CSC487/2503: Foundations of Computer Vision

Visual Tracking

David Fleet

Introduction

What is tracking?

Major players:

- Dynamics (model of temporal variation of target parameters)
- Measurements (relation between image & target parameters)
- Search (finding good values for target parameters)
- Initialization (establish initial state of tracker)

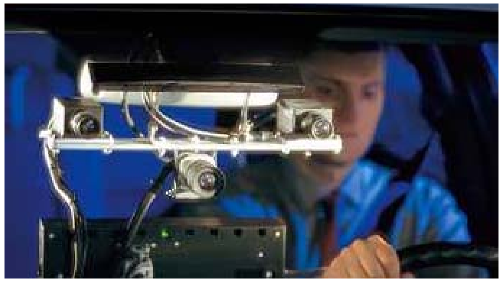
Why tracking (why not recognition?)

- Accumulated information (impoverished local measurements)
- Track then detect
- Establish what went where (correspondence)
- Effective / Improved search
- Simultaneous activity recognition / tracking

Introduction: Potential Applications

Automotive

- traffic analysis / control
- driver attentiveness
- obstacles (pedestrians and vehicles)

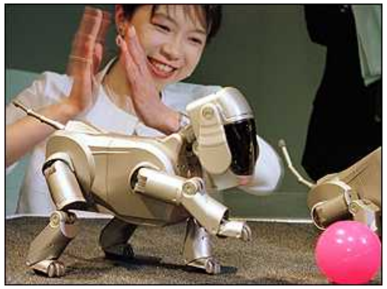


Looking @ People

- surveillance/security (homes, stores, …)
- annotating human activity (football plays)

Entertainment

- motion capture, interactive video w/ CG
- Interactive toys



Military

- target detection, tracking, and prediction

Science

- cell motility assays

Introduction

What's so hard?

- complex nonlinear dynamics, with high dimensional object models
- complex appearance, and temporal appearance variation (deformable objects, shadows & lighting variations, clothing, …)
- impoverished information due to occlusion or scale
- visual background clutter and distractors

Introduction

What are we tracking?

- position, velocity, acceleration
- shape, size, deformation
- 3d structure
- …

Which image properties should we track?

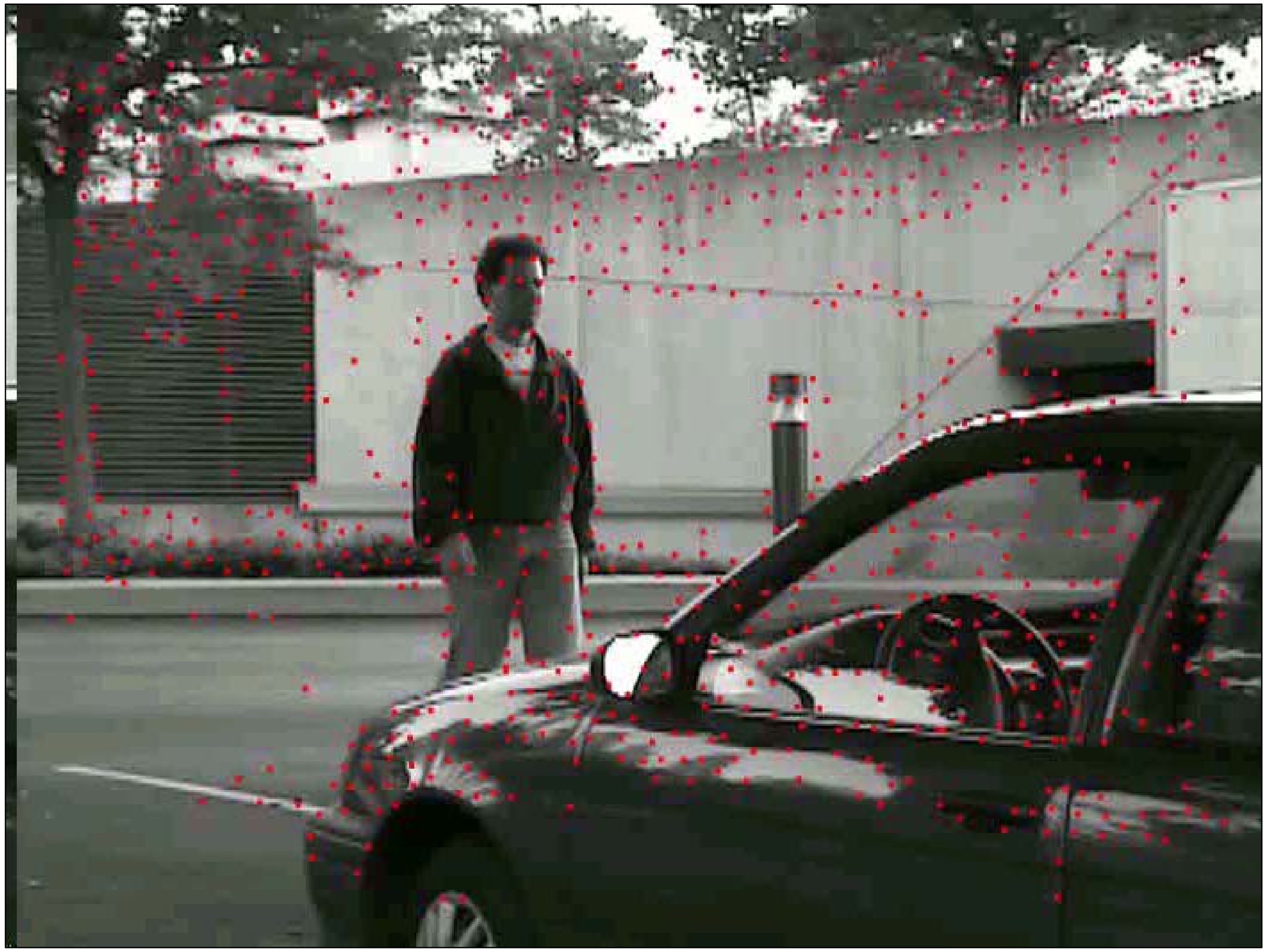
- intensity / colors
- region statistics
- contours (edges)
- shapes
- motion

Introduction

What simplifies the problem/solution in practice?

