# Combining Discriminative Features to Infer Complex Trajectories

**David A. Ross** DROSS@CS.TORONTO.EDU
**Simon Osindero** OSINDERO@CS.TORONTO.EDU
**Richard S. Zemel** ZEMEL@CS.TORONTO.EDU
Department of Computer Science, University of Toronto, Toronto, Ontario, CANADA M5S 3H5

## Abstract

We propose a new model for the probabilistic estimation of continuous state variables from a sequence of observations, such as tracking the position of an object in video. This mapping is modeled as a product of dynamics experts (features relating the state at adjacent time-steps) and observation experts (features relating the state to the image sequence). Individual features are flexible in that they can switch on or off at each time-step depending on their inferred relevance (or on additional side information), and discriminative in that they need not model the full generative likelihood of the data. When trained conditionally, this permits the inclusion of a broad range of rich features (for example, features relying on observations from multiple time-steps), and allows the relevance of features to be learned from labeled sequences.

## 1. Introduction

Many real-world problems involve estimating a time-series of continuous state vectors from a sequence of high-dimensional observations. Examples include inferring a trajectory of stock values based on the evolution of various economic indicators; tracking a patient's vital health signs through a myriad of symptoms; and finding the trajectory of a moving object in a video. A standard probabilistic approach is to fit the observations with a generative state-space model (SSM). These models propose that the state is a latent variable which evolves over time, and at each step is responsible for generating a noisy observation. State estimates are obtained from observations by inverting the probability model via Bayes' rule. A canonical example of a SSM is the Kalman filter, which models both the state dynamics and observations as linear functions of the state, corrupted by Gaussian noise, which leads to a Gaussian posterior distribution over the state at any time. Extensions of the Kalman filter allow for non-linearities in the state dynamics, and the relationship between the state and the observations, and also extend the posterior state distribution to be multi-modal.

Although highly successful, SSMs suffer some disadvantages. First, for computational tractability, SSMs usually assume that observations are conditionally independent of each other given the state. Thus the estimate of the state at a particular time can directly depend only on the observations at that same time (and the previous state), precluding the direct inclusion of evidence derived from observations across a range of time. Second, the relationship between state and observation at every time-step in an SSM is mediated through a single, identical likelihood function, which must generate the entire high-dimensional observation given only the value of the state. Crafting such a likelihood can be challenging, since it requires the ability to accurately model all aspects of the observation, including those that are irrelevant with respect to predicting the state.

An alternative approach, which we pursue here, is to directly model the conditional (posterior) distribution of the states given the observations. We propose fitting the posterior with a weighted log-linear combination of dynamics features (relating states at different time-steps) and observation features (relating the state to the observation sequence). Features are discriminative, leveraged to predict the state from the observations, thus avoiding the problem of explaining the high-dimensional observations faced by generative likelihoods. Using a conditional model also removes the need to assume independence of observations, hence each feature may incorporate evidence from any number of observations. Additionally, a wide variety of observation features may be combined and the system can learn, through supervised training, which features are relevant for any given task. Our system also includes the ability at each time-step to switch between

Appearing in *Proceedings of the 23$^{rd}$ International Conference on Machine Learning*, Pittsburgh, PA, 2006. Copyright 2006 by the author(s)/owner(s).

different dynamic features, and to selectively shut off unreliable observation features.

A canonical application of this approach involves tracking a moving object in a video as it follows a complicated trajectory. Consider watching a basketball game, and trying to just follow the ball. Generating a full description of the scene, including the complex interaction of all the players, based on the ball position at any time is hopeless. However, obtaining a estimate of the ball's location from the image is considerably easier, as features such as colour and shape can be highly discriminative. In addition, understanding the basic dynamics of the ball motion can be useful in tracking it even as it disappears behind some players.

We begin in Section 2 with a detailed description of our model, followed by details on inference and learning (Section 3). We relate our model to similar approaches in Section 4. Finally, in Section 5, we apply our model to a realistic tracking problem—estimating the position of a basketball in a video—using a number of different dynamics and observation features.

## 2. Model

Given a sequence of observations $\mathbf{Y}$ and corresponding sequence of states $\mathbf{X}$, we construct a model of the conditional distribution of $\mathbf{X}$ given $\mathbf{Y}$. The model combines a set of dynamics features $f_j(\mathbf{x}_{t-1}, \mathbf{x}_t)$ for $j = 1, \ldots, J$, and observation features $g_k(\mathbf{x}_t, \mathbf{Y})$ for $k = 1, \ldots, K$. The basic model then combines these features to provide a description of the conditional distribution:

$$\mathrm{P}(\mathbf{X}|\mathbf{Y}) \propto \exp\left(\sum_{j,t=2}^{J,T} f_j(\mathbf{x}_{t-1}, \mathbf{x}_t) + \sum_{k,t=1}^{K,T} g_k(\mathbf{x}_t, \mathbf{Y})\right)$$

