People Tracking Using Hybrid Monte Carlo Filtering

Kiam Choo[*†] **David J. Fleet**[†]

[†]Xerox Palo Alto Research Center, 3333 Coyote Hill Rd, Palo Alto, CA 94304
[*]Department of Computer Science, University of Toronto, Toronto, M5S 1A4

Abstract

Particle filters are used for hidden state estimation with nonlinear dynamical systems. The inference of 3-d human motion is a natural application, given the nonlinear dynamics of the body and the nonlinear relation between states and image observations. However, the application of particle filters has been limited to cases where the number of state variables is relatively small, because the number of samples needed with high dimensional problems can be prohibitive. We describe a filter that uses hybrid Monte Carlo (HMC) to obtain samples in high dimensional spaces. It uses multiple Markov chains that use posterior gradients to rapidly explore the state space, yielding fair samples from the posterior. We find that the HMC filter is several thousand times faster than a conventional particle filter on a 28D people tracking problem.

1 Introduction

Particle filters [8, 11, 13, 17] have become a popular way to infer time-varying properties of a scene from images. Applications include tracking rigid and articulated objects [13, 14, 18], gesture recognition [1], robot localization [10], and estimating occlusion boundaries [2]. Particle filters compute a sampled representation of the posterior probability distribution over scene properties of interest, conditioned on image observations. The interpretation of 3-d human motion from video is one particularly compelling application owing to the ease with which particle filters cope with nonlinear dynamics and nonlinear observation equations, and the ease with which they maintain uncertainty with multimodal distributions [23].

Nevertheless, the successful application of particle filters has been limited to situations where the number of state variables is relatively small. For high dimensional state spaces the algorithm can become computationally inefficient and thus ineffective. This paper describes a modified particle filter that uses a Markov chain Monte Carlo (MCMC) technique called hybrid Monte Carlo (HMC) [9, 19, 21] to filter more efficiently in high dimensional state spaces. Like the particle filter, it uses a number of particles. But rather than weighting each particle by its likelihood, each particle produces a Markov chain that can follow the gradient of the posterior over large distances. This allows it to rapidly explore the state space, while producing fair samples from the desired posterior distribution. We apply the HMC filter to the problem of estimating the time-varying, 3D shape of a moving person from a sequence of projected marker positions in a 2D image. On this 28D problem, we find that the HMC filter is several thousand times faster than a conventional particle filter.

2 Background and Previous Work

The goal of Bayesian filtering is to compute the posterior probability distribution $p(\mathbf{s}_t \mid \mathbf{z}_{1:t})$ over an unknown state $\mathbf{s}_t$ at time t, conditioned on image observations, $\mathbf{z}_{1:t} \equiv (\mathbf{z}_1, \ldots, \mathbf{z}_t)$, up to time t. Particle filters work by approximating the posterior distribution using a discrete set of samples (i.e., states), where each sample corresponds to some hypothesized set of model parameters. Each sample state is typically weighted by its likelihood, $p(\mathbf{z}_t \mid \mathbf{s}_t)$, the probability that the current observations were generated by the hypothesized state. With high dimensional state spaces, a major concern is the number of samples, and hence the number of likelihood computations that are required to adequately approximate the posterior.

One way to minimize the required computational effort is to reduce the effective size of the state space that must be searched. One can do this by exploiting problem-dependent constraints, such as conditional independence among the state variables. MacCormick and Isard [18] partition the state space and sample the partitioned variables in sequence, using importance reweighting.

One can also reduce the number of particles by choosing a better proposal distribution, for example by improving the dynamical model of the system or by finding a low-dimensional subspace in which the tracking can be performed [15, 23]. This is appropriate when such low-dimensional representations are available. Other ways to obtain better proposals involve importance sampling or sampling from low-level detectors in order to rapidly inject good hypotheses into the sample set [2, 14].

Deutscher et al. [7] and Cham and Rehg [4] tackle the problem of tracking people in high dimensional spaces by following gradients to good hypotheses. Although such

methods produce maximal-likelihood parameter estimates, they do not produce an approximation to the desired posterior. Even with multiple hypotheses [4], the samples are not likely to be properly weighted samples from the posterior.

The hybrid Monte Carlo filter proposed here can be seen as a way of following gradients of the posterior distribution to good hypotheses, but it is designed to generate samples that are correctly distributed according to the posterior. When properly tuned, it allows for long trajectories through state space so that the posterior can be sampled rapidly.

3 Basic Particle Filter

The particle filter is described elsewhere [11, 13, 17], but we include it here for completeness. If we model the time-varying, d-dimensional state $\mathbf{s}_t$ as a first-order Markov process, and assume that observations $\mathbf{z}_t$ are independent given $\mathbf{s}_t$, then we can factor the posterior $p(\mathbf{s}_t|\mathbf{z}_{1:t})$ to obtain

$$p(\mathbf{s}_t|\mathbf{z}_{1:t}) \;=\; \kappa\, p(\mathbf{z}_t|\mathbf{s}_t) \int p(\mathbf{s}_t|\mathbf{s}_{t-1})\, p(\mathbf{s}_{t-1}|\mathbf{z}_{1:t-1}) d\mathbf{s}_{t-1}\;, \quad (1)$$

where κ is a constant that is independent of $\mathbf{s}_t$, $p(\mathbf{z}_t|\mathbf{s}_t)$ is the likelihood function, and the integral is often referred to as the temporal prior over $\mathbf{s}_t$ given past observations.

The particle filter uses the temporal prior to bound the search for high likelihood states by drawing proposal states directly from it. Importance weights are then used to properly weight these states so they represent the posterior rather than the prior from which they were drawn. This yields a weighted set of samples, $\mathcal{S}_t = \{\mathbf{s}_t^i, w_t^i\}_{i=1}^N$, which is said to be properly weighted when summation over the particles approximates expectation under the posterior $\mathcal{P}_t$ [17]; i.e.,

$$\mathrm{E}_{\mathcal{S}_t}[f(\mathbf{s}_t)] \;\equiv\; \sum_{i=1}^{N} w_t^i f(\mathbf{s}_t^i) \;\stackrel{N\to\infty}{\longrightarrow}\; \mathrm{E}_{\mathcal{P}_t}[f(\mathbf{s}_t)]\;, \quad (2)$$

for sufficiently smooth functions f.