- known/stationary background (e.g., track blobs)
- distinct *a priori* colors (e.g., skin)
- multiple cameras (often 2 or 3)
- manual initialization
- strong dynamics models
- prior knowledge of the number of objects and object types
- sufficient object size
- limited occlusion

2D (Local) Region Tracking

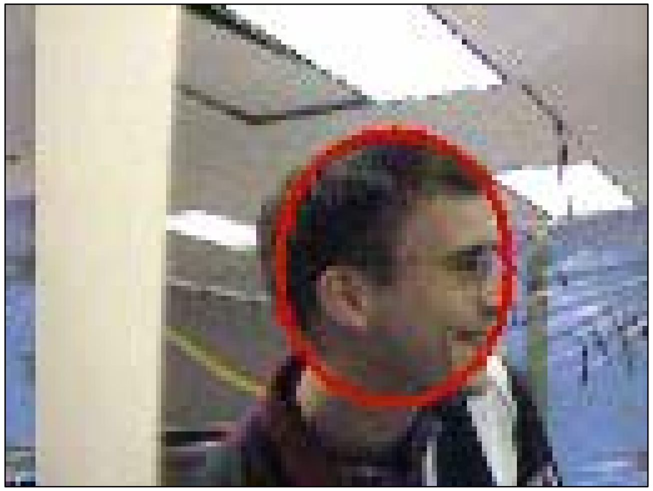


[Shi and Tomasi, “Good features to track.” Proc IEEE CVPR, 1994]

2D Region Tracking with Appearance Models

[Jepson, Fleet, & El-Maraghi, "Robust, on-line appearance models for visual tracking." IEEE Trans. On PAMI, 2003]

2D Region Tracking

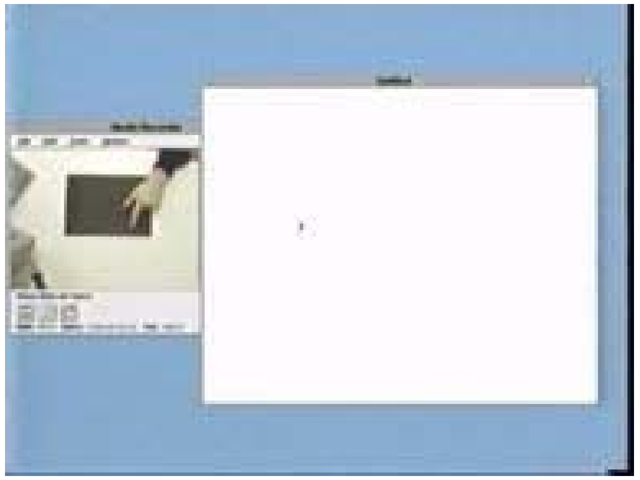


[Birchfield, “Elliptical head tracking using intensity gradients and color histograms.” Proc IEEE CVPR, 1998]

Joint Color/Space Tracking with Mean Shift

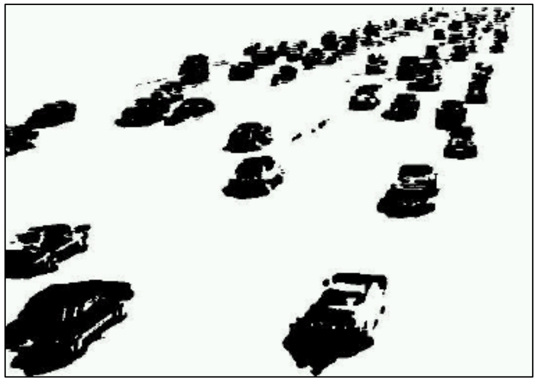
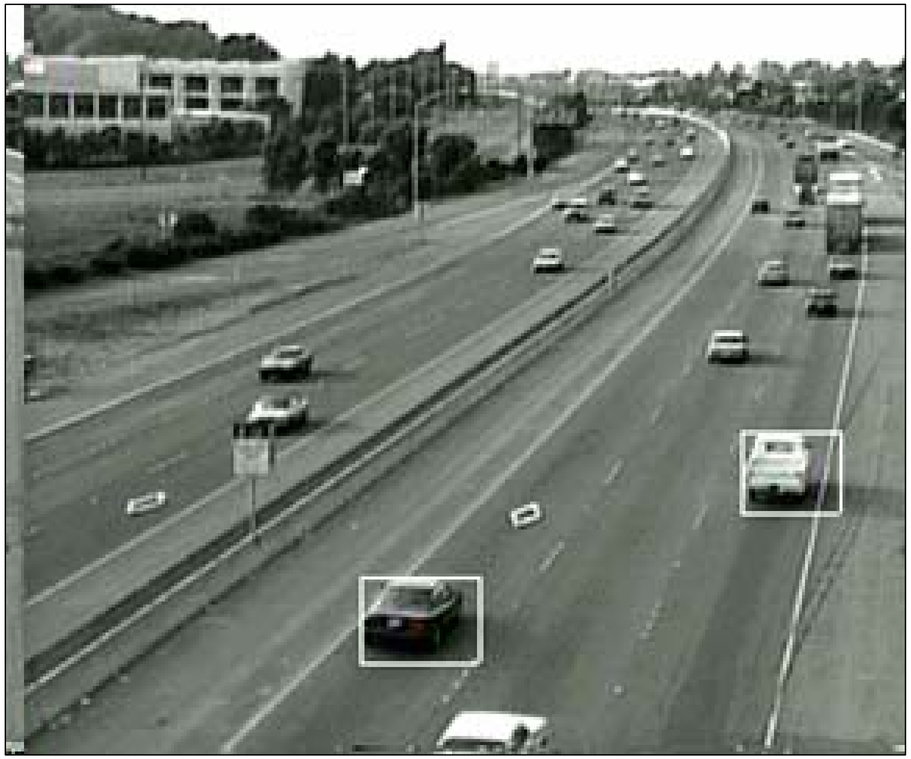
[Comaniciu and Meer, “Mean Shift …”, CVPR 2000]

