Recognition of Human Gaits

A. Bissacco  F. Nori  S. Soatto

‡ Department of Computer Science, University of California, Los Angeles – CA 90095
† Department of Electrical Engineering, Washington University, St.Louis – MO 63130
Tel: (310)825-4840, Fax: (310)794-5057, email soatto@ucla.edu

Abstract

We pose the problem of recognizing human gaits in a decision-theoretic framework in the space of dynamical systems where each gait is represented. Established techniques are employed to track a kinematic model of a human body in motion, and the trajectories of the parameters are used to learn a representation of a dynamical system, which defines a gait. Likelihood ratios among dynamical systems are then computed, as well as distances between models. These computations are non trivial due to the fact that the space of canonical realizations of linear dynamical systems is not linear.

1 Introduction

How do we recognize a person walking from one jumping, running, hopping or dancing, independent on the person and her pose? In this paper we address this problem in three stages by first tracking a parametric skeletal model of the person moving, then learning a model that captures the dynamics of the model parameters, and finally posing the recognition problem in the space of dynamical systems learned from data. The first stage renders the system invariant to photometric factors (e.g. illumination, clothing etc.), while the second stage guarantees invariance to geometric factors such as the distance and pose of the camera, the length of the person’s limbs etc. In this context, each gait is represented by a dynamical system identified from a time series.

While we borrow the first stage straight from the literature, subsequent stages require some attention. Learning a dynamical model of the joint trajectories must be done in a canonical way to guarantee that a particular dataset corresponds to one and only one model. Performing recognition in the space of models entails computing distances and probability distributions on manifolds, since the set of dynamical models in canonical form is not a linear space (even if the model itself is linear!) and therefore computing distances naively can lead one to conclude that very similar models are dramatically different.

1.1 Relation to previous work and contribution of this paper

This work poses the problem of recognizing human gaits as computing likelihood ratios or nearest neighbors in the space of dynamical models representing joint trajectories. To the best of our knowledge this has never been done before. Implementing this program requires defining probability distributions on the space of models and computing geodesic distances.

Our starting point is a collection of trajectories of joint positions and/or joint angles for an articulated body. We extract those from images using an algorithm similar to that of Bregler and Malik [7], manually initialized. Although manual initialization is not our ideal choice, performing automatic initialization is a research program in its own right, and is therefore not addressed in this paper. The emphasis of our work is not tracking. Therefore, we consider the output of any tracking module as the input of our algorithm that estimates a dynamical model of the geometric feature trajectories.

The literature on modelling and recognition of human motion is sizeable and growing (see [13] for a survey). A common approach consists in extracting low-level features by local spatio-temporal filtering on the images and using hidden Markov models (HMM) on the collection of sequences of points thus obtained for recognition and classification tasks [29, 30]. In [30], parametric HMMs are introduced for recognizing gestures that exhibit dependence on a set of parameters, and in [6] coupled HMMs are used for modeling interactions of two mobile parts. In [23, 2] Bayesian Networks are used for recognition
Local representation of motion based on optical flow has been exploited in [4, 16, 21], and view-based methods are proposed in [5, 3, 15].

Other approaches are based on principal component analysis [32], parameterization of the motion on joint angles [8] and snake fitting [25]. Estimation of motion from stereo [31] and multiple view systems [14] have also been investigated.

Our models are discrete-time, continuous-state dynamical systems. Unlike hidden Markov models, where the “action” is coded by the sequence of discrete states, in our model the action is coded in the dynamical model (i.e. the system parameters, not its state). We use a recently developed closed-form algorithm to learn a model from data.

The second important part of this paper is recognition: since the space of model is non-linear, computing a distance between models is non-trivial. We draw on the literature of system identification and signal processing, where the problem is an active area of research [9, 24]. We propose different methods that, on the admittedly limited dataset we have tried them on, give similarly encouraging results.

