the object A_0. The motion matrix A transforms a unit rectangle at the origin to the affine region enclosing the target object

$$[x_{im} \;\; y_{im} \;\; 1]^T = \mathrm{A}[x_{ob} \;\; y_{ob} \;\; 1]^T \tag{17}$$

where, the subscripts indicate the object coordinates and image coordinates respectively. The inverse transform A^{-1} is also an affine motion matrix and transforms the image coordinates to the object coordinates as illustrated in Figure 3. Notice that, the transformation A_0^{-1} moves the object region back to the unit rectangle and the image in the object coordinates is denoted as $I(\mathrm{A}_0^{-1})$.

The appearance of an object is described with an feature vector. We use only the pixel values inside the unit rectangle. Since we expect the feature vector to be an indicator of affine motion, we use a motion sensitive region feature. The target region is represented with a concatenated set of orientation histograms computed at a regular grid inside the unit rectangle in object coordinates (see Figure 3). With this, a d-dimensional vector $f\left(I(\mathrm{A}^{-1})\right) \in \mathbb{R}^d$ is obtained. The unit rectangle is divided

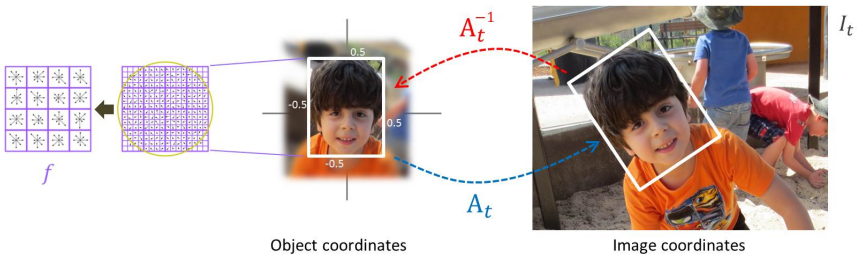

Fig. 3 The mapping and its inverse, between the object and image coordinates. The gradient weighted orientation histograms are utilized as region descriptors.

into $6 \times 6 = 36$ cells and a cell histogram is computed in each of them. Each histogram is quantized at $\pi/4$ degrees between 0 and 2π. The size of each histogram is eight dimensional and the descriptor is $d = 288$ dimensional. Similar to SIFT descriptors [17], the contribution of each pixel to the histogram is proportional to its gradient magnitude. During tracking the peripheral pixels are frequently contaminated by the background, hence we leave a 10% boundary outside the unit rectangle and construct the descriptor inside the inner rectangle.

After obtaining the training pairs, we form the data matrices as in (10) and apply (14) to learn the linear regression function Ω to model the correlation between the tangent space projection of the affine motion matrices and their corresponding observed descriptors.

Tracking Region in the Next Frame:

Tracking process estimates the transformation matrix A_t, given the observations $I_{0\ldots t}$ up to time t, and the initial location A_0. We model the transformations incrementally

$$\mathrm{A}_t = \mathrm{M}_t.\mathrm{A}_{t-1} \tag{18}$$

and estimate the increments M_t at each time. The transformation M_t corresponds to motion of target from time $t-1$ to t in the object coordinates. Given the previous location of the object A_{t-1} and the current image I_t, we estimate the incremental motion M_t by the regression function

$$\mathrm{M}_t = \exp\left(\left(f\left(I_t(\mathrm{A}_{t-1}^{-1})\right)^T \Omega\right).\right. \tag{19}$$

After learning the regression function Ω, the tracking problem reduces to estimating the motion via (19) using current observation I_t and updating the target location via (18). To better localize the target, at each frame we repeat the motion estimation using Ω a maximum of $K = 10$ times or the estimated incremental motion M_t becomes equal to identity.

Algorithm 1 Affine motion tracking

Require: Initial location A_0, images I_t, λ, γ, update frequency p, max iteration K

procedure TRAINING($t = 0$)
  Generate N motion matrices M_i, $i = 1 \dots N$
  Extract features $f_i = f\left(I_0(\mathrm{A}_0^{-1}.\mathrm{M}_i^{-1})\right)$
  Form $\mathbf{X}$, $\mathbf{Y}$
  Learn Ω by Eqn.14

procedure TRACKING($t > 0$)
  repeat
    $\mathrm{M}_t = \Omega f\left(I_t(\mathrm{A}_{t-1}^{-1})\right)$
    $\mathrm{A}_t \leftarrow \mathrm{M}_t.\mathrm{A}_{t-1}$
    $k \leftarrow k + 1$
  until $\mathrm{M}_t = \mathrm{I}$ or $k \leq K$
  if $\mathrm{mod}(t, p) = 0$ **then**
    Update Ω by Eqn.20
  $t \leftarrow t + 1$

Model Update:

Since objects can undergo appearance changes in time, it is necessary to adapt to these variations. In our case, we update the regression function Ω.