In our model, both the dynamics and observation features can be viewed as functions that predict the state $\mathbf{x}_t$. That is, associated with each dynamics feature $f_j(\mathbf{x}_{t-1}, \mathbf{x}_t)$ is a function $\phi_j(\mathbf{x}_{t-1})$, and with each observation feature $g_k(\mathbf{x}_t, \mathbf{Y})$ a function $\gamma_k(\mathbf{Y}, t)$. Each feature then computes the distance between $\mathbf{x}_t$ and its respective function, scaled by some learned set of parameters:

$$f_j(\mathbf{x}_{t-1}, \mathbf{x}_t) = -\frac{1}{2}\left(\mathbf{x}_t - \phi_j(\mathbf{x}_{t-1})\right)^T \boldsymbol{\alpha}_j \left(\mathbf{x}_t - \phi_j(\mathbf{x}_{t-1})\right)$$
$$g_k(\mathbf{x}_t, \mathbf{Y}) = -\frac{1}{2}\left(\mathbf{x}_t - \gamma_k(\mathbf{Y}, t)\right)^T \boldsymbol{\beta}_k \left(\mathbf{x}_t - \gamma_k(\mathbf{Y}, t)\right).$$

The range of possible functions $\phi_j()$ and $\gamma_k()$ is broad, including any off-the-shelf method of predicting the state from other states or observations. In this paper we will restrict our attention to linear functions for the dynamics:

$$\phi_j(\mathbf{x}_{t-1}) = \mathbf{T}_j \mathbf{x}_{t-1} - \mathbf{d}_j$$

For example, in describing the motion of object in two dimensions, $\mathbf{x}_t$ can include a pair of components for its position, velocity, and acceleration, and $(\mathbf{T}_j, \mathbf{d}_j)$ could correspond to constant-velocity and no acceleration, or constant acceleration.

*Table 1.* A summary of notation

| | |
|---|---|
| $\mathbf{X} = [\mathbf{x}_1 \ldots \mathbf{x}_T]$ | state sequence |
| $\mathbf{Y} = [\mathbf{y}_1 \ldots \mathbf{y}_T]$ | observation sequence |
| $f_j(\mathbf{x}_{t-1}, \mathbf{x}_t)$ | $j^{th}$ dynamics feature function, |
| $\boldsymbol{\alpha}_j$ | its parameter |
| $\phi_j(\mathbf{x}_{t-1})$ | its prediction function |
| $u_{jt}$ | binary switch on dynamics |
| $\mathcal{F}_j(\mathbf{Y}, t)$ | dynamics switch potential |
| $g_k(\mathbf{x}_t, \mathbf{Y})$ | $k^{th}$ observation feature function, |
| $\boldsymbol{\beta}_k$ | its parameter |
| $\gamma_k(\mathbf{Y}, t)$ | its prediction function |
| $v_{kt}$ | binary switches on observations |
| $\mathcal{G}_k(\mathbf{Y}, t)$ | observation switch potential |

When including a large number of features, it is likely that at any given time, some of them give very poor predictions, and their contributions should be disregarded. This problem can be addressed by making the set of features flexible; at each time-step features can be turned off (meaning that their prediction will not be included in the state estimate) based on their inferred relevance. This is accomplished through the introduction of hidden binary *switch* variables, $u_{jt}$ and $v_{kt}$, one for each feature at each time-step.

Obtaining an accurate estimate of the state therefore is highly dependent on appropriately setting the switches, to only include the relevant features in the state representation. One piece of information potentially relevant to determining the switches involves evaluating the agreement between the feature predictions; intuitively a feature making a very divergent prediction can be switched off. Often, there is information available in the observations to suggest which dynamics/observation features might be relevant. This side information is captured by including learned potential functions for each switch, $\mathcal{F}_j$ and $\mathcal{G}_k$, which again can be off-the-shelf classifiers, trained discriminatively in the same framework.

Putting these together, we arrive at the following log-

probability of $\mathbf{X}$ given $\mathbf{Y}$:

$$\mathcal{L} = \log \sum_{\mathbf{u},\mathbf{v}} \exp\Bigg( \sum_{t,j} f_j(\mathbf{x}_{t-1}, \mathbf{x}_t) u_{jt} + \sum_{t,k} g_k(\mathbf{x}_t, \mathbf{Y}) v_{kt} + \sum_{t,j} \mathcal{F}_j(\mathbf{Y}, t) u_{jt} + \sum_{t,k} \mathcal{G}_k(\mathbf{Y}, t) v_{kt} \Bigg) - \log Z(\mathbf{Y}).$$

where $Z(\mathbf{Y})$ is the distribution's normalizing function, or *partition function*.

Note that because the switches are hidden, when we integrate over our uncertainty for the switches we effectively get a mixture-of-Gaussian prediction for the posterior state distribution at time $t$, which allows us to elegantly capture multimodality. Further, during inference, the posterior distribution over switches will capture the *probability* that a given feature is going to be of use at that point in the sequence.

The non-switching or "non-flexible" version of this model—where $u_{jt} = v_{kt} = 1$—is an interesting special case. Because the dynamics and observation features are quadratic in $\mathbf{X}$, the resulting conditional distribution is Gaussian. This provides some advantages: exact inference, partition function, and gradient computations can be done efficiently, and learning (with respect to $\boldsymbol{\alpha}_j$ and $\boldsymbol{\beta}_j$) becomes a convex problem. Practically, however, there are two considerable disadvantages. First, because the Gaussian is unimodal, the resulting state distribution will be unimodal at all time-steps; this can lead to an inability to recover from errors in the state prediction. Second, the prediction is no longer robust, which means that only dynamic features which are all simultaneously applicable can be included, and observation features must be always accurate.