2 Preliminaries

2.1 From images to skeletons

In this section we briefly describe the algorithm we use to extract joint trajectories from pictorial image sequences. We would like to stress that the emphasis of this work is on the recognition of gaits intended as dynamic models that generate joint trajectories, as we will see shortly. Therefore, any algorithm for the estimation of joint trajectory – for instance those described in Section 1.1 – can be used as a front-end.

In particular, we have implemented a variation of [7], where a human skeleton is represented as a kinematic chain supporting ellipsoidal texture patches. The parameters of the chain are set by hand by clicking on the desired joints in the first image of a sequence. We exploit symmetry by considering only half a skeleton. Each link is then represented by an ellipse whose major axis equals the length of the link and whose minor axis is also set by hand. At each tracking step, an expectation-maximization iteration is performed where the joint parameters are estimated for a given support region, followed by an update of the support region based on a local measure of similarity among corresponding regions in adjacent images.

The result of this algorithm – or of any other similar algorithm – is a sequence of joint positions, which we call $y(t), t = 1 \ldots \tau$. These are used to identify a dynamical model of the joint evolution, as we describe next.

2.2 From joint angle trajectories to dynamical models

We start from the assumption that a sequence of joint angle trajectories $y(t), t = 1 \ldots \tau$ is a realization from a second-order stationary stochastic process. This means that the joint statistics between two instants is shift-invariant. Although this may seem a severely restrictive assumption, we show that it is sufficient to characterize models that are general enough for the purpose of recognition.

It is well known that a positive definite covariance sequence with rational spectrum corresponds to an equivalence class of second order stationary processes [22]. It is then possible to choose as a representative of each class a Gauss-Markov model – that is the output of a linear dynamical system driven by white, zero-mean Gaussian noise – with the given covariance. In other words, we can assume that there exists a positive integer $n$, a process $\{x(t)\}$ (the “state”) with initial condition $x_0 \in \mathbb{R}^n \sim \mathcal{N}(0, P)$ and a symmetric positive semi-definite matrix $\begin{bmatrix} Q & S \\ S^T & R \end{bmatrix} \geq 0$ such that $y(t)$ is the output of the following Gauss-Markov “ARMA” model\(^1\):

\[
\begin{align*}
x(t + 1) &= A x(t) + v(t) \quad v(t) \sim \mathcal{N}(0, Q); \quad x(0) = x_0 \\
y(t) &= C x(t) + w(t); \quad w(t) \sim \mathcal{N}(0, R)
\end{align*}
\]

for some matrices $A \in \mathbb{R}^{n \times n}$ and $C \in \mathbb{R}^{m \times n}$.

The first observation concerning the model (1) is that the choice of matrices $A, C, Q, R, S$ is not unique, in the sense that there are infinitely many models that give rise to exactly the same measured covariance sequence starting from suitable initial conditions. The first source of non-uniqueness has to do with the choice of basis for the state space: one can substitute $A$

\(^1\)ARMA stands for autoregressive moving average.
with $WT^{-1}$, $C$ with $CT^{-1}$, $Q$ with $TQT^T$, $S$ with $TS$, and choose the initial condition $T x_0$, where $T \in \mathcal{G}(n)$ is any invertible $n \times n$ matrix and obtain the same output covariance sequence; indeed, one also obtains the same output realization. The second source of non-uniqueness has to do with issues in spectral factorization that are beyond the scope of this paper [22]. Suffices for our purpose to say that one can transform the model (1) into a particular form – the so-called “innovation representation” – that is unique. In order to be able to identify a unique model of the type (1) from a sample path $y(t)$, it is therefore necessary to choose a representative of each equivalence class (i.e. a basis of the state-space): such a representative is called a canonical model realization (or simply canonical realization). It is canonical in the sense that it does not depend on the choice of the state space (because it has been fixed).