During tracking, we generate a set of random observations at each frame. The observations stored for last $p = 100$ frames constitute the update training set. Let $\mathbf{X}_p$ and $\mathbf{Y}_p$ be the new training set stored in the matrix form, and Ω_p be the previous model. After each p frames of tracking, we update the coefficients of the regression function by minimizing the loss

$$L \approx \mathrm{tr}\big((\mathbf{Y}_p - \mathbf{X}_p\Omega)^T(\mathbf{Y}_p - \mathbf{X}_p\Omega) + \lambda\|\Omega\|^2 + \gamma\|\Omega - \Omega_p\|^2.$$

The error function is similar to (13), but another constraint is introduced on the difference of regression coefficients. The minimum is achieved at

$$\Omega = (\mathbf{X}_p^T\mathbf{X}_p + (\lambda + \gamma)\mathrm{I})^{-1}(\mathbf{X}_p^T\mathbf{Y}_p + \gamma\Omega_p) \tag{20}$$

where the parameter γ controls how much change on the regression parameters are allowed from the last estimation. To take into account the bias terms all the function estimations are performed using centered data.

A pseudo-code of the tracking algorithm is given in Algorithm 1.

Experiments:

We compare the Lie algebra based parametrization with the linearization (21) around the identity matrix [9, 15, 28]

$$\mathrm{M}(x_0 + \delta x) \approx \mathrm{M}(x_0) + \frac{\partial \mathrm{M}}{\partial x}\delta x \tag{21}$$

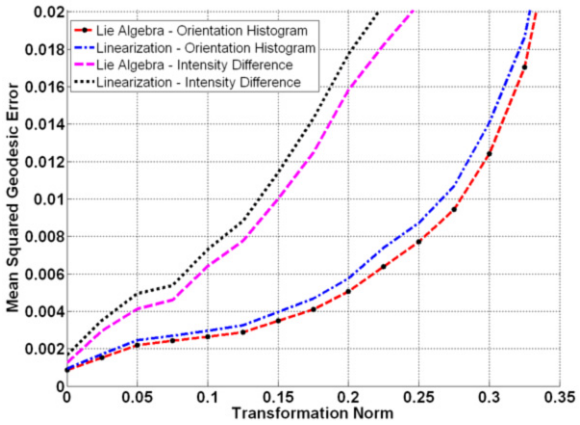

Fig. 4 Estimation errors of the Lie algebra and the linearization methods using orientation histograms and intensity features.

where $\mathrm{M}(x_0) = \mathrm{I}$, by measuring the estimation errors. We also compare orientation histograms with the intensity difference features used in optical flow estimation and tracking.

We generate a training set of $N = 200$ samples by random affine transformations of a single object. The motions are generated on the Lie algebra, by giving random values between -0.2 and 0.2 to each of the six parameters, and mapped to affine matrices via exponentiation. Since the size of the training set is large enough, there is no rank deficiency problem. The function Ω is estimated by ridge regression with $\lambda = 2.10^{-3}$ for orientation histograms and $\lambda = 5.0$ for intensity features, determined by cross validation. Each test set consists of $N_T = 1000$ samples. The samples inside a set have fixed norm. The norms $\|\log(\mathrm{M})\|$ vary from 0.025 to 0.35. We perform a single tracking iteration by each method, and measure the mean squared geodesic error

$$\frac{1}{N_T} \sum_{i=1}^{N_T} \rho^2 \left(\Omega f_i, \mathrm{M}_i \right) \tag{22}$$

between the estimations and the true values.

As shown in Figure 4, the estimation based on the Lie algebra is better than the linearization for transformation of all norms. The ratio is almost constant and on the average the linearization have 12% larger error. This is expected since our approach minimizes the sum of squared geodesic error. The estimations with orientation histograms are significantly better than the intensity based features.

We show sample tracking examples in Figure 5. In the experiments, the parameters of the ridge regression were $\lambda = \gamma = 2.10^{-3}$, which were learned offline via cross validation. The training dataset is generated on the Lie algebra, by giving random values between -0.1 and 0.1 to each of the six parameters. Although we track the targets with an affine model, these targets are not planar. Therefore, an affine model cannot perfectly fit the target but produces the best affine approxima-

Fig. 5 Sample affine tracking results. Target region boundaries are color-coded.

tion. Since nonplanar objects undergo significant appearance variations due to pose changes, the model update becomes important. The targets have large in-plane and off-plane rotations, translations, scale changes and occlusions. The estimations are accurate, which shows the robustness of the tracking approach.