Beginning with a sample set $\{\mathbf{s}_{t-1}^i, w_{t-1}^i\}_{i=1}^N$ that approximates $p(\mathbf{s}_{t-1}|\mathbf{z}_{1:t-1})$, the computation at time t is:

1. Treating the weights as probabilities, draw N particles with replacement from $\{\mathbf{s}_{t-1}^i\}_{i=1}^N$ to obtain a uniformly weighted sample set $\{\mathbf{u}_{t-1}^i\}_{i=1}^N$.
2. For each particle $\mathbf{u}_{t-1}^i$ draw a sample from transition density (the model dynamics) $p(\mathbf{s}_t \,|\, \mathbf{s}_{t-1} \;=\; \mathbf{u}_{t-1}^i)$, yielding a proposal set $\{\mathbf{s}_t^i\}_{i=1}^N$.
3. Compute the importance weights w_t^i from the normalized likelihoods, i.e., $w_t^i \;=\; c\, p(\mathbf{z}_t \,|\, \mathbf{s}_t \;=\; \mathbf{s}_t^i)$ where $c^{-1} = \sum_j p(\mathbf{z}_t \,|\, \mathbf{s}_t = \mathbf{s}_t^j)$ is the normalization constant.

The success of a particle filter depends on its ability to maintain an good approximation to the posterior. Following (2), one natural way to assess the quality of the filter is to examine the variability of sample expectations, $\mathrm{E}_{\mathcal{S}_t}[f(\mathbf{s}_t)]$, as compared to expectations under $\mathcal{P}_t$, that is:

$$\mathrm{Var}\left[\mathrm{E}_{\mathcal{S}_t}[f(\mathbf{s}_t)]\right] \;\equiv\; \mathrm{E}\left[\left(\mathrm{E}_{\mathcal{S}_t}[f(\mathbf{s}_t)] - \mathrm{E}_{\mathcal{P}_t}[f(\mathbf{s}_t)]\right)^2\right], \quad (3)$$

where expectation is taken over different runs of the particle filter. Not surprisingly, small variances are preferred.

The variance in (3) depends on the sample set. If one had N independent samples, $\{\mathbf{v}_t^i\}_{i=1}^N$, from the posterior $\mathcal{P}_t$, then the sample mean and its variance are

$$\mathrm{E}_{\mathcal{V}_t}[f(\mathbf{s}_t)] \;=\; \frac{1}{N}\sum_{i=1}^{N} f(\mathbf{v}_t^i) \quad (4)$$

$$\mathrm{Var}[\mathrm{E}_{\mathcal{V}_t}[f(\mathbf{s}_t)]] \;=\; \frac{1}{N}\mathrm{Var}_{\mathcal{P}_t}[f(\mathbf{s}_t)]\;. \quad (5)$$

The variance in (3) will generally be larger than that in (5). It is sometimes approximated by

$$\mathrm{Var}\left[\mathrm{E}_{\mathcal{S}_t}[f(\mathbf{s}_t)]\right] \;\approx\; \left(\sum_{i=1}^{N}(w_t^i)^2\right)\mathrm{Var}_{\mathcal{P}_t}\left[f(\mathbf{s}_t)\right]\;, \quad (6)$$

where $1/\sum(w_t^i)^2$ is called the *effective number of samples* [3, 16, 18].

In practice, random variability can be problematic for particle filters. Some regions of state space may receive fewer than the expected number of samples, while samples in other regions may occur closer together than necessary given the smoothness of the posterior. One can get a rough estimate of the minimum number of particles required by comparing the effective volumes (variances) of the search space (the temporal prior density) and the target (the posterior). For example, if the prior and posterior be constant in d-dimensional hyperellipses with radii $\{\Lambda_j\}_{j=1}^d$ and $\{\lambda_j\}_{j=1}^d$, and zero elsewhere, then a simple calculation shows that

$$\mathrm{Var}[\mathrm{E}_{\mathcal{S}_t}[f(\mathbf{s}_t)]] \;\approx\; \frac{(\Lambda/\lambda)^d}{N}\,\mathrm{Var}_{\mathcal{P}_t}\left[f(\mathbf{s}_t)\right]\;, \quad (7)$$

where $\Lambda = \prod \Lambda_j^{1/d}$ and $\lambda = \prod \lambda_j^{1/d}$. This implies that the minimum number of particles, approximately $N(\lambda/\Lambda)^d$, is exponential in the state dimension. Random variability will lead to poor tracking when d is high, or when $\lambda \ll \Lambda$.

To cope with sampling inefficiency in high dimensional spaces, we propose the use of a MCMC method called hybrid Monte Carlo (HMC) for filtering [9, 21]. After reviewing the relevant aspects of MCMC methods, we discuss the HMC filter below.

4 Markov Chain Monte Carlo

Central to MCMC methods is the Markov chain (MC), a sequence of random variables X_0, X_1, ..., that satisfy $p(X_i|X_{i-1}, X_{i-2}, ..., X_0) = p(X_i|X_{i-1})$. For filtering, the utility of Markov chains lies in the fact that, by choosing a suitable transition distribution $p(X_i|X_{i-1})$, a Markov chain can be made to converge to the posterior, $\mathcal{P}$. That is, regardless of starting state, as the number of samples in the MC grows sufficiently large, they become fair samples from $\mathcal{P}$. For this to happen, the transition $p(X_i|X_{i-1})$ must leave

$\mathcal{P}$ invariant; i.e., if X_{i-1} is a fair sample from $\mathcal{P}$, then X_i drawn from $p(X_i|X_{i-1})$ must also be a fair sample from $\mathcal{P}$. MCMC methods often use Metropolis tests to achieve invariance through detailed balance [21]. The transition must also be irreducible and aperiodic in order to ensure convergence [22].