While there are many possible choices of canonical realizations (see for instance [19]), we are interested in one that is “tailored” to the data, in the sense of having a diagonal state covariance. Such a model realization is called balanced [1]. The problem of going from data to models can then be formulated as follows: given measurements of a sample path of the process: $y(1), \ldots, y(\tau) \triangleright n$, estimate $A, C, R, Q$, a canonical realization of the process $\{y(t)\}$. Ideally, we would want the maximum likelihood solution from the finite sample, that is the argument of

$$\max_{A,C,R,Q} p(y(1), \ldots, y(\tau)|A,C,Q,R).$$

The closed-form asymptotically optimal solution to this problem has been derived in [26]. From this point on, therefore, we will assume that we have available – for each sample sequence – a model in the form $\{A,C,Q,R\}$. While the state transition $A$ and the output transition $C$ are an intrinsic characteristic of the model, the input and output noise covariances $Q$ and $R$ are not significant for the purpose of recognition (we want to be able to recognize trajectories measured up to different levels of noise as the same). Therefore, from this point on we will concentrate our attention on the matrices $A$ and $C$ that describe a gait.

### 2.3 Geometric structure of the space of models

Models, learned from data as described in the previous section, do not live on a linear space. While the matrix $A$ is only constrained to be stable (eigenvalues within the unit circle), the matrix $C$ has non-trivial geometric structure for its columns form an orthogonal set. The set of $n$ orthogonal vectors in $\mathbb{R}^n$ is a differentiable manifold called “Stiefel manifold”. In this section we present the basic elements of the geometry of Stiefel manifolds as they will be used in later sections. Let $C \in V(m,n)$, $m \geq n$ be a point on the Stiefel manifold of $n$-frames in $\mathbb{R}^m$, $C^TC = I_n$, endowed with the Euclidean metric $g(X,Y) = \text{tr}(X^TY)$ where $X,Y \in TV(m,n)$, the tangent plane to the Stiefel manifold. It is shown in [11] that geodesic trajectories in $V(m,n)$ have the general form (relative to the canonical metric)

$$R \exp(Xt)I_{m,n} \quad \text{where} \quad I_{m,n} = \begin{bmatrix} I_n & 0 \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{m \times n}$$

and $R \in O(m)$; $X$ is a skew-symmetric matrix having blocks

$$X = \begin{bmatrix} F & -G^T \\ G & 0 \end{bmatrix}.$$

Note that $X$ belongs to a linear space that is isomorphic to $\mathbb{R}^{mn-m(n+1)/2}$, and could therefore be used as a local coordinatization of the Stiefel manifold $V(m,n)$. We will use the structure of the geodesic in order to define a distance in $V(m,n)$ as follows: consider two points $C_1, C_2 \in V(m,n)$ and the geodesic connecting them: $C(t) | C(0) = [C_1 U] \exp(0)I_{m,n} = C_1$ and $C(t) = [C_1 U] \exp(Xt)I_{n,m} = C_2$ for a particular value of $X, t$ and for any $U$, an orthogonal completion of $C_1$. Then we define

$$d: V(m,n) \times V(m,n) \longrightarrow \mathbb{R}: (C_1, C_2) \mapsto \|Xt\|_F$$

where the subscript $F$ indicates the Frobenius norm. In order to provide a statistical description on the space of models, one would like to specify a class of probability distributions. In order to simplify the scenario, we assume that $A$ and $C$ are independent, so that their statistical description can be addressed separately. While specifying a probability density in the space of transition matrices $A$ is straightforward (indeed, we will adopt a Gaussian density), doing so for the output matrices $C$ is not trivial since, as we have just seen, the space has a non-trivial curvature.
In this paragraph we introduce a class of probability densities on the Stiefel manifold that can be used to model the statistics of $C$. Consider the following function $p : V(m,n) \to \mathbb{R}$