Contour Tracking



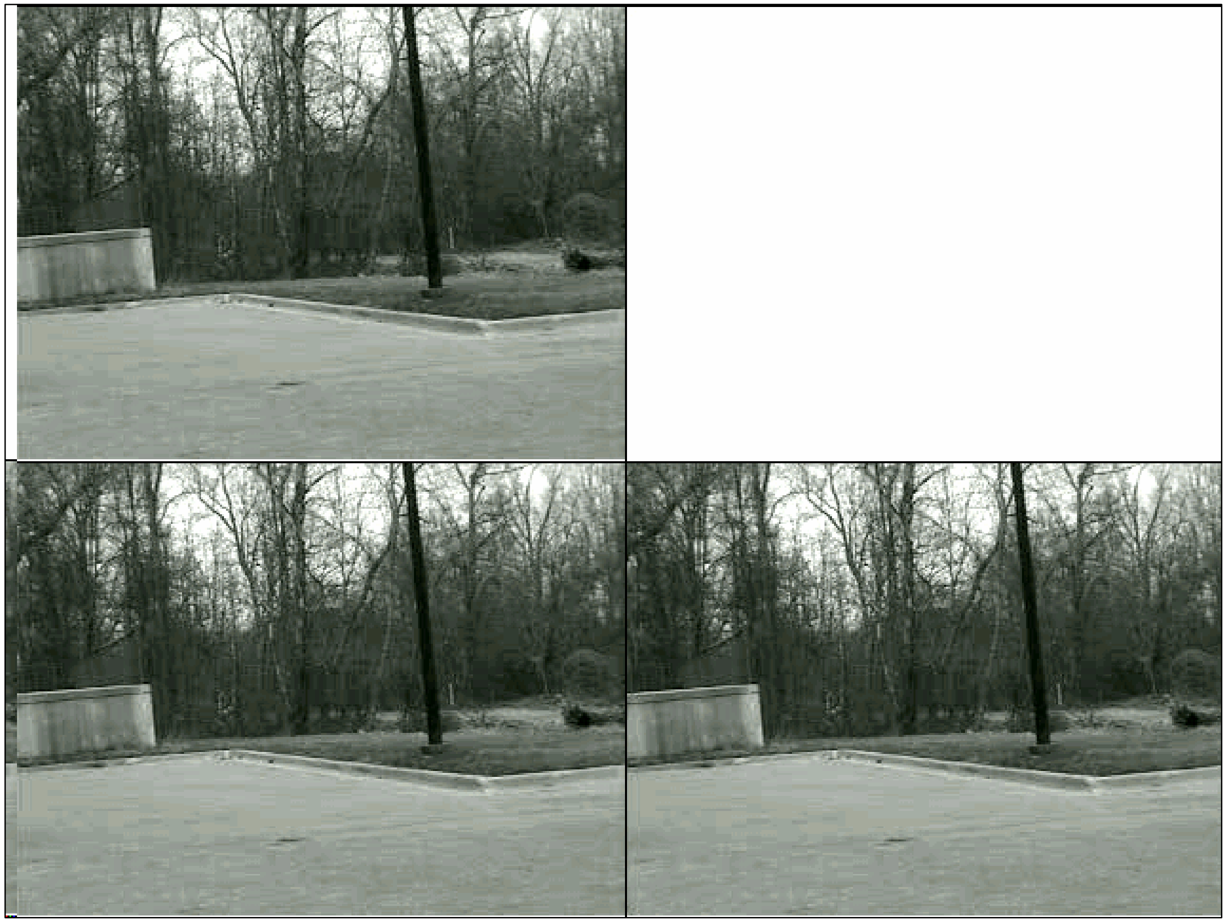
[Isard & Blake, "Condensation - conditional density propagation for visual tracking." IJCV, 1998]

Car Tracking / Background Subtraction

[Koller, Weber & Malik, “Robust multiple car tracking with occlusion reasoning.” Proc ECCV,1994]

2.5D People Tracking

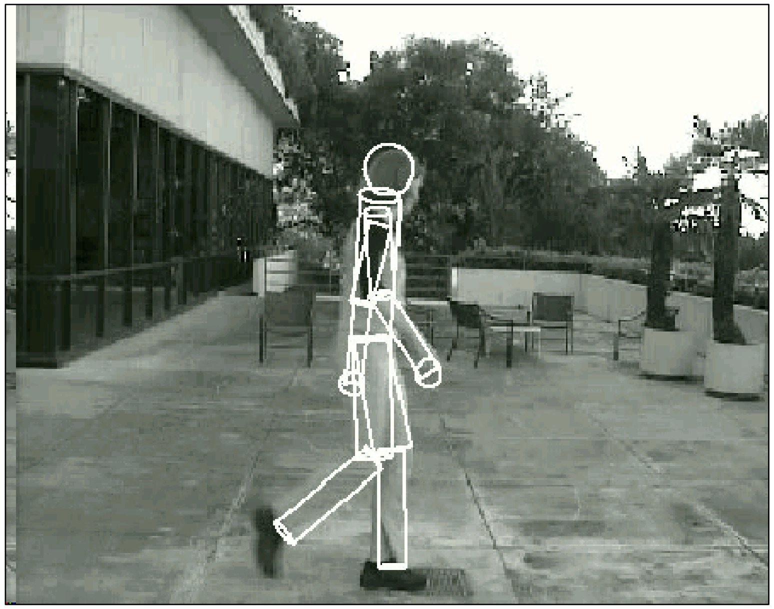


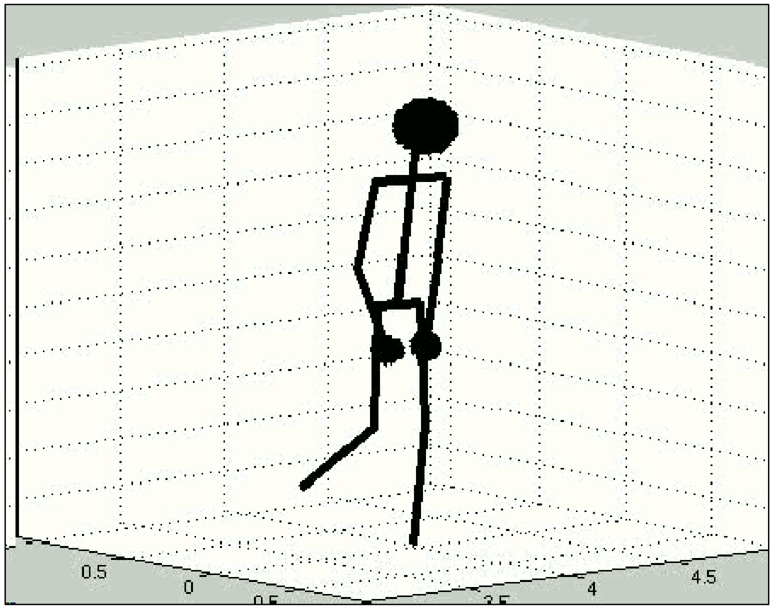
[Haritaoglu, Harwood & Davis, "W4: Who, when, where, what: A real-time system for detecting and tracking people." Proc Face & Gesture Conf, 1998]

2.1 People Tracking

[Isard and MacCormick, “Bramble: … .” Proc ICCV, 2001]