In this paper, we use an MCMC technique to obtain samples $\mathcal{M}_t = \{\mathbf{s}_t^i\}_{i=1}^N$ from the posterior $\mathcal{P}_t = p(\mathbf{s}_t|\mathbf{z}_{1:t})$. Expectations of functions $f(\mathbf{s}_t)$ under $\mathcal{P}_t$ are then computed by averaging as in (4). However, because samples from a Markov chain are not independent in general, the variances of such statistics converge more slowly than $1/N$:

$$\mathrm{V}ar[\mathrm{E}_{\mathcal{M}_t}[f(\mathbf{s}_t)]] \;=\; \frac{\tau}{N}\mathrm{V}ar_{\mathcal{P}_t}[f(\mathbf{s}_t)]\;, \tag{8}$$

Here, the inefficiency factor, τ, depends on the autocorrelation of the MC samples. Comparing (8) to (7), we see that the particle filter, in effect, has an inefficiency factor that rises exponentially with dimensionality.

5 Hybrid Monte Carlo (HMC) Filter

The particle filter outlined in Sec. 3 produces weighted samples. In contrast, the algorithm we propose uses M Markov chains, each with $R+1$ samples; let $(\mathbf{s}_t^{i,0}, \mathbf{s}_t^{i,1}, ..., \mathbf{s}_t^{i,R})$ denote the samples from the i^{th} Markov chain. The HMC filter is like a particle filter as it begins each time step with M particles. But each particle then spawns a MC that converges to the posterior. Although a single MC will eventually explore the entire state space, it often requires many samples to move between different modes of the posterior. We use multiple, independent chains to explore multiple modes more efficiently.

The algorithm begins with a temporal prior. Like the particle filter, we take this to be a linear mixture of transition densities, conditioned on samples from the posterior at the previous time. Here we use only the final sample from each MC, $\{\mathbf{s}_{t-1}^{i,R}\}_{i=1}^M$, from which we obtain

$$p(\mathbf{s}_t \,|\, \mathbf{z}_{1:t-1}) \;=\; \frac{1}{M}\sum_{i=1}^{M} p(\mathbf{s}_t \,|\, \mathbf{s}_{t-1}{=}\mathbf{s}_{t-1}^{i,R})\;. \tag{9}$$

Combining this with the likelihood function yields the posterior at time *t*:

$$p(\mathbf{s}_t|\mathbf{z}_{1:t}) \;=\; \kappa\, p(\mathbf{z}_t|\mathbf{s}_t)\, p(\mathbf{s}_t \,|\, \mathbf{z}_{1:t-1})\;. \tag{10}$$

The HMC filter then proceeds as follows:

1. For each particle in $\{\mathbf{s}_{t-1}^{i,R}\}_{i=1}^M$, draw proposal states $\mathbf{u}_t^i \sim p(\mathbf{s}_t \,|\, \mathbf{s}_{t-1} = \mathbf{s}_{t-1}^{i,R})$, and evaluate the weights $w_t^i = c\, p(\mathbf{z}_t \,|\, \mathbf{s}_t{=}\mathbf{u}_t^i)$, where $c^{-1} = \sum_i p(\mathbf{z}_t|\mathbf{s}_t{=}\mathbf{u}_t^i)$.
2. Treating the weights as probabilities, draw M particles with replacement from the proposal set $\{\mathbf{u}_t^i\}_{i=1}^M$ to obtain $\{\mathbf{s}_t^{i,0}\}_{i=1}^M$, the initial Markov chain states.
3. Set the unnormalized target posterior as $\mathcal{P}_t = p(\mathbf{z}_t|\mathbf{s}_t)\, p(\mathbf{s}_t \,|\, \mathbf{z}_{1:t-1})$. (The normalization constant is not significant for the hybrid Monte Carlo updates.)
4. For each particle, use HMC updates to compute its MC states $\mathbf{s}_t^{i,r} = U(\mathbf{s}_t^{i,r-1};\, \mathcal{P}_t)$ for $r = 1, ..., R$. The transition $U(\mathbf{s}_t^{i,r-1};\, \mathcal{P}_t)$ is designed to leave $\mathcal{P}_t$ invariant.

In practice, MCMC methods require some *burn-in* time before the samples reach equilibrium; only those samples that are drawn after the chain has reached equilibrium are representative of the posterior. It is therefore common to discard the early samples of each chain. The initial MC states computed in step (2) are drawn from a rough approximation to $\mathcal{P}_t$ in order to minimize the number of burn-in samples. However, as there is no universally good way to determine when one reaches equilibrium [6], we currently set by hand the number of burn-in samples to be the same across all chains.

Each particle is updated to keep the target posterior $\mathcal{P}_t$ invariant. As explained below, the hybrid Monte Carlo algorithm requires that we are able to evaluate both the density and the gradient of the target distribution. In practice, we use Gaussian distributions for the transition density, so this is not a significant restriction on the prior. Many likelihood functions used in vision are also differentiable, so this is not a severe restriction either.

6 How Hybrid Monte Carlo Works

To produce samples from the target posterior distribution $\mathcal{P}(\mathbf{s})$, hybrid Monte Carlo performs a physical simulation of an energy-conserving system with a potential energy bowl equal to $-\log \mathcal{P}(\mathbf{s})$ [9, 21]. The intuition is that if you observe the state of the system at regular intervals, then the collection of observed states forms a Markov chain that comes from $\mathcal{P}$, provided that you replace the system's momentum after every observation by a sample from a unit Gaussian. These momentum resamplings ensure that the system can acquire enough energy to visit unlikely states with nonzero probability.