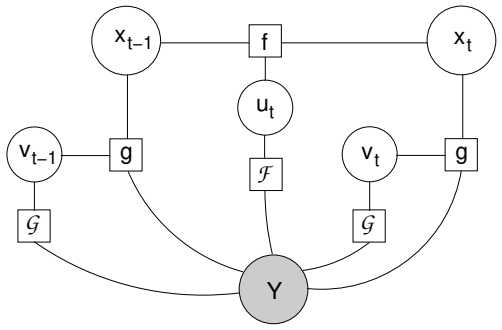


*Figure 1.* Factor graph of the model for two time-steps.

The parameters of the feature functions, $\boldsymbol{\alpha}_j$ and $\boldsymbol{\beta}_k$ can be learned using a straight-forward application of the Contrastive Divergence learning algorithm (Hinton, 2002). Contrastive Divergence is an approximate gradient-descent in parameter space, used in undirected graphical models with intractable partition functions.

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \boldsymbol{\alpha}_j} &= E\big[\sum_t (\mathbf{x}_t - \phi_j)(\mathbf{x}_t - \phi_j)^T | \mathbf{X}, \mathbf{Y}\big] \\
&\quad - E\big[\sum_t (\mathbf{x}_t - \phi_j)(\mathbf{x}_t - \phi_j)^T | \mathbf{Y}\big] \\
\frac{\partial \mathcal{L}}{\partial \boldsymbol{\beta}_k} &= E\big[\sum_t (\mathbf{x}_t - \gamma_k)(\mathbf{x}_t - \gamma_k)^T\big] | \mathbf{X}, \mathbf{Y}\big] \\
&\quad - E\big[\sum_t (\mathbf{x}_t - \gamma_k)(\mathbf{x}_t - \gamma_k)^T | \mathbf{Y}\big]
\end{aligned} \tag{1}$$

## 3. Inference

Given a sequence of observations, inferring the corresponding state sequence consists of computing the probability distribution $\mathrm{P}(\mathbf{X}|\mathbf{Y})$. Performing this calculation directly is infeasible, unfortunately, since it requires marginalization over all possible settings of the hidden variables, $u_{jt}$ and $v_{kt}$. However, several variational and approximation schemes readily apply to this formulation. Here we focus on a particular MCMC method that exploits special structure in the model to allow efficient approximate inference.

Given the hidden variables, the state sequence $\mathbf{X}$ forms an (undirected) linear-Gaussian Markov chain, thus $\mathrm{P}(\mathbf{X}|\mathbf{U}, \mathbf{V}, \mathbf{Y})$ can be readily computed. Similarly, given the state sequence, the switches are conditionally independent, so inference of $\mathrm{P}(\mathbf{U}, \mathbf{V}|\mathbf{X}, \mathbf{Y})$ is easy. From these facts, we arrive at a simple method for drawing samples from $\mathrm{P}(\mathbf{X}, \mathbf{U}, \mathbf{V}|\mathbf{Y})$.

1. Obtain an initial estimate $\hat{\mathbf{U}}, \hat{\mathbf{V}}$ of the switch variables. For example, these can be based on the side-information provided by features $\mathcal{F}_j$ and $\mathcal{G}_k$.
2. Infer $\mathrm{P}(\mathbf{X}|\hat{\mathbf{U}}, \hat{\mathbf{V}}, \mathbf{Y})$, a Gaussian in $\mathbf{X}$, and draw from it a state sequence sample $\hat{\mathbf{X}}$.
3. Infer $\mathrm{P}(\mathbf{U}, \mathbf{Y}|\hat{\mathbf{X}}, \mathbf{Y})$, and from it draw samples of the switches $\hat{\mathbf{U}}$ and $\hat{\mathbf{V}}$.
4. Goto 2, and repeat this sampling procedure for the desired number of iterations.

We now present a message-passing scheme for inferring the state given the switches, followed by a pair of simple equations for inferring the switches given the state. Each iteration of this scheme (and of learning) has a computational cost that scales linearly in the length of the sequence and the number of observation and dynamics features, but, like Kalman smoothing, scales cubically in the dimensionality of the state.

### 3.1. Inferring State X given Switches U, V

Given the switches and observations, the belief propagation algorithm can be used to exactly compute the marginal $\mathrm{P}(\mathbf{x}_t|\mathbf{U},\mathbf{V},\mathbf{Y})$ and pairwise marginal $\mathrm{P}(\mathbf{x}_{t-1},\mathbf{x}_t|\mathbf{U},\mathbf{V},\mathbf{Y})$ distributions. These can be used to draw samples of the state sequence, as well as to compute the expectations (1) required for learning. Note that because the features are quadratic in $\mathbf{X}$, the marginals and pairwise marginals will be Gaussian distributions.

Inference using belief propagation requires a two-phase message passing schedule, much like the Kalman smoother; messages are passed forward, from the beginning of the state sequence to the end, and backward in the opposite direction. Each message consists of a Gaussian distribution, with mean vector $\mu$ and precision matrix $\boldsymbol{\tau}$ (which, for notational convenience, we will use in place of the inverse covariance matrix). As in Kalman smoothing, the messages can be written recursively, each in terms of the message preceding it. We will break this into four steps: forward prediction of $\mathbf{x}_t$ given $\mathbf{x}_{t-1}$, forward correction to incorporate observations at time $t$, backward prediction of $\mathbf{x}_t$ given $\mathbf{x}_{t+1}$, and backward correction.