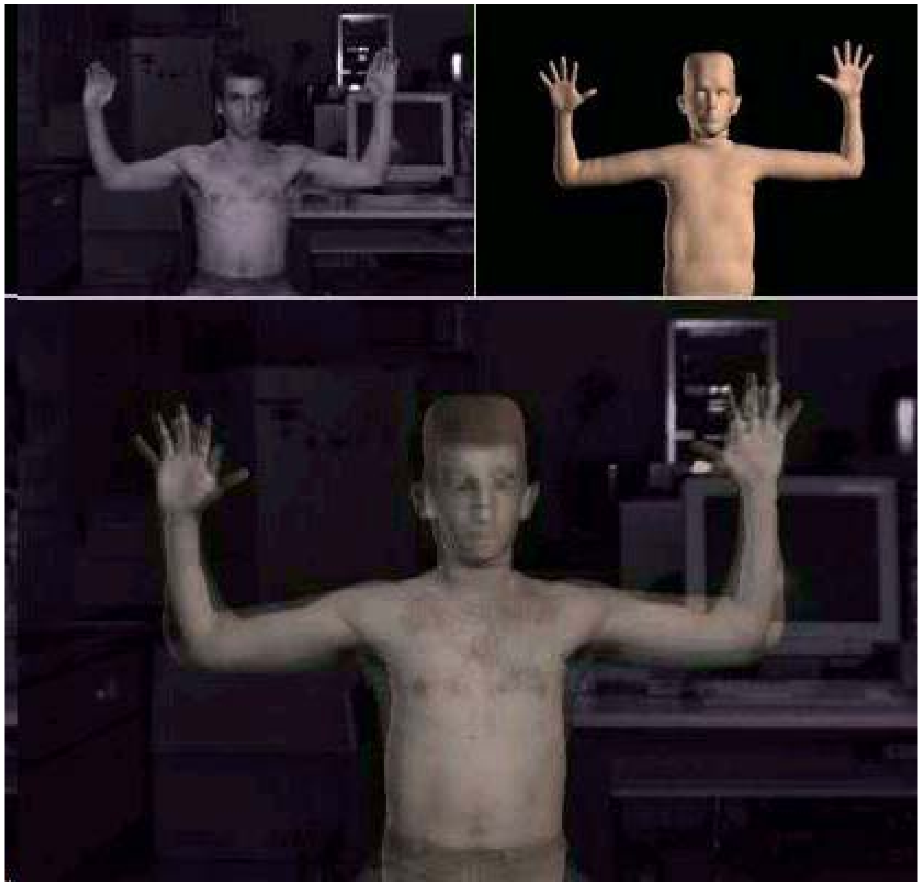
3D People Tracking





[Sidenbladh, Black & Fleet, “3D people tracking using particle filtering.” Proc ECCV, 2000]

3D People Tracking



[Plankers & Fua, “Articulated soft objects for video-based body modeling.” Proc IEEE ICCV, 2001]

Introduction

What’s the goal (state of the art)?

- how many frames … 10, 100, 1000, 10000 …
- batch or on-line
- prior specification / learning of object model
- real-time
- what class of motion (e.g., low-dim 2d deformation, rigid objects, smooth textured surfaces, deformable objects, …)

State Space

Object model is a set of variables (properties of the image or scene) that we wish to estimate as a function of time?

State: n-vector containing variables to be estimated: $\vec{\mathbf{x}}_t$

- continuous (real-valued) state space
  [e.g., position, velocity, acceleration, size, shape, …]
- discrete state space
  [e.g., number of objects, gender, activity type, …]
- hybrid state space (with discrete & continuous variables being functions of one another)

Observations / Measurements: data vector from which we estimate the state variables: $\vec{\mathbf{z}}_t = f(\vec{\mathbf{x}}_t)$

Probabilistic Formulation

Remember:

- Conditioning (factorization):

$$p(a, b) = p(a|b)\, p(b) = p(b|a)\, p(a)$$

- Bayes' rule:

$$p(a|b) = \frac{p(b|a)\, p(a)}{p(b)}$$

- Marginalization:

$$p(b) = \int p(a, b)\, da \qquad\qquad p(b) = \sum_a p(a, b)$$

Probabilistic Formulation

- State history: $\vec{\mathbf{x}}_{1:t} = (\vec{\mathbf{x}}_1, \ldots, \vec{\mathbf{x}}_t)$
- Observation history: $\vec{\mathbf{z}}_{1:t} = (\vec{\mathbf{z}}_1, \ldots, \vec{\mathbf{z}}_t)$
- Probability distribution over states conditioned on observations

$$p(\vec{\mathbf{x}}_{1:t} \,|\, \vec{\mathbf{z}}_{1:t})$$

- Marginal distribution over current state given observation history

$$p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{z}}_{1:t}) \;=\; \int_{\vec{\mathbf{x}}_1} \cdots \int_{\vec{\mathbf{x}}_{t-1}} p(\vec{\mathbf{x}}_{1:t} \,|\, \vec{\mathbf{z}}_{1:t})$$

called posterior distribution (or filtering distribution)

Probabilistic Models / Assumptions

- First-order Markov model for the state dynamics:

$$p(\vec{\mathbf{x}}_t \mid \vec{\mathbf{x}}_{1:t-1}) \;=\; p(\vec{\mathbf{x}}_t \mid \vec{\mathbf{x}}_{t-1})$$

therefore

$$p(\vec{\mathbf{x}}_{1:t}) \;=\; \left(\prod_{j=2}^{t} p(\vec{\mathbf{x}}_j \mid \vec{\mathbf{x}}_{j-1}) \right) p(\vec{\mathbf{x}}_1)$$

- Conditional independence of observations

$$\underbrace{p(\vec{\mathbf{z}}_{1:t} \mid \vec{\mathbf{x}}_{1:t})}_{\text{joint likelihood}} \;=\; p(\vec{\mathbf{z}}_t \mid \vec{\mathbf{x}}_t)\, p(\vec{\mathbf{z}}_{1:t-1} \mid \vec{\mathbf{x}}_{1:t-1})$$

$$=\; \prod_{\tau=1}^{t} \underbrace{p(\vec{\mathbf{z}}_\tau \mid \vec{\mathbf{x}}_\tau)}_{\text{likelihood at time } \tau}$$