In the physical simulation, each state variable s_j is paired with a momentum variable p_j. On this extended state space, the Hamiltonian, or total energy, is defined as $H(\mathbf{s}, \mathbf{p}) = E(\mathbf{s}) + K(\mathbf{p})$, where $E(\mathbf{s}) = -\log \mathcal{P}(\mathbf{s})$ is the potential energy and $K(\mathbf{p}) = \frac{1}{2}\mathbf{p}^T\mathbf{p}$ is the kinetic energy for a system with a unit mass matrix. The new target distribution is

$$\mathcal{P}'(\mathbf{s}, \mathbf{p}) = C \exp(-H(\mathbf{s}, \mathbf{p})) \tag{11}$$

where C is a normalizing constant. By construction $\mathcal{P}'$ is separable, so the marginal distribution of $\mathbf{s}$ under $\mathcal{P}'$ is simply the desired posterior $\mathcal{P}$. Thus, if we can get a sample $(\mathbf{s}, \mathbf{p})$ from $\mathcal{P}'$, then $\mathbf{s}$ is also a fair sample from $\mathcal{P}$.

Hybrid Monte Carlo produces MC samples with a transition $(\mathbf{s}^r, \mathbf{p}^r) \rightarrow (\mathbf{s}^{r+1}, \mathbf{p}^{r+1})$ that leaves $\mathcal{P}'$ invariant. (In

what follows, we drop the superscript i and the subscript t with the understanding the discussion applies to chain i at time t.) The HMC transition is composed of two steps, each of which leaves $\mathcal{P}'$ invariant. First, $\mathbf{p}^r$ is replaced by $\tilde{\mathbf{p}}^r$, which is sampled from a unit Gaussian. This leaves $\mathcal{P}'$ invariant as $\mathbf{p}$ is independent of $\mathbf{s}$, and we have not changed $\mathbf{p}$'s distribution.

The second step, $(\mathbf{s}^r, \tilde{\mathbf{p}}^r) \rightarrow (\mathbf{s}^{r+1}, \mathbf{p}^{r+1})$, involves the physical simulation. Starting from $(\mathbf{s}^r, \tilde{\mathbf{p}}^r)$, the system evolves according to Hamiltonian dynamics:

$$\frac{d\mathbf{p}}{dt} = -\nabla E(\mathbf{s}) \ , \quad \frac{d\mathbf{s}}{dt} = (p_1, \, p_2, \, ..., \, p_d) \ . \tag{12}$$

Because Hamiltonian dynamics conserves H, is reversible, and preserves the phase space volume, it leaves $\mathcal{P}'$ invariant [21]. In practice, we use a discretized simulation made up of L deterministic leapfrog steps, called a leapfrog trajectory. With the initial state $(\hat{\mathbf{s}}^0, \hat{\mathbf{p}}^0) \equiv (\mathbf{s}^r, \tilde{\mathbf{p}}^r)$, the l^{th} leapfrog step, with stepsize ϵ, is specified component-wise by

$$\hat{p}_j^{\,l-1/2} = \hat{p}_j^{\,l-1} - \frac{\epsilon}{2} \frac{\partial E(\hat{\mathbf{s}}^{l-1})}{\partial s_j} \tag{13}$$

$$\hat{s}_j^{\,l} = \hat{s}_j^{\,l-1} + \epsilon \, \hat{p}_j^{\,l-1/2} \tag{14}$$

$$\hat{p}_j^{\,l} = \hat{p}_j^{\,l-1/2} - \frac{\epsilon}{2} \frac{\partial E(\hat{\mathbf{s}}^{l})}{\partial s_j} \tag{15}$$

where each line is computed for $j = 1...d$ before moving on to the next.

Because the simulation is discrete, it is not guaranteed to conserve H, nor to leave $\mathcal{P}'$ invariant as a consequence. We therefore perform a Metropolis rejection test [20] at the end of each leapfrog trajectory. That is, we accept the proposal $(\hat{\mathbf{s}}^L, \hat{\mathbf{p}}^L)$ with probability

$$\min\{1, \exp[-H(\hat{\mathbf{s}}^L, \hat{\mathbf{p}}^L) + H(\mathbf{s}^r, \tilde{\mathbf{p}}^r)]\} \ . \tag{16}$$

If the proposal is accepted, we set $(\mathbf{s}^{r+1}, \mathbf{p}^{r+1})$ to $(\hat{\mathbf{s}}^L, \hat{\mathbf{p}}^L)$. If rejected, $(\mathbf{s}^{r+1}, \mathbf{p}^{r+1})$ is set to $(\mathbf{s}^r, \tilde{\mathbf{p}}^r)$. High rejection rates should be avoided as they usually lead to inefficient exploration of the state space.

The Metropolis rejection test is guaranteed to yield a MC update that keeps the target distribution invariant if it is used with deterministic proposals that are self-inverting and have Jacobian 1 [5]. The leapfrog trajectory has both these properties. In principle, other types of proposals can be used with Metropolis tests to obtain samples from $\mathcal{P}'$. The key attraction of the leapfrog physical simulation is that, because it is a simulation of Hamilton's equations, it leaves H roughly constant even for long trajectories so long as ϵ is not so large that the simulation becomes unstable. As can be seen from (16), keeping H roughly constant keeps rejection rates, and thus MC autocorrelations, low. Furthermore, long trajectories avoid random walks, and thereby produce samples from distributions efficiently.

In (13)-(15), the stepsize ϵ is identical for each state component, $j = 1...d$. This is fine for isotropic energy bowls. Otherwise, the stepsize in each direction should ideally scale with the width of the energy bowl in that direction. Thus, one might use a separate stepsize ϵ_j for each state component, $j = 1...d$; following [21], we set the stepsizes as follows:

$$\epsilon_j \approx \epsilon \left(\frac{\partial^2 E}{\partial s_j^2} \right)^{-\frac{1}{2}} \tag{17}$$

where ϵ is called the stepsize adjustment factor. The second derivative of E must be estimated using a heuristic so that it does not depend on $\mathbf{s}$ (with the articulated model below, variables high in the kinematic tree have larger second derivatives than ones lower in the tree).

In general, leapfrog updates are valid for any nonzero setting of ϵ_j so long as their choice does not depend on $\mathbf{s}$. The self-inverting property of the mapping would be violated otherwise. Our formulation here with different stepsizes for each component is still equivalent to a Hamiltonian formulation, but with a diagonal mass matrix instead of the unit mass matrix used above. In particular, the different step sizes given in (17) would be equivalent to a mass matrix with diagonal elements given by ϵ_j^2/ϵ^2.