To begin with, we define a number of terms which will appear several times in the message-passing equations. Note that because they depend on the switches, these terms must be recomputed at each time-step.

$$\mathbf{A} = \sum_j \alpha_j u_{jt} \qquad \mathbf{B} = \sum_k \beta_k v_{kt}$$

$$\alpha T = \sum_j \alpha_j \mathbf{T}_j u_{jt} \qquad T\alpha T = \sum_j \mathbf{T}_j^T \alpha_j \mathbf{T}_j u_{jt}$$

$$\alpha d = \sum_j \alpha_j \mathbf{d}_j u_{jt} \qquad T\alpha d = \sum_j \mathbf{T}_j \alpha_j \mathbf{d}_j u_{jt}$$

$$\mathbf{y}'_t = \sum_k \beta_k \gamma_k(\mathbf{Y},t) v_{kt}$$

Forward Prediction:

$$\boldsymbol{\tau}_{t|t-1} = \mathbf{A} - \alpha T(T\alpha T + \boldsymbol{\tau}^f_{t-1})^{-1}\alpha T^T$$
$$\boldsymbol{\mu}_{t|t-1} = (\boldsymbol{\tau}_{t|t-1})^{-1}[\alpha T(T\alpha T + \boldsymbol{\tau}^f_{t-1})^{-1} (\boldsymbol{\tau}^f_{t-1}\boldsymbol{\mu}^f_{t-1} - T\alpha d) + \alpha d]$$

Incorporating Evidence (forward correction):

$$\boldsymbol{\tau}^f_t = \boldsymbol{\tau}_{t|t-1} + \mathbf{B} \qquad \boldsymbol{\mu}^f_t = (\boldsymbol{\tau}^f_t)^{-1}(\boldsymbol{\tau}_{t|t-1}\boldsymbol{\mu}_{t|t-1} + \mathbf{y}'_t)$$

Backward Prediction:

$$\boldsymbol{\tau}_{t|t+1} = T\alpha T - \alpha T^T(\mathbf{A} + \boldsymbol{\tau}^b_{t+1})^{-1}\alpha T$$
$$\boldsymbol{\mu}_{t|t+1} = (\boldsymbol{\tau}_{t|t+1})^{-1}[\alpha T^T(\mathbf{A} + \boldsymbol{\tau}^b_{t+1})^{-1} (\boldsymbol{\tau}^b_{t+1}\boldsymbol{\mu}^b_{t+1} + \alpha d) - T\alpha d]$$

Backward Correction:

$$\boldsymbol{\tau}^b_t = \boldsymbol{\tau}_{t|t+1} + \mathbf{B} \qquad \boldsymbol{\mu}^b_t = (\boldsymbol{\tau}^b_t)^{-1}(\boldsymbol{\tau}_{t|t+1}\boldsymbol{\mu}_{t|t+1} + \mathbf{y}'_t)$$

The marginal distribution of $\mathbf{x}_t$ can be obtained by multiplying all messages coming into it:

$$\boldsymbol{\tau}_t = \boldsymbol{\tau}_{t|t-1} + \mathbf{B} + \boldsymbol{\tau}_{t|t+1}$$
$$\boldsymbol{\mu}_t = (\boldsymbol{\tau}_t)^{-1}(\boldsymbol{\tau}_{t|t-1}\boldsymbol{\mu}_{t|t-1} + \mathbf{y}'_t + \boldsymbol{\tau}_{t|t+1}\boldsymbol{\mu}_{t|t+1}).$$

The pairwise marginal distribution is obtained by multiplying together the forward message into $\mathbf{x}_{t-1}$, the backward message into $\mathbf{x}_t$, and an additional factor arising from the dynamics features. The resulting mean is simply the concatenation of the marginal means, $(\mathbf{x}_{t-1},\mathbf{x}_t)$, and the precision is sum of the precisions from the messages and dynamics factor

$$\boldsymbol{\tau}_{t-1,t} = \begin{bmatrix} \boldsymbol{\tau}_{t-1|t-2} + \mathbf{B} + T\alpha T & -\alpha T^T \\ -\alpha T & \mathbf{A} + \mathbf{B} + \boldsymbol{\tau}_{t|t+1} \end{bmatrix}.$$

### 3.2. Inferring Switches U, V given State X

As mentioned above, given the state sequence the switch variables are independent, thus $\mathrm{P}(\mathbf{U},\mathbf{V}|\mathbf{X},\mathbf{Y})$ factorizes into a product of simple distributions.

The posteriors of the observation switches $v_{kt}$ are independent Bernoulli distributions, with probability

$$\mathrm{P}(v_{kt}=1) = \sigma\left(\exp(g_k(\mathbf{x}_t,\mathbf{Y}) + \mathcal{G}_k(\mathbf{Y},t))\right)$$

where $\sigma()$ is the *logistic* function $\sigma(x) = 1/(1+e^{-x})$. The posterior distribution of the dynamics switches $u_{jt}$ at each time-step is Multinomial—a discrete choice over $J$ options. The probability that switch $k$ is on at time $t$ is

$$\mathrm{P}(u_{jt}=1) = \frac{\exp(f_j(\mathbf{x}_{t-1},\mathbf{x}_t) + \mathcal{F}_j(\mathbf{Y},t))}{\sum_{j'}\exp(f_{j'}(\mathbf{x}_{t-1},\mathbf{x}_t) + \mathcal{F}_{j'}(\mathbf{Y},t))}.$$