$$p(C) = \frac{1}{Z} \exp(\text{tr}(\Sigma \mu^T C))$$

where $\mu \in V(m,n)$, $\Sigma = \Sigma^T \geq 0$ and $Z = \int d\Sigma P(\Sigma)$ where $d\Sigma P(\Sigma) = p(\Sigma) d\Sigma(\Sigma)$ with $d\Sigma$ the base (Haar) measure on $V(m,n)$. The normalization factor $Z$ can be computed explicitly when $\Sigma = \sigma I$, in which case it can be shown that $Z = \frac{1}{\pi^{n/2} \Gamma\left(\frac{n}{2}\right)} e^{-\sigma^2}$ where $I_n$ is a Bessel function of the first type. We call this function a *Langevin (or Gibbs) density* on $V(m,n)$, owing to its similarity to Langevin distributions on the sphere. $\mu$ plays the role of the *mode* of the density, and $\Sigma$ plays the role of the *dispersion*. It is easy to verify that the above density has a unique maximum for $C = \mu$.

On the other hand, the mean (expectation) is not a good approximation of a set of samples. Indeed, in general, it does not even belong to $V(m,n)$. $m = \int C dP(C) \notin V(m,n)$

In order to estimate the sufficient statistics from samples, let $C_i$, $i = 1 \ldots N$ be a fair sample from the density $p(C)$. It follows from the law of large numbers that

$$m = \frac{1}{N} \sum_{i=1}^N C_i = \frac{1}{N} \sum_{i=1}^N \text{trace}(\Sigma \mu^T C_i).$$

Having a closed-form expression of the integral $m(\mu, \Sigma)$, one could use samples to compute $\hat{m}$ and use the equation above to compute statistics. However, we do not pursue that avenue further here. Instead, we consider the maximum likelihood estimation of the sufficient statistics by considering the joint density of a fair sample $p(C_1, \ldots, C_N|\mu, \Sigma)$, which can be written as

$$\prod_{i=1}^N p(C_i) = \frac{1}{\prod_{i=1}^N Z_i} \exp(\sum_{i=1}^N \text{trace}(\Sigma \mu^T C_i)).$$

For example, for the case $\Sigma = I$ we can look for $\hat{\mu}$ that solves the following problem

$$\max_{\mu \in V(m,n)} p(C_1, \ldots, C_N|\mu) = \max_{\mu \in V(m,n)} p(C_1, \ldots, C_N) = \max_{\mu \in V(m,n)} \mu^T \sum_{i=1}^N C_i.$$

Letting $\frac{1}{N} \sum_{i=1}^N C_i = U \Sigma \Sigma V^T$ be a singular value decomposition, then

$$\hat{\mu} = U \Sigma V^T.$$

### 2.4 Definition of gait

In order for the recognition problem to be well posed, we need a definition of what is to be recognized:

**Definition 1** Given a set of joint trajectories of an articulated model $\{y(t)\}$, we define a gait to be a pair $A, C$ with $A \in \mathcal{G} \mathcal{L}(n)$ a stable matrix and $C \in V(m,n)$ a matrix with orthonormal columns, such that there exist symmetric, positive semi-definite matrices $Q \in \mathbb{R}^{m \times m}$ and $R \in \mathbb{R}^{m \times m}$, a vector $x_0$ and a realization of two white, zero-mean, Gaussian white noise processes $\{v(t)\}$ and $\{w(t)\}$ such that (1) is satisfied.

Notice that the joint trajectories could be specified either in space $y(t) \in \mathbb{R}^d$ or on the image plane, in which case $y(t) \in \mathbb{R}^2$. The definition of gait given above captures the dynamics of the trajectory, and therefore it is to a large extent insensitive to the particular choice of joint representation.

One issue that we have not elaborated on is the choice of the model order $n$. This is performed empirically according to the measured value of the canonical correlations, as discussed in the experimental section 4.