References

1. M. R. Absil, P.A. and R. Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009. 3
2. T. Albrecht, M. Luthi, and T. Vetter. A statistical deformation prior for non-rigid image and shape registration. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008. 5, 6
3. S. Avidan. Ensemble tracking. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005. 6
4. S. Baker and I. Matthews. Equivalence and efficiency of image alignment algorithms. *Proc. IEEE Conf. on Computer Vision and Pattern Recognition,* Kauai, HI, 1:1090–1097, 2001. 6
5. S. Baker and I. Matthews. Lucas-Kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, 56(3):221–255, 2004. 5
6. E. Bayro-Corrochano and J. Ortegon-Aguilar. Lie algebra template tracking. *Proceedings of the 17th International Conference on Pattern Recognition*, 2:56–59, 2004. 7
7. Y. Boykov and D. Huttenlocher. Adaptive Bayesian recognition in tracking rigid objects. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition,* Hilton Head, SC, volume II, pages 697–704, 2000. 5, 6
8. G. Chirikjian. *Stochastic Models, Information Theory, and Lie Groups*. Birkhauser, Boston, 2011. 3
9. T. Cootes, G. Edwards, and C. Taylor. Active appearance models. In *Proc. European Conf. on Computer Vision,* Freiburg, Germany, pages 484–498, 1998. 5, 6, 10
10. T. Drummond and R. Cipolla. Application of Lie algebras to visual servoing. *Intl. J. of Comp. Vision*, 37:21–41, 2000. 7
11. S. Hare, A. Saffari, and P. H. S. Torr. Struck: Structured output tracking with kernels. In *IEEE International Conference on Computer Vision (ICCV)*, 2011. 6

12. T. Hastie, R. Tibshirani, and J. Freidman. *The Elements of Statistical Learning*. Springer, 2001. 5
13. W. He, T. Yamashita, H. Lu, and S. Lao. SURF tracking. In *International Conference on Computer Vision (ICCV)*, 2009. 5, 6
14. M. Isard and I. Blake. Condensation – conditional density propagation for visual tracking. In *Intl. J. of Comp. Vision*, volume 29, pages 5–28, 1998. 5, 6
15. F. Jurie and M. Dhome. Hyperplane approximation for template matching. *IEEE Trans. Pattern Anal. Machine Intell.*, 24:996–1000, 2002. 10
16. J. Kwon, K. M. Lee, and F. Park. Visual tracking via geometric particle filtering on the affine group with optimal importance functions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 7
17. D. Lowe. Distinctive image features from scale-invariant keypoints. *Intl. J. of Comp. Vision*, 60(2):91–110, 2004. 9
18. S. S. K. J. Ma, Y. and S. Sastry. *An invitation to 3-d vision: from images to geometric models*. Springer, 2003. 3
19. I. Matthews and S. Baker. Active appearance models revisited. *Intl. J. of Comp. Vision*, 60:135–164, 2004. 5
20. M. Ozuysal, P. Fua, and V. Lepetit. Fast keypoint recognition in ten lines of code. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007. 5, 6
21. F. Porikli, O. Tuzel, and P. Meer. Covariance tracking using model update based on Lie algebra. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006. 6
22. W. Rossmann. *Lie Groups: An Introduction Through Linear Groups*. Oxford Press, 2002. 4
23. A. W. M. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah. Visual tracking: An experimental survey. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 2014. 6
24. O. Tuzel, F. Porikli, and P. Meer. Learning on Lie groups for invariant detection and tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008. 1, 7
25. T. Vetter and T. Poggio. Linear object classes and image synthesis from a single example image. *IEEE Trans. Pattern Anal. Machine Intell.*, 19:733–742, 1997. 5, 6
26. D. Wagner, T. Langlotz, and D. Schmalstieg. Robust and unobtrusive marker tracking on mobile phones. In *ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, 2008. 6
27. N. Wang, S. Li, A. Gupta, and D. Yeung. Transferring rich feature hierarchies for robust visual tracking. *CoRR*, 2015. 6
28. O. Williams, A. Blake, and R. Cipolla. Sparse Bayesian learning for efficient visual tracking. *IEEE Trans. Pattern Anal. Machine Intell.*, 27:1292–1304, 2005. 6, 10
29. Y. Wu, J. Lim, and M. H. Yang. Online object tracking: A benchmark. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. 6
30. G. Zhu, F. Porikli, and H. Li. Lie-Struck: affine tracking on Lie groups using structured SVM. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2015. 1