7 Experiments

To evaluate the hybrid Monte Carlo (HMC) filter, we used it to infer the 3D motion of people walking and dancing, from 2D moving light displays obtained from a commerical motion capture system. Inference of 3D structure from motion capture data is interesting in its own right, but we selected this problem rather than inference from video [23] to simplify the likelihood function, allowing us to focus our analysis on the filtering algorithm. Below we compare the HMC filter to the particle filter on problems of different dimensions, and show that the HMC filter is far superior in high dimensions.

7.1 Observations Used for Tracking

The observations are labeled 2D positions that correspond to 3D joint locations on a human subject. Figure 1A shows the 3D marker locations on the body. The observations are given by the perspective projection of the 3D points onto the image plane plus additive noise. With the camera centered at (X_c, Y_c, Z_c) and its optical axis aligned with the Y-axis, the 2D location of the l^{th} marker located at (X_m^l, Y_m^l, Z_m^l) is given by

$$\mathbf{d}_l = \left(\frac{X_m^l - X_c}{Y_m^l - Y_c}, \frac{Z_m^l - Z_c}{Y_m^l - Y_c} \right) + \mathbf{n} \tag{18}$$

where $\mathbf{n}$ is mean-zero Gaussian noise with variance σ^2.

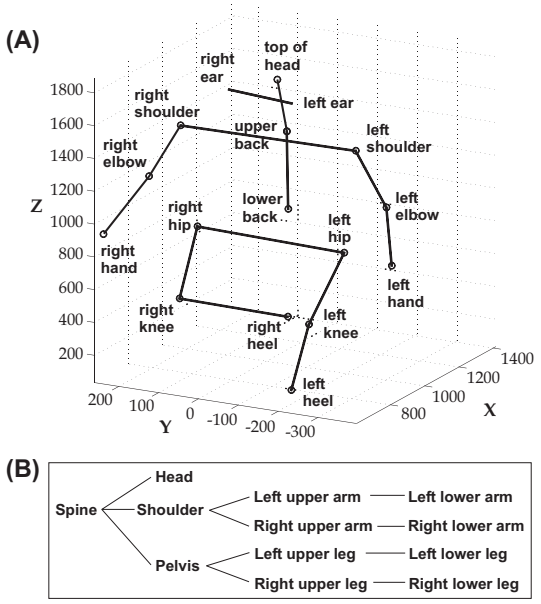

Figure 1. (A) 3D positions of 17 markers on a human subject. (B) The kinematic tree used in our model.

7.2 Likelihood and Temporal Prior

The person is modeled as a hierarchical articulated model. The hierarchy forms a kinematic tree rooted in the spine as shown in Fig. 1B. Each body part has a local reference frame with its z axis running through what is naturally thought of as the longest axis of that part. The pose of a body part's reference frame is specified using 3D rotations and translations that map its frame to its parent's. Angles (α, β, γ) denote angles of rotation about the X, Y and Z axes. These angles, with the translations between part coordinate origins, define the homogeneous transformations, $R_X(\alpha)$, $R_Y(\beta)$ and $R_Z(\gamma)$. Taken in succession, they define the transformation matrix that maps a point $\mathbf{X}_c$ in some child frame to $\mathbf{X}_p$ in its parent's frame:

$$\mathbf{X}_p = R_Z(\gamma) R_Y(\beta) R_X(\alpha)\, \mathbf{X}_c \tag{19}$$

Given a state, **s**, one can traverse the kinematic tree to generate the 3D marker locations. These are then projected into the image to obtain predicted 2D locations. Let $\hat{\mathbf{d}}_l(\mathbf{s})$ be the 2D location for the l^{th} marker that is predicted by state **s**. With this, we can write the likelihood function for state **s** and image observations $\mathbf{z} = \{\mathbf{d}_l(\mathbf{s})\}_{l=1}^{17}$ as

$$p(\mathbf{z}\,|\,\mathbf{s}) = \left(\frac{1}{2\pi\sigma^2}\right)^{17} \exp\left[-\frac{\sum_{l=1}^{17} ||\mathbf{d}_l - \hat{\mathbf{d}}_l(\mathbf{s})||^2}{2\sigma^2}\right]. \tag{20}$$

The state space is 28-dimensional: There are 6 degrees of freedom (DOF) for spine position and orientation, 3 DOFs for the head and for each hip and arm/shoulder joint, 2 DOFs for the pelvis, and 1 DOF joints for the shoulder/spine, the elbows and the knees. For a temporal prior over these state variables, we simply assume that states change slowly over time. As mentioned above, the temporal prior for states $\mathbf{s}_t$ given $\mathbf{s}_{t-1}$ is a Gaussian centered at $\mathbf{s}_{t-1}$ with variances σ_d^2 and σ_a^2 for translation and angular variables. This prior, along with (20), defines the unnormalized posterior which is straightforward to differentiate, as is required for HMC transitions.

d	# Particle filter samples (1000's)	# HMC chains	R	ϵ	L
4	0.32, 0.64, 1.28, 2.56, 5.12, 10.24, 20.48	4	8, 16, 32, 64, 128, 256, 512	10.3	2
10	5.7, 11.4, 22.8, 45.6, 91.2, 182.4, 364.8	8	8, 16, 32, 64, 128, 256, 512	4.0	20
28	27, 54, 108, 216, 432, 864, 1,728	10	8, 16, 32, 64, 128, 256, 512	3.0	40

Table 1. Particle filter and HMC settings for single frame experiments. Each row gives settings used for dimension d. Numbers of samples used for the particle filter are in column 2. Remaining columns show HMC settings (only R was varied to equate computation times of the two filters).

In practice one could exploit the Markov tree model of the body to sample the structure more efficiently, as in [12, 18]. Here, while we do examine the tracking of the entire body as well as its parts, our main concern is the relative performance of HMC filters and particle filters. Also, note that our human model is weak in that we do not model limits on how far joints can rotate in reality. It is therefore possible for the tracker to recover anatomically impossible poses. Similarly, limb lengths are determined from the dimensions of the particular human subject before tracking begins, and are then fixed during tracking. This is a source of error because real joints are more complicated than our model, causing some parts such as the shoulder to vary in length over time.