Filtering Equations

Bayes' rule

$$p(a|b) = \frac{p(b|a)\,p(a)}{p(b)}$$

$$\begin{aligned}
p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{z}}_{1:t}) &= \int_{\vec{\mathbf{x}}_1} \cdots \int_{\vec{\mathbf{x}}_{t-1}} p(\vec{\mathbf{x}}_{1:t} \,|\, \vec{\mathbf{z}}_{1:t}) \\
&= \frac{1}{p(\vec{\mathbf{z}}_{1:t})} \int_{\vec{\mathbf{x}}_1} \cdots \int_{\vec{\mathbf{x}}_{t-1}} p(\vec{\mathbf{z}}_{1:t} \,|\, \vec{\mathbf{x}}_{1:t})\, p(\vec{\mathbf{x}}_{1:t}) \\
&= c \int_{\vec{\mathbf{x}}_1} \cdots \int_{\vec{\mathbf{x}}_{t-1}} p(\vec{\mathbf{z}}_t \,|\, \vec{\mathbf{x}}_t)\, p(\vec{\mathbf{z}}_{1:t-1} \,|\, \vec{\mathbf{x}}_{1:t-1})\, p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{x}}_{t-1}) p(\vec{\mathbf{x}}_{1:t-1}) \\
&= c\, p(\vec{\mathbf{z}}_t \,|\, \vec{\mathbf{x}}_t) \int_{\vec{\mathbf{x}}_1} \cdots \int_{\vec{\mathbf{x}}_{t-1}} p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{x}}_{t-1})\, p(\vec{\mathbf{x}}_{1:t-1}, \vec{\mathbf{z}}_{1:t-1}) \\
&= c\, p(\vec{\mathbf{z}}_t \,|\, \vec{\mathbf{x}}_t) \int_{\vec{\mathbf{x}}_{t-1}} p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{x}}_{t-1}) \int_{\vec{\mathbf{x}}_1} \cdots \int_{\vec{\mathbf{x}}_{t-2}} p(\vec{\mathbf{x}}_{1:t-1}, \vec{\mathbf{z}}_{1:t-1}) \\
&= c\, p(\vec{\mathbf{z}}_t \,|\, \vec{\mathbf{x}}_t) \int_{\vec{\mathbf{x}}_{t-1}} p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{x}}_{t-1})\, p(\vec{\mathbf{x}}_{t-1}, \vec{\mathbf{z}}_{1:t-1})
\end{aligned}$$

Filtering Equations

$$\begin{aligned} p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{z}}_{1:t}) &= c\, p(\vec{\mathbf{z}}_t \,|\, \vec{\mathbf{x}}_t) \int_{\vec{\mathbf{x}}_{t-1}} p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{x}}_{t-1})\, p(\vec{\mathbf{x}}_{t-1},\, \vec{\mathbf{z}}_{1:t-1}) \\ &= \frac{c\, p(\vec{\mathbf{z}}_t \,|\, \vec{\mathbf{x}}_t)}{p(\vec{\mathbf{z}}_{1:t-1})} \boxed{\int_{\vec{\mathbf{x}}_{t-1}} p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{x}}_{t-1})\, p(\vec{\mathbf{x}}_{t-1} \,|\, \vec{\mathbf{z}}_{1:t-1})} \\ &= c'\, p(\vec{\mathbf{z}}_t \,|\, \vec{\mathbf{x}}_t)\, p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{z}}_{1:t-1}) \end{aligned}$$

Prediction distribution (temporal prior):

$$p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{z}}_{1:t-1}) = \int_{\vec{\mathbf{x}}_{t-1}} p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{x}}_{t-1})\, p(\vec{\mathbf{x}}_{t-1} \,|\, \vec{\mathbf{z}}_{1:t-1})$$

Filtering Equations

Recursive (filtering) computation:

- begin with posterior from previous time $p(\vec{\mathbf{x}}_{t-1} \,|\, \vec{\mathbf{z}}_{1:t-1})$
- propagate forward in time using dynamical model $p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{x}}_{t-1})$ to form the prediction distribution $p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{z}}_{1:t-1})$
- incorporate current observations (evidence) $p(\vec{\mathbf{z}}_t \,|\, \vec{\mathbf{x}}_t)$ to form the new filtering (posterior) distribution $p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{z}}_{1:t})$

Animation?

Filtering and Smoothing

Recursive computation (smoothing) backwards in time:

$$
\begin{aligned}
p(\vec{\mathbf{x}}_\tau \,|\, \vec{\mathbf{z}}_{\tau:t}) &= c\, p(\vec{\mathbf{z}}_\tau \,|\, \vec{\mathbf{x}}_\tau) \int_{\vec{\mathbf{x}}_{\tau+1}} p(\vec{\mathbf{x}}_\tau \,|\, \vec{\mathbf{x}}_{\tau+1})\, p(\vec{\mathbf{x}}_{\tau+1},\, \vec{\mathbf{z}}_{\tau+1:t}) \\
&= c\, p(\vec{\mathbf{z}}_\tau \,|\, \vec{\mathbf{x}}_\tau)\, p(\vec{\mathbf{x}}_\tau \,|\, \vec{\mathbf{z}}_{\tau+1:t})
\end{aligned}
$$

Optimal computation given entire sequence: filter & smooth

$$
p(\vec{\mathbf{x}}_\tau \,|\, \vec{\mathbf{z}}_{1:t}) = c\, \underbrace{p(\vec{\mathbf{z}}_\tau \,|\, \vec{\mathbf{x}}_\tau)}_{\text{current evidence}}\, \underbrace{p(\vec{\mathbf{x}}_\tau \,|\, \vec{\mathbf{z}}_{1:\tau-1})}_{\text{prediction from past data}}\, \underbrace{p(\vec{\mathbf{x}}_\tau \,|\, \vec{\mathbf{z}}_{\tau+1:t})}_{\text{prediction from future data}}
$$

[Belief Propagation: combining predictions (or messages) from neighbors with local evidence to compute local estimates of state]

Temporal Dynamics

Linear dynamical models:

- Random walk with zero velocity

$$\vec{\mathbf{x}}_t = \vec{\mathbf{x}}_{t-1} + \boxed{\vec{\eta}_d} \longleftarrow \text{process noise}$$

- Random walk with zero acceleration

$$\text{extended state space} \quad \boxed{\begin{pmatrix} \vec{\mathbf{x}}_t \\ \vec{\mathbf{v}}_t \end{pmatrix}} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \vec{\mathbf{x}}_{t-1} \\ \vec{\mathbf{v}}_{t-1} \end{pmatrix} + \begin{pmatrix} \vec{\eta}_d \\ \vec{\epsilon}_d \end{pmatrix}$$

- Harmonic oscillation (1D case)

$$\boxed{\frac{d^2x}{dt^2} = -x} \qquad \frac{d\vec{\mathbf{u}}}{dt} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \vec{\mathbf{u}} \quad \text{where } \vec{\mathbf{u}} = \begin{pmatrix} x \\ v \end{pmatrix}$$

so

$$\begin{aligned} \vec{\mathbf{u}}_t &= \vec{\mathbf{u}}_{t-1} + \Delta t \frac{d\vec{\mathbf{u}}}{dt} + \begin{pmatrix} \eta \\ \epsilon \end{pmatrix} \\ &= \begin{pmatrix} 1 & \Delta t \\ -\Delta t & 1 \end{pmatrix} \vec{\mathbf{u}}_{t-1} + \begin{pmatrix} \eta \\ \epsilon \end{pmatrix} \quad \text{linear dynamics} \end{aligned}$$

Temporal Dynamics

Linear dynamical models:

- Second-order Markov

$$p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{x}}_{1:t-1}) \;=\; p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{x}}_{t-1}, \, \vec{\mathbf{x}}_{t-2})$$

Express as first-order in an augmented state space:

$$\begin{pmatrix} \vec{\mathbf{x}}_t \\ \vec{\mathbf{y}}_t \end{pmatrix} \;=\; \begin{pmatrix} ? & ? \\ 1 & 0 \end{pmatrix} \begin{pmatrix} \vec{\mathbf{x}}_{t-1} \\ \vec{\mathbf{y}}_{t-1} \end{pmatrix} + \begin{pmatrix} \vec{\eta}_d \\ \vec{0} \end{pmatrix}$$

e.g.

$$\begin{pmatrix} \vec{\mathbf{x}}_t \\ \vec{\mathbf{y}}_t \end{pmatrix} \;=\; \begin{pmatrix} 2 & -1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} \vec{\mathbf{x}}_{t-1} \\ \vec{\mathbf{y}}_{t-1} \end{pmatrix} + \begin{pmatrix} \vec{\eta}_d \\ \vec{0} \end{pmatrix}$$

constant velocity

$$\vec{\mathbf{x}}_t \;=\; \vec{\mathbf{x}}_{t-1} + (\vec{\mathbf{x}}_{t-1} - \vec{\mathbf{x}}_{t-2}) + \vec{\eta}_d$$

Appropriate for a wide range of system dynamics, including harmonic oscillation and random walk with drift.

Temporal Dynamics

Hybrid dynamics (continuous variables are functions of discrete variables):

- Discrete dynamics given by transition probability matrix
- For each position in the matrix we have a continuous dynamical model that involves the relevant continuous state variables

Learned dynamics:

- Use training ensemble of motions to learn low-dimensional linear subspaces or nonlinear manifolds

Kalman Filter

Assume linearity + Gaussian uncertainty for both observation equations and state dynamics:

$$\begin{aligned} \vec{\mathbf{x}}_t &= A\,\vec{\mathbf{x}}_{t-1} + \vec{\eta}_d & \qquad \vec{\eta}_d &\sim N(0, C_d) \\ \vec{\mathbf{z}}_t &= M\,\vec{\mathbf{x}}_t + \vec{\eta}_m & \qquad \vec{\eta}_m &\sim N(0, C_m) \end{aligned}$$

The conditional Markov and likelihood distributions become:

$$\begin{aligned} p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{x}}_{t-1}) &= G(\vec{\mathbf{x}}_t - A\vec{\mathbf{x}}_{t-1},\, C_d) \\ p(\vec{\mathbf{z}}_t \,|\, \vec{\mathbf{x}}_t) &= G(\vec{\mathbf{z}}_t - M\vec{\mathbf{x}}_t,\, C_m) \end{aligned}$$

where (for D-dimensional vector $\vec{\mathbf{x}}$)

$$G(\vec{\mathbf{x}}, C) = \frac{|C|^{-1/2}}{(2\pi)^{D/2}} \exp\left(\frac{-\vec{\mathbf{x}}^T C^{-1} \vec{\mathbf{x}}}{2} \right)$$

Kalman Filter

Remember

- product of two Gaussian densities is Gaussian
- marginalization of multi-dimensional Gaussian is also Gaussian

Key Result: Filtering and prediction distributions are Gaussian, so they may be represented by their respective means and covariances:

$$p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{z}}_{1:t-1}) = \int_{\vec{\mathbf{x}}_{t-1}} p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{x}}_{t-1})\, p(\vec{\mathbf{x}}_{t-1} \,|\, \vec{\mathbf{z}}_{1:t-1}) \;\sim\; N(\vec{\mathbf{x}}_t^-, C_t^-)$$

prediction mean

prediction covariance

$$p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{z}}_{1:t}) = c'\, p(\vec{\mathbf{z}}_t \,|\, \vec{\mathbf{x}}_t)\, p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{z}}_{1:t-1}) \;\sim\; N(\vec{\mathbf{x}}_t^+, C_t^+)$$

posterior mean

posterior covariance

Kalman Equations

Remember:

Convolution of two Gaussians with covariances C_1 and C_2 is Gaussian with covariance $C_1 + C_2$

If $\vec{\mathbf{x}} \sim N(\vec{\mathbf{0}}, C)$ and $\vec{\mathbf{y}} = A\vec{\mathbf{x}}$ then $\vec{\mathbf{y}} \sim N(\vec{\mathbf{0}}, A\,C\,A^T)$

So, with

$$p(\vec{\mathbf{x}}_{t-1} \,|\, \vec{\mathbf{z}}_{1:t-1}) \sim N(\vec{\mathbf{x}}^+_{t-1}, C^+_{t-1})$$

$$\vec{\mathbf{x}}_t = A\,\vec{\mathbf{x}}_{t-1} + \vec{\eta}_d \qquad \vec{\eta}_d \sim N(0, C_d)$$

it follows that the prediction mean and variance are given by

$$\begin{aligned} \vec{\mathbf{x}}^-_t &= A\,\vec{\mathbf{x}}^+_{t-1} \\ C^-_t &= A\,C^+_{t-1}A^T + C_d \end{aligned}$$

Kalman Equations

Combine prediction $\vec{\mathbf{x}}_t^-$, C_t^- with measurement $\vec{\mathbf{z}}_t$ to estimate the posterior mean and variance $\vec{\mathbf{x}}_t^+$, C_t^+:

Kalman gain:

$$K_t = C_{t-1}^- M^T \left(M C_{t-1}^- M^T + C_m \right)^{-1}$$

Posterior mean:

$$\vec{\mathbf{x}}_t^+ = \vec{\mathbf{x}}_t^- + K_t \boxed{\left(\vec{\mathbf{z}}_t - M \vec{\mathbf{x}}_t^- \right)}$$