### 3 Recognizing gaits

As we have articulated in the previous section, a gait is represented by a linear dynamical system and described by the matrices $A, C$ that live in the space $\mathcal{G} \mathcal{L}(n) \times V(m,n)$. This space has a non-trivial curvature structure that must be taken into account when doing comparisons between models.
In this section we consider two approaches to recognition. One involves computing likelihood ratios, with an explicit form for the probability density of models. The other only involves computing distances between models. For the preliminary experiments we performed, the two approaches exhibit similar performance, although extensive experimentation is yet to be carried out.

3.1 Likelihood ratios
Suppose that two groups of points on \( V(m, n) \) are given: \( U_1, \ldots, U_k \), fair samples from a Langevin with mean \( \mu_U \) and dispersion \( \Sigma_U \), and \( V_1, \ldots, V_i \) samples from a distribution with mean \( \mu_V \) and dispersion \( \Sigma_V \). Given a new point \( C \), we want to decide to which “group” it belongs. From a decision-theoretic point of view, the goal is to construct a density corresponding to each hypothesis, \( p(C|U) \), \( p(C|V) \) and to compute the likelihood ratio

\[
 p(C) = \frac{p(C|U)}{p(C|V)} \tag{9}
\]

where the parameters \( \Sigma_U \) and \( \mu_U \) in \( p(C|U) \) are computed from the samples \( U_i \) as above, and so for \( \Sigma_V \) and \( \mu_V \). A decision can then be made based on whether the ratio is larger or smaller than one. This setting can be generalized to decisions among a number of hypotheses in a straightforward fashion \[28\].

3.2 Distribution-independent clustering
From a pattern recognition viewpoint, constructing a probability density is not necessary to solve problems such as “clustering” or “grouping”. For instance the \( k \)-nearest neighbor algorithm only requires a distance to be implemented, and it is therefore easily extended to Stiefel manifolds under the notion of distance that we have defined. Suppose a set of samples \( C_1, C_2, \ldots \), is given, where each sample is labelled as belonging to one of \( c \) classes \( \lambda_j \). Given a new sample \( C \), the label \( \lambda_m \) is chosen by taking a vote among the \( k \) nearest samples. That is, \( \lambda_m \) is selected if the majority of the \( k \) nearest neighbors have label \( \lambda_m \), which happens with probability

\[
 \sum_{i=(k+1)/2}^{k} \binom{k}{i} P(\lambda_m|C)^i(1-P(\lambda_m|C))^{k-i}. \tag{10}
\]

It can be shown \[10\] that if \( k \) is odd the large-sample 2-class error rate is bounded above by the smallest concave function of \( P^* \) – the optimal error rate – greater than

\[
 \sum_{i=0}^{(k-1)/2} \binom{k}{i} \left( P^{*+1}(1-P^*)^k + P^{*k}(1-P^*)^{k+1} \right). \tag{11}
\]

Note that the analysis holds for \( k \) fixed as \( n \to \infty \), and that the rule approaches the minimum error rate for \( k \to \infty \). For small samples, there are no known results except negative counter-examples that show that an arbitrarily bad error rate can be achieved.

4 Experiments
In this section we describe some preliminary experiments in recognizing gaits. We have collected several sequences of human gaits, including walking, running, dancing, jumping etc., 10 per each gait. On each sequence, we have then defined a reduced kinematic model (half of the skeleton along a sagittal section, since the other half follows by symmetry and therefore does not provide additional information on the gait), and considered the time trajectory of the projection of 4 joints onto the image plane: shoulder, elbow, hip and knee. For each gait, we have changed the viewing position, distance and subject.

For each sequence of joint trajectories we have identified a dynamical model of order \( n = 4 \). We have decided the order of the model empirically by setting a threshold on the canonical correlation coefficients \[26\]. The Matlab code for identifying the model is reported in Figure 4.