7.3 Single Frame Experiments

Because filtering is recursive, it is useful to examine performance for a single time step. For both HMC and particle filters we implemented three trackers: a 4D arm tracker, a 10D lower body tracker and the 28D full body tracker. Each was applied to three different, wide-spaced frames from sequences of people dancing and walking. When tracking a restricted subset of the states, as in the arm and the lower body trackers, we clamp the untracked state variables to their true values (obtained from the motion capture system).

In all cases the standard deviation of the observation noise in (18) was set to $\sigma = 0.003$ (about 3% the length of an upright spine projected onto the image). The standard deviations used for the Gaussian dynamics in temporal prior were $\sigma_d = 15.0$mm and $\sigma_a = 0.15$ radians for the translational and angular state variables. The HMC and particle filters have the same prior and the same likelihood and hence they are both aiming to approximate the same posterior.

As suggested in Sec. 3, we assess the two filters on how

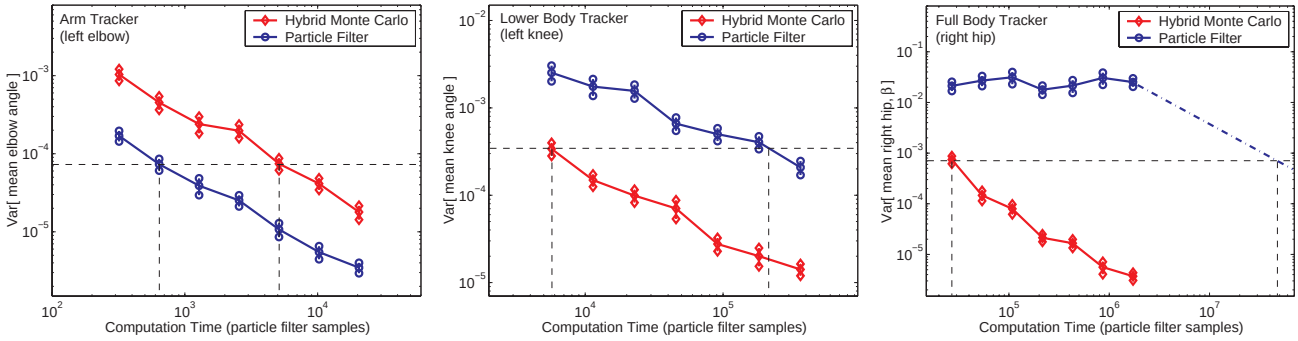

Figure 2. Estimator variances for the mean state for individual state variables in single frame experiments. Error bars show one standard error. As computation time increases, estimator variances decrease. In each case one can fix the variance, and compare the compute times required to acheive that variance. In this way, notice that the particle filter about 8 times faster in the 4D case. But in 10D the HMC filter is about 50 times faster than the particle filter. In the 28D case, the same analysis requires that we extrapolate the data for the particle filter, which we did assuming optimal statistical efficiency of each subsequent sample drawn by the particle filter. But even with this optimistic extrapolant, the HMC filter remains over 2000 times faster than the particle filter.

well they estimate the posterior mean. The mean estimated by the particle filter is simply the weighted mean state as suggested by (2). For the HMC filter, the mean is given by (4), for which we use all samples from all Markov chains, except for the first 6 (the burn-in samples) along each chain. To get the true posterior mean for the arm and lower body experiments, we ran the particle filter with orders of magnitude more particles than used in the tests below. For the full body tracker, we approximated the true posterior mean using an extremely long HMC run to obtain many samples.

From 50 runs of the two filters, with random sampling and noise on each trial, we computed the estimator variance. The marginal variances of each state component are given by (3) with f equal to a projection operator from $\mathbf{s}$ to $\mathbf{s}_i$. Of course, these variances are only comparable if the two filters use the same amount of computation time. Table 1 shows the parameters used to equate computation time; for the particle filter the computation time depends on the number of particles, while for the HMC filter we vary the number of samples R produced by each Markov chain.

Figure 2 shows marginal variances for single state variables as a function of computation time. For the 4D arm tracker the particle filter outperforms. For the 10D lower body tracker the situation reversed, with the HMC filter outperforming. For the 28D full-body tracker the HMC filter is vastly superior; the particle filter shows little variance reduction as the number of particles increases, and the effective number of particles stays close to 1. By fixing the estimator variance, one can also use these plots to compare computation times. For the arm tracker the particle fitler is approximately 8 times faster. But for the 10D tracker the HMC filter is about 50 times faster, and as explained in the figure caption, for the 28D full body tracker the HMC method is more than 2000 times faster.

To put this in perspective, as discussed in Sec. 3, one can use the effective volumes of the prior and the posterior to estimate a lower bound on number of particles needed for these problems. Using more than enough particles for the arm and lower body trackers we measured the covariances of the prior and the posterior. Following Sec. 3, the geometrical means of the marginal variances measure the volumes contained within 1-standard deviation hyperellipses, the ratios of which bound the number of particles. For the 4D tracker we find a lower bound of about 10^2 particles, while for the 10D tracker the bound increases to 10^5 particles. We are unable to perform this test in 28D due to the large number of particles required, but an extrapolation from the smaller cases yields a bound of about 10^{13} particles.

Finally, to get a feeling for the estimator variances of other state variables, we show the geometric mean variance, computed over all variables. Fig. 3 shows the geometric mean variance as a function of computation time at three randomly selected frames in a walking sequence. These mean variances behave like the marginal variances in Fig. 2.

7.4 Multiple Frame Experiments

It is also useful to compare the two filters over multiple time steps. We applied the 10D lower-body and the 28D full-body trackers to a 25-frame walking sequence. All parameters and initialization were identical to the single frame case. Using Table 1, for the lower body tracker we used $R = 32$ for HMC, and $N = 23,000$ for the particle filter. For the full body, we used $R = 32$ and $N = 108,000$.