## 4. Related Work

Our approach has notable similarities to previous work using Conditional Random Fields (CRFs) (Lafferty et al., 2001) and also Products-of-Experts (PoE's) and energy-based models (Hinton, 2002; Teh et al., 2003), and indeed in some ways can be considered a variant or extension of either of these frameworks. However, typically CRFs are concerned with discrete states and include simple, discrete features (such as delta-function indicator-variables). In contrast our model works with a combination of continuous and discrete state, incorporates unobserved latent variables during training and testing, and employs continuous-valued feature functions. Some of these elements have been used

individually in other models proposed recently, such as (Quattoni et al., ; Sudderth et al., ), but to our knowledge the particular combination of these elements is novel to the model presented here. Likewise, PoE's or energy-based models have been predominantly applied to modelling the full joint density, rather than the conditional posterior as in our case. Within this class of models, our work has some commonalities with exponential family harmoniums (Welling et al., ), where we use Bernoulli/Multinomial and Gaussian layers, with the extension in our model that this distribution is *conditioned* on a set of observation features, and that the Gaussians units have a linear chain dependency. Our work also bears some similarities to the Fields-of-Experts (Roth & Black, 2005), in the sense that we learn an undirected, translation-invariant dependency structure.

Our model draws additional inspiration from several approaches in the general sequential state-estimation literature and it shares some commonalities with models based around switching state space models (switching-SSMs) (Ghahramani & Hinton, 2000). As with our work, these generative models employ a set of switches that select between distinct state-transition functions, as well as having switch-dependent emission distributions. Switching-SSMs typically couple the switch states through time in a directed Markov chain; although we currently do not do this, it would be a computationally feasible extension to our model. In relation to switching-SSMs, our model is able to leverage the usual advantages of discriminative training in conditional models — namely that the switch variable can be set up to have a rather rich dependency on the observation sequence without incurring extra difficulties with inference or tractability. Our model differs from switching-SSMs in another important way: SSMs fit the joint probability of state and observations, whereas our model disregards the observation density, instead fitting only the conditional probability of state given observations.

With respect to the particular application to object tracking, a number of models employing CRF-style approaches have recently been suggested, including (Sminchisescu et al., 2005; Taycher et al., 2005). While our approach shares some of the same modelling philosophies as these approaches, including employing a variety of features, discriminative training, and dynamic models, the overall form and components of our approach is significantly different. Also from the broader tracking literature there are several approaches that have some aspects in common with our work. In particular, Collins et al. (2005) shares the notion that it is advantageous to have a candidate pool of mechanisms to estimate the object position, and to swap these in and out based on their local performance and consistency. However, this model has no explicit representation of dynamics, and is restricted to simpler features than those in our framework. In a similar vein, Forsyth and Ponce (2002) suggests an approach that employs a Kalman filter in conjunction with "gated" observations. Lastly, we note that Isard and Blake (1998) use particle filtering and a switching dynamics model to follow a simple bouncing ball, which influenced our choice of an illustrative experiment using a bouncing ball in a more realistic setting.

## 5. A visual tracking application

As a test of our model, we apply it to the problem of tracking the position of a basketball in video. Here, we show that by combining several different simple (and often unreliable) observation and dynamics features, we can obtain a reliable tracker.

In this setting the observations are a sequence of grayscale images. For the state at time $t$ we use a 6-dimensional vector encoding the position, velocity, and acceleration of the ball. Augmenting the state-space with velocity and acceleration is a standard transformation (Forsyth & Ponce, 2002), allowing higher-order dynamics to be modeled using features that only look at pairs of temporally adjacent states. Training data consists of a sequence of images, as well as the ground truth locations of the target object (with velocity and acceleration computed via finite differencing).

In our tracker, we include eight different observation features. The first six are based on small template images. Given an observation image, each template is compared (efficiently, using convolution) to all possible sub-patches in the image, and $\gamma_k(\mathbf{Y}, t)$ returns the location of the most-similar patch based on sum-of-squares distance. The next feature uses a 3-component principal components analysis (PCA) subspace. Again, the subspace is applied to all areas of the current image, and $\gamma_k()$ returns the location of the image patch with lowest sum-of-squares reconstruction error. The final feature is based on temporally-local background subtraction. It takes five observation images (the current, two proceeding, and two following images), computes the mean image, and returns the point in the current image that differs most from this mean (after Gaussian blur of the difference image). As expected, this feature can work well when there is only one rapidly-moving object, but can be very unreliable when there is any other motion (including camera motion) in the image. None of these features are able to estimate the velocity or acceleration, thus for these

dimensions of the state the $\phi_k()$'s always predict zero. Note that in our model it is perfectly acceptable for a feature to consider only a subset of the state dimensions.

For each of these observation features we include side-information to help determine the values of the switches. For the template and PCA features, we compute the sum-of-squares error of the best-matching image patch, and for the background subtraction feature, we compute the maximum squared difference between the current and mean images. Each switch potential $\mathcal{G}_k$ takes a (different) linear combination of these values and returns the result. Thus each $\mathcal{G}_k$ can be thought of as a logistic-regression classifier, attempting to determine the observation switches $v_{kt}$ using only information from the observation images.

Four dynamics features are included, each using a linear predictor of $\mathbf{x}_t$ from $\mathbf{x}_{t-1}$. We do not include any side-information for the dynamics, thus $\mathcal{F}_j$ is simply a constant bias.