innovation

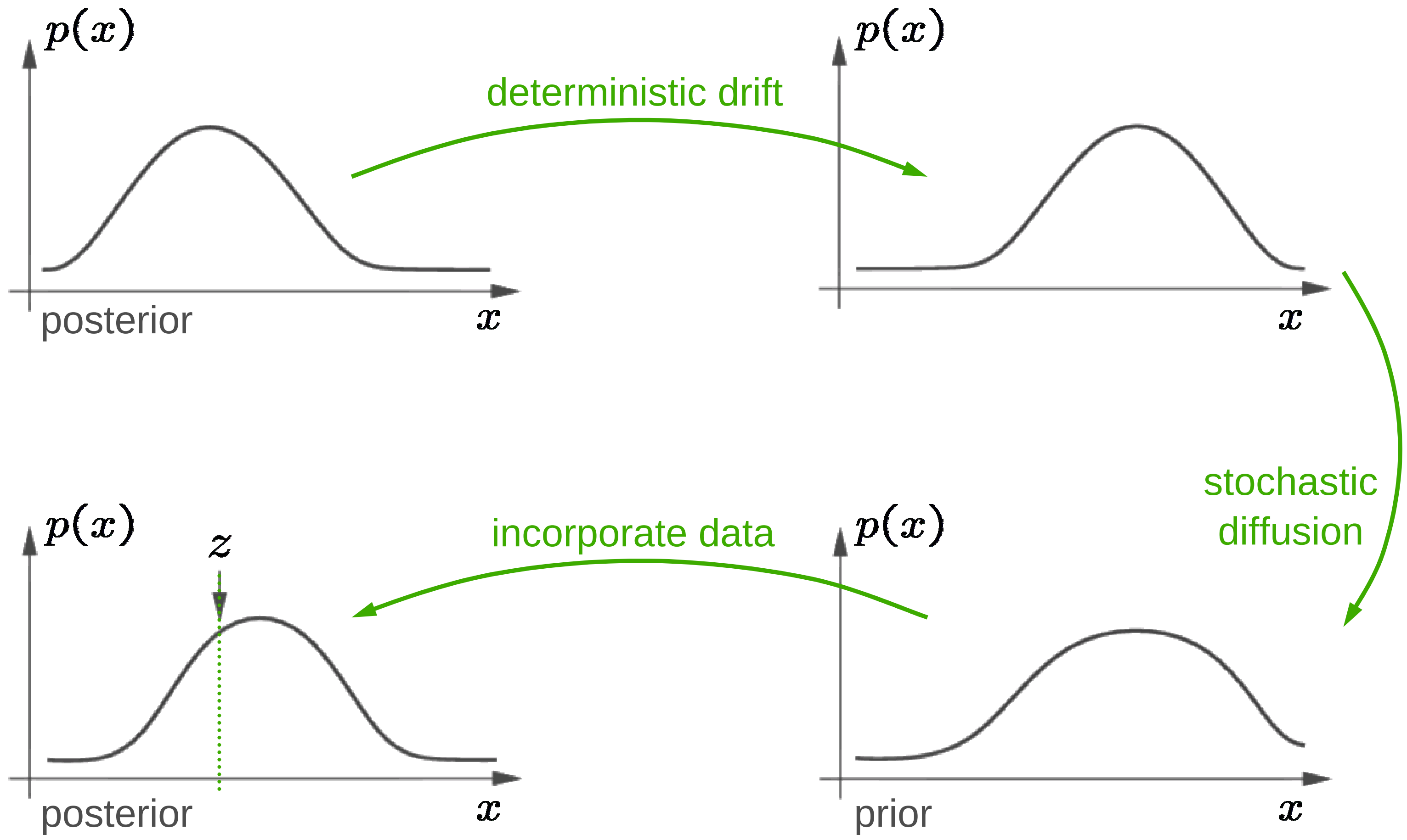
Posterior covariance:

$$\begin{aligned} C_t^+ &= (\mathbf{I} - K_t M)\, C_t^- \\ &= \boxed{(\mathbf{I} - K_t) C_t^- (\mathbf{I} - K_t)^T + K_t C_m K_t^T} \end{aligned}$$

numerically more stable

Identity matrix

Kalman Filter Cartoon

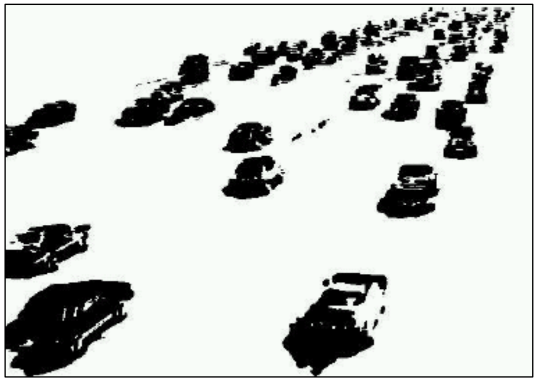


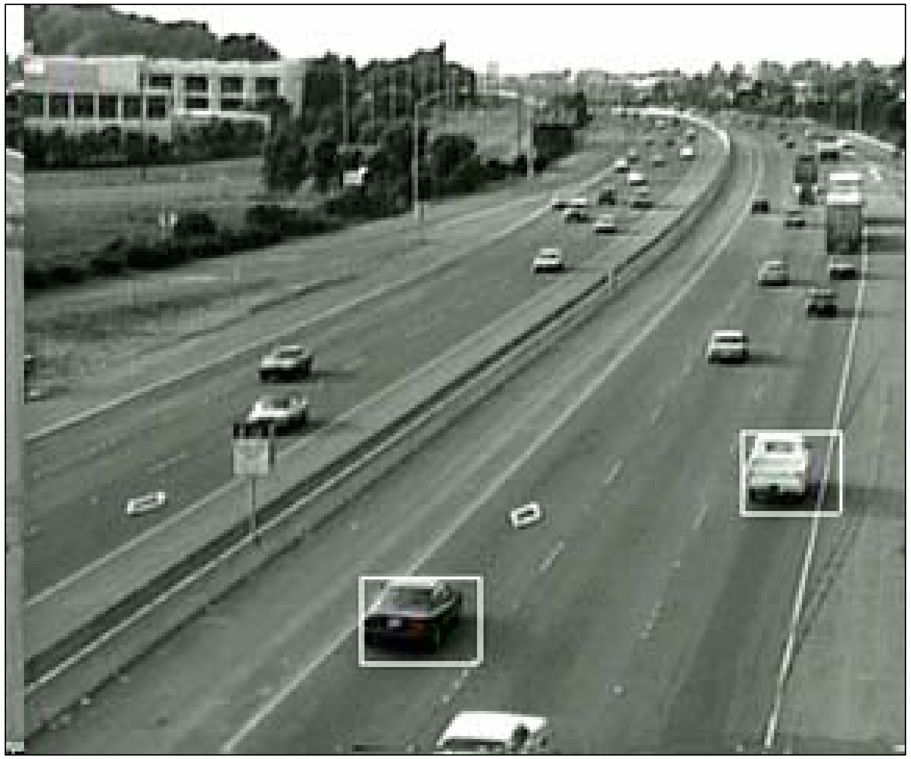
Kalman Filter

First well-known uses in computer vision:

- Road following by tracking lane markers
  [Dickmanns & Graefe, “Dynamic monocular machine vision.” Machine Vision and Applications, 1988]
- Rigid structure from feature tracks under perspective projection
  [Broida et al., “Recursive estimation of 3D motion from monocular image sequence. IEEE Trans. Aerosp. & Elec. Sys., 1990]

Car Tracking / Background Subtraction





[Koller, Weber & Malik, "Robust multiple car tracking with occlusion reasoning." Proc ECCV,1994]

Extended Kalman Filter

Problem: Non-linear dynamical systems.