Tracking was performed 50 times using different observation noise instantiations each time. We average the log likelihood of the mean state over the 50 runs, and do the same for the log prior density of the true state. Both of these quantities are expected to be larger if the tracker is performing well. As expected from the single frame experiments, Fig. 4 and 5 show the superiority of the HMC filter.

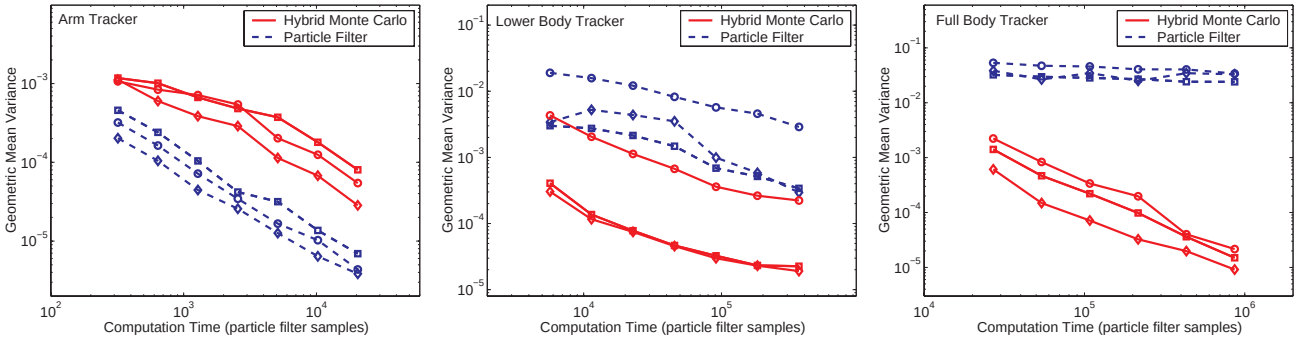

Figure 3. Geometric mean variances over all state variable, for the HMC filter and the particle filter on 3 tracking experiments.

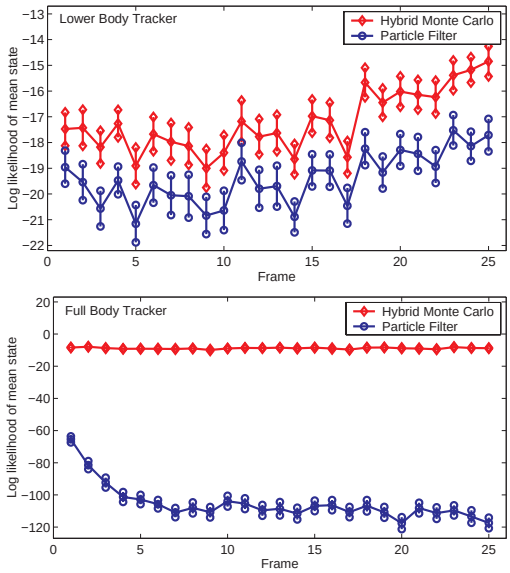

Figure 4. The log likelihood of the mean state over 25 frames. Error bars are 1 standard error.

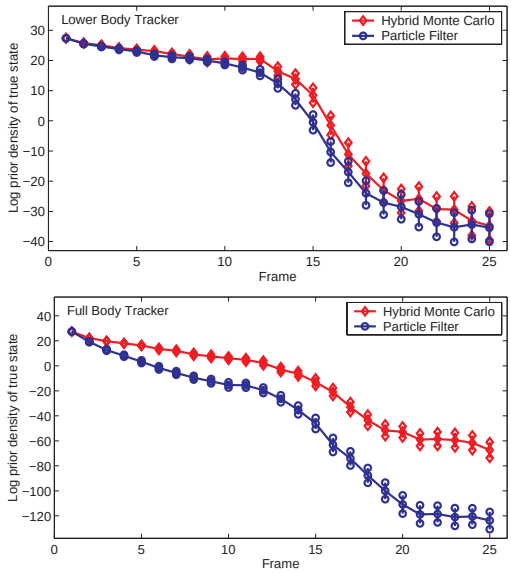

Figure 5. The log prior of the true state over 25 frames. Error bars are 1 standard error.

Finally, Fig. 6 shows six poses recovered by each method at frame 10, compared to the true pose. HMC estimates are closer to the true pose and produce consistently higher likelihoods. Because we only have monocular observations, and our human model is weak, there are insufficient constraints to recover the exact pose at each frame. For instance, the mean arms found by the HMC filter appear consistent with the observations, but are clearly variable in 3D.

7.5 Tuning Hybrid Monte Carlo

We tuned the parameters for our runs carefully but not obsessively. The HMC filter has 5 tuning parameters, namely, the number of chains M, the stepsize adjustment ϵ, the leapfrog trajectory length L, the number of HMC updates R, and the number of burn-in samples b. Here, M should be chosen to be the number of independent samples from which one hopes to form the temporal prior. The stepsize adjustment ϵ is best set to give roughly 15% rejection rate. For tracking over multiple frames, good ϵ's will differ from frame to frame, so it may be desirable to adapt ϵ for each frame; in this paper, ϵ was fixed.

The number of leapfrog steps L governs how rapidly you sample from the posterior. Chains with small L may reach equilibrium faster, but often lead to inefficient random walks if too small. For multiple frame tracking, we fixed L by examining the behaviour of a few MCs for the first frame only; i.e., we looked for a value of L that yields low MC autocorrelations, with rejection rates close to 15%.

Finally, b should be chosen so that each chain has reached equilibrium after b updates. There is no guarantee that equilibrium will be reached, but available diagnostics [6] can help to adaptively determine b for each chain.