### 5.1. Tracking a basketball

The video data for this evaluation consisted of four video sequences, totalling around 8400 frames, for which the ground truth location of the basketball was obtained by hand. We trained two separate trackers to evaluate performance during weak and strong generalization scenarios. In the first experiment we used one sequence (referred to as `Simon`, 1796 frames) containing a single player bouncing, throwing and dribbling a basketball. The tracker was trained using the first 500 frames (as well as the corresponding ground truth points) and tested on the remaining frames. In the second experiment we used three sequences in which players pass a basketball by rolling (`roll`, 1556 frames), bouncing (`bounce`, 1897 frames), and by both rolling and bouncing (`roll+bounce`, 3126 frames). Here the tracker was trained on the first 500 frames of `roll` and `bounce`, and tested on the held-out `roll` and `bounce` frames (weak generalization), as well as on the previously-unseen `roll+bounce` sequence (strong generalization).

To train the template features, we extracted $19 \times 19$-pixel image patches of the basketball from the training images. The first five templates were obtained by running K-means clustering on the patches, while the sixth was simply one of the training patches (we chose the last image). The PCA model was also fit using these training patches. The linear parameters of the switch potentials $\mathcal{G}_k$ were fit using logistic regression. Although they can often correctly locate the basketball, none of the features is always "on the ball". The reliability of each observation feature (the frequency with which it predicts a location within 5 pixels of the basketball), and the frequency with which our tracker switches it on at test time are given in Table 2. The most reliable feature is PCA (0.81), and the "background subtraction" feature (0.08) is the least.

*Table 2.* Fraction of time that each observation feature correctly locates the ball in the `Simon` sequence: the ground truth, and as estimated by our tracker.

| Feature | Locates the ball | |
|---|---|---|
| | True | Estimated |
| K-means 1 | 0.34 | 0.36 |
| K-means 2 | 0.53 | 0.53 |
| K-means 3 | 0.61 | 0.61 |
| K-means 4 | 0.63 | 0.60 |
| K-means 5 | 0.63 | 0.63 |
| Last training patch | 0.33 | 0.34 |
| PCA | 0.81 | 0.81 |
| Background Sub. | 0.08 | 0.83 |

To train the dynamics features ($\mathbf{T}_j$ and $\mathbf{d}_j$), in the first experiment we hand-segmented the ground-truth states from the training data into four regimes: *flying* (the basketball in free-flight), *holding* (the basketball in the hands of the player), *bouncing off the ground*, and *bouncing off the wall*. The parameters of each $\phi_j()$ were chosen to minimize $\|\mathbf{x}_t - (\mathbf{T}_j\mathbf{x}_{t-1} + \mathbf{d}_j)\|$ for the set of corresponding ground-truth states. Segmenting the data can be time consuming, so in the second experiment we used manually-chosen dynamics features, corresponding to flight, rolling, bouncing, and holding.

Finally, the feature precisions $\boldsymbol{\alpha}_j$ and $\boldsymbol{\beta}_k$, as well as the logistic-regression parameters, were refined using 300 iterations of Contrastive Divergence learning. Given the learned model, we track the basketball by applying 20 iterations of the inference method described in Section 3, producing an estimate of the state sequence and the switches.

### 5.2. Results

To quantitatively assess the trackers' performance, for each of the test sequences we computed an error rate defined to be the fraction of frames in which the predicted state was more than 5 pixels away from the true location of the basketball. As a baseline, we also attempted to track the sequences using an SSM, specifically a Kalman filter fit to the training data. We experimented with including different subsets of the observations features , as well as including/not includ-

ing velocity and acceleration in the state, but in all cases the Kalman filter performed very poorly. For further comparison, we applied the incremental visual tracker (IVT) recently proposed by Lim et al. (). IVT uses particle-filter dynamics with a PCA likelihood, and has demonstrated good performance.

The result of tracking the `Simon` sequence is shown in Figure 2. As can be seen the basketball is tracked well throughout the sequence. Although the tracker loses the ball briefly on four occasions, it quickly recovers. The error rate of the tracker was 0.1196, versus 0.7229 for the Kalman filter. The IVT was able to track the first part of the sequence without error, but it lost track of the ball after 688 frames and was unable to recover, resulting in an error rate of 0.613.

The results of the second experiment are shown in Figure 3. The error rates for the tracker were 0.0208 on `roll`, 0.0766 on `bounce`, and 0.1004 on `roll+bounce`. The corresponding rates for the Kalman filter were 0.8333, 0.9434, and 0.9697. The IVT tracked the first 700 frames of `roll`, 125 frames of `bounce`, and 730 frames of `roll+bounce` before failing, giving error rates of 0.37, 0.919, and 0.767. The discrepancy in error rates between the three trackers highlights the difficulty in choosing an appropriate metric for quantitatively comparing tracking results. However, we feel that fraction of time "on-the-ball" seems the most appropriate measure for this application.

### 5.3. Dealing with missing observations

To test our model's ability to deal with missing observations, and the quality of the learned dynamics features, we modified the above sequence so that all observation features would be off for 20 consecutive frames, while the ball is in free flight. Occlusion is a notoriously difficult problem for tracking, as state-of-the-art trackers perform simple diffusion until telltale observations enable the tracker to locate the object. (Jepson et al., 2001)

The result can be seen in Figure 2 (lower-right image). The model is able to successfully track through the 20-frame (98-pixel displacement) simulated occlusion. Note that the uncertainty in the state estimate (indicated by blue circles of one standard deviation) grows during the missing observations, peaking at the middle of the occlusion.