We have then computed the mutual distance between each model in a number of ways: first we have computed the “naive” distance (the 2-norm of the difference between corresponding system matrices, without taking the geometry of the Stiefel
manifold into account), the geodesic distance between models, and the compounded distance between observability subspaces – also taking into account the geometry of the Stiefel manifold. Not surprisingly the first choice led to very disappointing results, while the latter two gave very similar results, with an advantage for the last one. The Matlab code for computing distances between models is reported in Figure 5.

For the purpose of illustration, we show the pictorial result of an experiment with three classes of motions that result in similar gaits: walking, running and going up and down a staircase. Notice that these three gaits are quite similar to each other (as opposed, say, to dancing or jumping), and yet the algorithm proposed is capable of distinguishing between them with high probability. In Figure 1 we show sample frames from the training datasets. Figure 2 shows the pairwise distance between each model in the dataset. As it can be seen, similar gaits result in smaller distances, with a few outliers. Although this is a very restricted database, it suffices to test our hypothesis. Acquiring an extensive dataset of gaits is part of our future research agenda.

![Sample frames from the dataset of the gaits: waking, running and walking a staircase.](image)

Figure 1: Sample frames from the dataset of the gaits: waking, running and walking a staircase.

We have now chosen a few sample sequences for each category as a test sequence. For each of the sequences we have estimated a model by first pre-processing the sequence (after manual initialization) using the ideas described in [7] to extract joint coordinates, and finally compared the models using a nearest neighbor criterion as described in Section 3.2. A sample frame from the test sequence is shown in Figure 3, while the first two corresponding nearest neighbors are shown to the right. Although this dataset is quite small, the discriminating power of the model as a representation of the dynamic sequence is visible.

Acknowledgements

This research is supported in part by NSF grant IIS-9876145, ARO grant DAAD19-99-1-0139 and Intel grant 8029.

References


Figure 2: The pairwise distance between each sequence in the dataset is displayed in this plot. Each row/column of a matrix represents a sequence, and sequences correspondence to similar gaits are grouped in block rows/columns. Dark indicates a small distance, light a large distance. The minimum distance is of course along the diagonal, and for each column the next closest sequence is indicated by a circle, while the second nearest is indicated by a cross.


Figure 3: For each gait we have chosen a few sample sequences (left) and computed the distance to every other sequence in the dataset. The closest sequence is shown in the central column, while the second nearest is shown in the right column. With a few exceptions, the nearest neighbor belongs to the same gait as the test sequence. Notice that these gaits are quite similar; analogous experiments performed on much more diverse gaits such as jumping or dancing return 100% correct classifications.
function [X,A,K,C,E] = canonical_realization(Y,n)

tau = size(Y,2);
Y1 = Y(:,1:tau-1);
Y2 = Y(:,2:tau);
Lf = chol(1/(tau - 1)*Y2*Y2')';
Lp = chol(1/(tau - 1)*Y1*Y1')';
[U S V] = svd(1/(tau -1)*inv(Lf)*Y2*(Y1')*inv(Lp'));
Un = U(:,1:n);
Vn = V(:,1:n);
Sn = S(1:n,1:n);
Sn_sqrt = diag(sqrt(diag(Sn)));
C = Lf*Un*Sn_sqrt;
X = Sn_sqrt*Vn'*inv(Lp)*Y;
X1 = X(:,1:tau-1);
X2 = X(:,2:tau);
A = 1/(tau - 1)*X2*(X1')*diag(1./diag(Sn));
E = Y - C*X;
K = (X2-A*X1)*pinv(E(:,1:tau-1));

Figure 4: Estimating the state-space model from the data requires only a few lines of Matlab code (notice that this code returns the canonical innovation model).

function d=subspace_distance(A1,C1,A2,C2)

n = size(A1,1);
O1 = C1;
O2 = C2;
for i = 1 : n
    O1 = [O1; C1*A1^i];
    O2 = [O2; C2*A2^i];
end
d = subspace(O1,O2);

Figure 5: Matlab code for computing the distance between models (the function subspace is a primitive of Matlab’s Identification Toolbox.)