8 Discussion and Future Work

In summary, we have seen how the hybrid Monte Carlo filter samples high dimensional distributions more efficiently than the particle filter. Unlike the particle filter, for which the number of particles grows exponentially with state space dimensionality, the HMC filter scales so much better that it offers speedups of several orders of magnitude

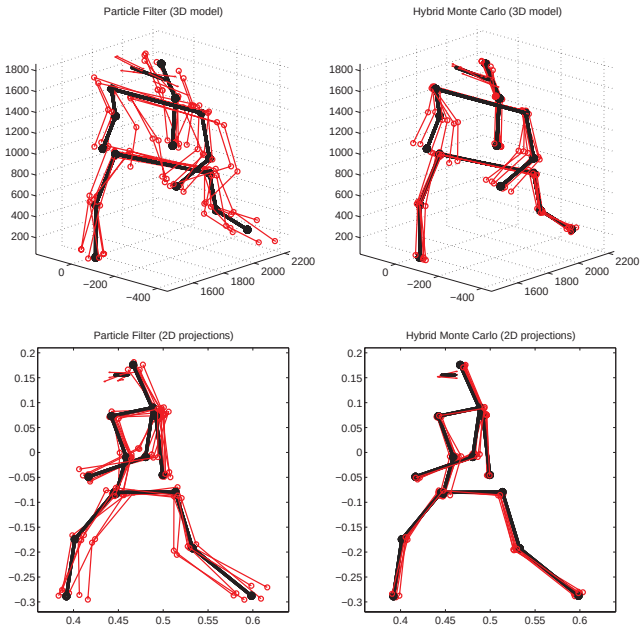

Figure 6. Posterior mean poses recovered by each method at frame 10, from 6 random trials of the 28D tracker. The true 3D data (top) and the 2D observations (bottom) are shown with thick black lines. The estimates from the two filters are drawn with thin red lines. 108,000 particles were used for the particle filter, and equivalent computation time was used by the HMC filter.

in our 10D and 28D people tracking experiments. We believe that the hybrid Monte Carlo filter represents a promising new class of MCMC-based filters.

For future work, more sophisticated versions of the HMC filter are possible. HMC particles can have a hard time moving between modes, which is why we use multiple chains; for strongly multimodal distributions, multiple Markov chains should be started in different parts of state space to capture all the modes. One can also use more than just the final state of the Markov chains to form the temporal prior in the next frame; states from earlier in each chain can be used as well. Methods for diagnosing equilibrium would also be helpful. As mentioned however, there is no foolproof way of diagnosing equilibrium, and this is true for any MCMC-based technique. But it does not matter if we use chains that have not fully reached equilibrium; so long as a method provides a better approximation to the desired posterior in a given amount of time, it is preferable.

Acknowledgements: The authors thank Allan Jepson and Radford Neal for their comments on this research, and Michael Gleicher for the motion capture data.

References

[1] M.J. Black and A.D. Jepson. A probabilistic framework for matching temporal trajectories: Condensation-based recognition of gestures and expressions. ECCV-98, pp. 909–924

[2] M.J. Black and D.J. Fleet. Probabilistic detection and tracking of motion discontinuities. *IJCV*, 38:229–243, 2000

[3] J. Carpenter, P. Clifford, and P. Fearnhead. Improved particle filter for nonlinear problems. *IEE Proc.-Radar, Sonar Navig.*, 146:2–4, 1999

[4] T. Cham and J.M. Rehg. A multiple hypothesis approach to figure tracking. CVPR-98, Fort Collins, V II, pp. 239–245

[5] K. Choo. Learning hyperparameters for neural network models using Hamiltonian dynamics. Master's thesis, Dept. Computer Science, Univ. Toronto, 2000

[6] M.K. Cowles and B.P. Carlin. Markov chain Monte Carlo convergence diagnostics: A comparative review. *JASA*, 91:883–904, 1996

[7] J. Deutscher, A. Blake, and I. Reid. Articulated body motion capture by annealled particle filtering. CVPR-00, Hilton Head, V II pp. 126–133

[8] A. Doucet, N. de Freitas, and N. Gordon. *Sequential Monte Carlo Methods in Practice*. Springer-Verlag, Berlin, 2001

[9] S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth. Hybrid Monte Carlo. *Physics Letters B*, 195:216–222, 1987

[10] D. Fox, W. Burgard, F. Dellaert, and S. Thrun. Monte carlo localization: Efficient position estimation for mobile robotics. AAAI-99, pp. 343–349

[11] N.J. Gordon, D.J. Salmond, and A.F.M. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proc. Radar, Sonar and Navig.*, 140:107–113, 1993

[12] S. Ioffe and D. Forsyth. Finding people by sampling. ICCV-99, Corfu, V. II, pp. 1092–1097

[13] M. Isard and A. Blake. Condensation - conditional density propagation for visual tracking. *IJCV*, 29:2–28, 1998

[14] M. Isard and A. Blake. Icondensation: Unifying low-level and high-level tracking in a stochastic framework. ECCV-98, Freiburg, V. I, pp. 893–908

[15] M. Leventon and W. Freeman. Bayesian estimation of 3-d human motion from an image sequence. MERL TR-98-06

[16] J. Liu and M. West. Combined parameter and state estimation in simulation-based filtering. In A. Doucet, N. de Freitas, and N. Gordon, editors, *Sequential Monte Carlo Methods in Practice*. Springer-Verlag, Berlin, 2001.

[17] J.S. Liu and R. Chen. Sequential Monte Carlo methods for dynamic systems. *JASA*, 93:1032–1044, 1998

[18] J. MacCormick and M. Isard. Partitioned sampling, articulated objects, and interface-quality hand tracking. ECCV-00, Dublin, V II, pp. 3–19

[19] D.J.C. MacKay. Introduction to Monte Carlo methods. In M.I. Jordan (ed), *Learning In Graphical Models*, pp. 175–204. MIT Press, 1999.

[20] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *J. Chem. Phys.*, 21:1087–1092, 1953

[21] R.M. Neal. *Bayesian Learning for Neural Networks*. Springer-Verlag, NY, 1996. Lecture Notes in Stats., No. 118

[22] J.S. Rosenthal. A review of asymptotic convergence of general state space markov chains. *Far East J. Theor. Stat.*, 5:37-50, 2001

[23] H. Sidenbladh, M.J. Black, and D.J. Fleet. Stochastic tracking of 3d human figures using 2d image motion. ECCV-00, Dublin, V II, pp. 702–718