## 6. Discussion

We have presented a novel framework for inferring complex trajectories from high-dimensional and noisy data. One of the key advantages of our approach is that we have complete flexibility about the observation and dynamics features that we use in our model. The discriminative learning procedure can appropriately weight the confidence of different predictors, as well as integrating these predictions over time with a versatile dynamics model, and learning to effectively gate in and out different features based on their inferred accuracy and relevance. Although the features used in this paper are relatively simple, we still obtain impressive results. One exciting prospect is the possibility of using rather more powerful predictors in combination — for instance one could use as observation features the outputs from state-of-the-art object detectors, or even *other trackers*. We could also use information from multiple frames (for example estimated optical flow) to help make predictions.

Our model can also readily be extended by improving the side information that constrains the inference of which features to use. Our current model does not utilize side information to help select the dynamics switches; some interactions could be used here to improve performance. Also, the switches are conditionally independent at each time-step. Whilst it might not be practical to couple all the observation switches over time, it seems feasible to take the multinomial switches controlling the dynamics and couple them in a linear Markov chain. Inference (for smoothing) would then consist of a forwards-backwards pass of belief propagation for both the continuous state *and* the dynamics switches, whilst the observation switches would remain conditionally independent given the continuous state.

One of the main drawbacks of our approach is that we require a fully labelled sequence to train on, and in certain applications such data (and in sufficient quantities to constrain all the parameters of the model without encountering problems with overfitting) may be hard to come by. A simple way around this would be to bootstrap from other tracking methods, using them to provide an initial labelling of sequences for training and using human intervention only on the more challenging segments.

In addition to incorporating more powerful features, there are a number of other interesting directions in which this work may be taken. One area that we have started to explore is the performance of different inference algorithms. The dependency structure of our model allows for a relatively efficient MCMC procedure for obtaining samples from the conditional posterior. Other (approximate) inference methods such as variational approximations, particle filtering or nonparametric belief propagation may also be of use in

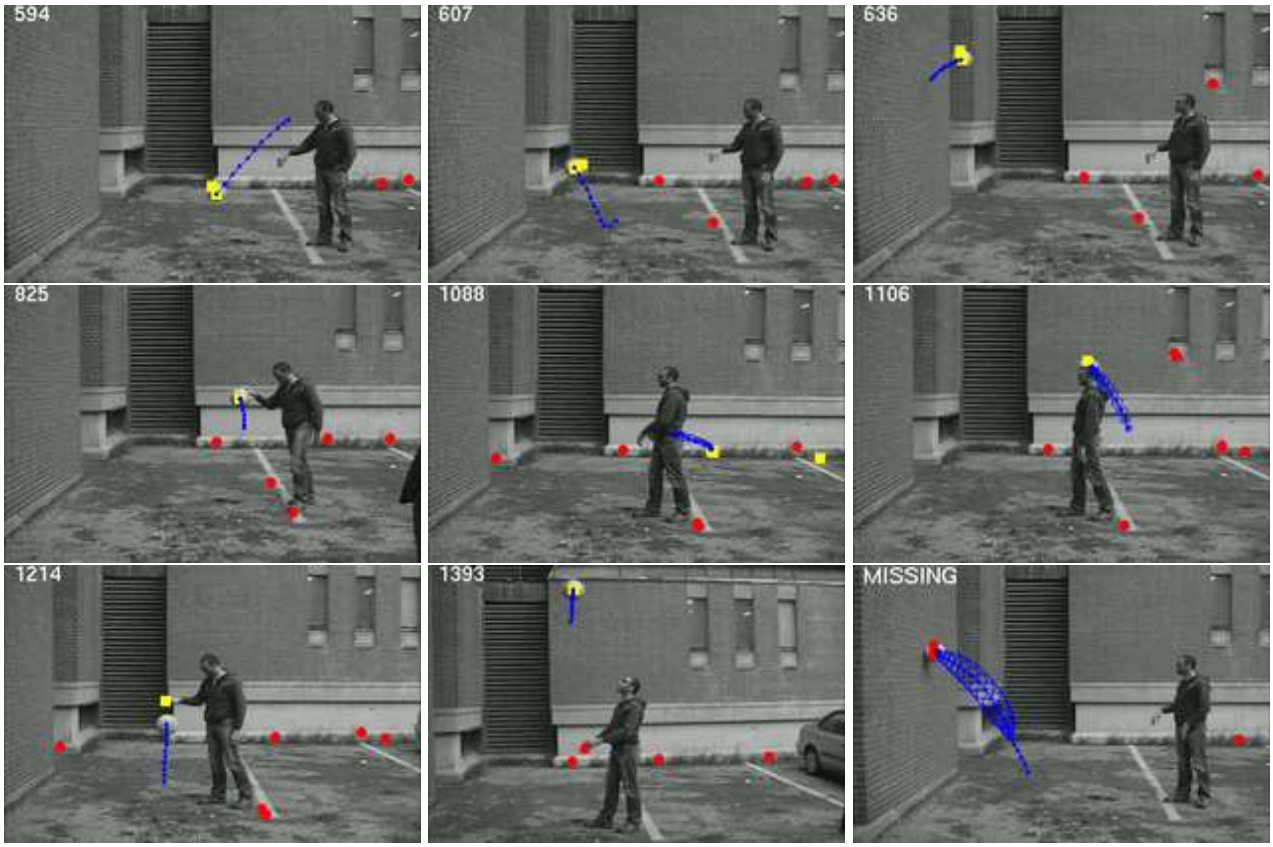


*Figure 2.* Tracking a basketball (`Simon` sequence). The full video is available at http://www.cs.toronto.edu/~dross/cdf_icml06/. The location of the ball, as predicted by the model, is given in blue. The locations predicted by the eight observations are drawn as yellow boxes (if the corresponding switch is 1) or as red circles (if the switch is 0). The ball is successfully tracked during fast motion (frame 594), when bouncing off the ground (607) and the wall (636), and when dribbled by the player (825). The model does lose track of the ball when it is occluded (1088), but quickly recovers when the ball becomes visible again (1106). The dynamics allow the ball to be tracked even when all observation features' predictions are erroneous (and one bad feature is on!) (1214). The model can also cope with motion of the camera (1393). (MISSING DATA: lower right) The basketball is successfully tracked through a 20-frame (98 pixel displacement) simulated occlusion.

our model, particularly for real-time or online/filtering applications.

Another interesting direction for this work might be the inclusion of multi-modal observations and multi-sensory fusion for inference of state. The features we use need not entirely constrain the state, and one of the strengths of our model is that it is able to combine many weak predictors to give a single good estimate of state. As a example relative to the experiments presented here, one could imagine a simple auditory cue as being rather useful in inferring the occurrence of a bounce — potentially allowing us to deal with changes in behaviour even when the ball is visually occluded.

Moving beyond the scope of tracking applications considered here, the general ideas behind our model seem to hold promise for a wide range of applications — examples under consideration include articulated body recovery, prediction of financial time series, and flexible combination of stereoscopic depth predictions.

## References

Collins, R., Liu, Y., & Leordeanu, M. (2005). On-line selection of discriminative tracking features. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, *27*, 1631 – 1643.

Forsyth, D. A., & Ponce, J. (2002). *Computer vision: A modern approach.* Prentice Hall.

Ghahramani, Z., & Hinton, G. E. (2000). Variational

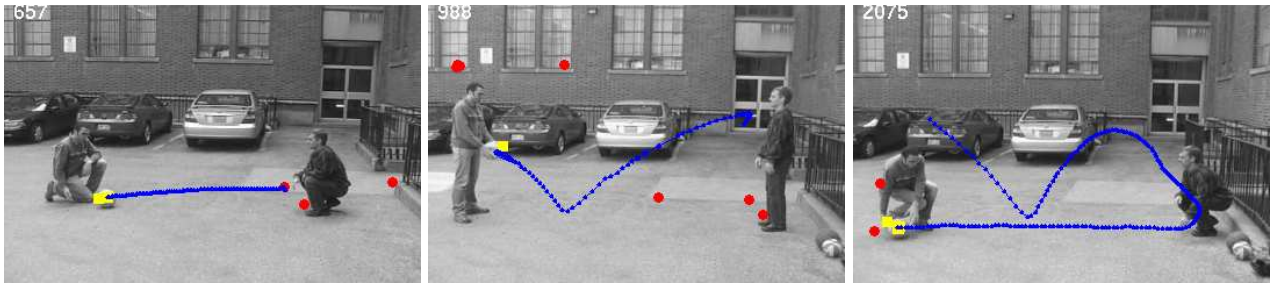


*Figure 3.* Results on the `roll`, `bounce`, and `roll+bounce` sequences. The full videos are available at http://www.cs.toronto.edu/~dross/cdf_icml06/.

learning for switching state-space models. *Neural Computation*, *12*, 831–864.

Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. *Neural Computation*, *14*, 1771–1800.

Isard, M., & Blake, A. (1998). A mixed-state Condensation tracker with automatic model-switching. *Proceedings of the International Conference on Computer Vision (ICCV)*.

Jepson, A. D., Fleet, D. J., & El-Maraghi, T. F. (2001). Robust online appearance models for visual tracking. *CVPR* (pp. 415–422).

Lafferty, J., McCallum, A., & Pereira, F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *International Conference on Machine Learning (ICML)*.

Lim, J., Ross, D. A., Lin, R.-S., & Yang, M.-H. Incremental learning for visual tracking. In L. K. Saul, Y. Weiss and L. Bottou (Eds.), *Nips 17*.

Quattoni, A., Collins, M., & Darrell, T. Conditional random fields for object recognition. In L. K. Saul, Y. Weiss and L. Bottou (Eds.), *Nips 17*.

Roth, S., & Black, M. J. (2005). Fields of experts: A framework for learning image priors. *CVPR*.

Sminchisescu, C., Kanaujia, A., Li, Z., & Metaxas, D. (2005). Discrimintative density propagation for 3d human motion estimation. *CVPR*.

Sudderth, E. B., Mandel, M. I., Freeman, W. T., & Willsky, A. S. Distributed occlusion reasoning for tracking with nonparametric belief propagation. In L. K. Saul, Y. Weiss and L. Bottou (Eds.), *Nips 17*.

Taycher, L., Shakhnarovich, G., Demirdjian, D., & Darrell, T. (2005). *Condition random people: Tracking humans wth crfs and grid filters* (Technical Report 2005-079). MIT-CSAIL.

Teh, Y., Welling, M., Osindero, S., & Hinton, G. E. (2003). Energy-based models for sparse overcomplete representations. *Journal of Machine Learning Research*, *4*, 1235–1260.

Welling, M., Rosen-Zvi, M., & Hinton, G. Exponential family harmoniums with an application to information retrieval. In L. K. Saul, Y. Weiss and L. Bottou (Eds.), *Nips 17*.