Extended Kalman Filter (EKF): If system dynamics are nonlinear, then we can linearize the dynamics at estimated state at time t-1.

$$\begin{aligned} \vec{\mathbf{x}}_t &= f(\vec{\mathbf{x}}_{t-1}) + \vec{\eta}_d \\ &\approx A\,\vec{\mathbf{x}}_{t-1} + \vec{\eta}_d \end{aligned} \qquad \vec{\eta}_d \sim N(0, C_d)$$

where $A = \nabla f(\vec{\mathbf{x}})\,|_{\vec{\mathbf{x}}=\vec{\mathbf{x}}_{t-1}}$

Iterated, Extended Kalman Filter (IEKF): Iterate estimation with EKF, computing the Jacobian for each iteration at the most recent state estimate.

Remarks: computation of Jacobian, accumulated bias and under-estimation of variance.

Unscented Kalman Filter

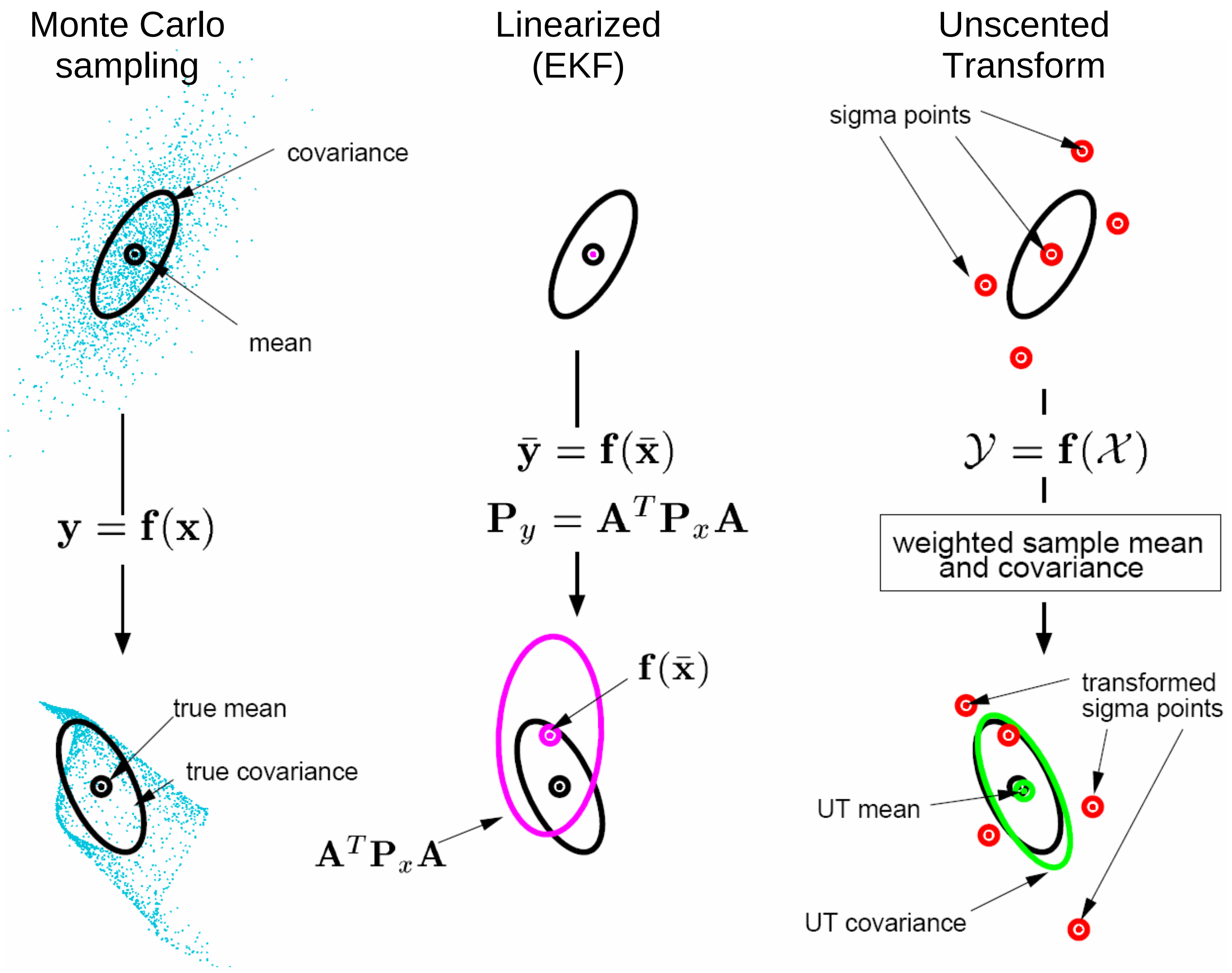
Problem: Errors in estimated mean and variance with EKF.

Solution: Use *unscented transform* to obtain mean and variance estimates to second-order with arbitrary non-linear dynamics [Julier & Uhlmann 97].

Sketch of Idea: Rather than approximate the state dynamics to ensure a Gaussian prediction distribution, using $A = \nabla f(\vec{\mathbf{x}})|_{\vec{\mathbf{x}}_t}$, use exact first and second moments of the prediction density under the nonlinear dynamics to obtain the Gaussian prediction.

- choose a set of *sigma points* $\vec{\mathbf{x}}_j$ whose sample mean & covariance given the mean and variance of the Gaussian posterior at t
- apply the dynamics to each sigma points: $\vec{\mathbf{y}}_j = f(\vec{\mathbf{x}}_j)$
- calculate the sample mean and covariances of points $\vec{\mathbf{y}}_j$

Unscented Transform



Monte Carlo sampling
covariance
mean
$\mathbf{y} = \mathbf{f}(\mathbf{x})$
true mean
true covariance
Linearized (EKF)
$\bar{\mathbf{y}} = \mathbf{f}(\bar{\mathbf{x}})$
$\mathbf{P}_y = \mathbf{A}^T\mathbf{P}_x\mathbf{A}$
$\mathbf{f}(\bar{\mathbf{x}})$
$\mathbf{A}^T\mathbf{P}_x\mathbf{A}$
Unscented Transform
sigma points
$\mathcal{Y} = \mathbf{f}(\mathcal{X})$
weighted sample mean and covariance
transformed sigma points
UT mean
UT covariance

Wan and van der Merwe, 2000

Data Association

Problem: Non-Gaussian and multi-modal likelihood functions, caused by the complex clutter that exists in most natural scenes.

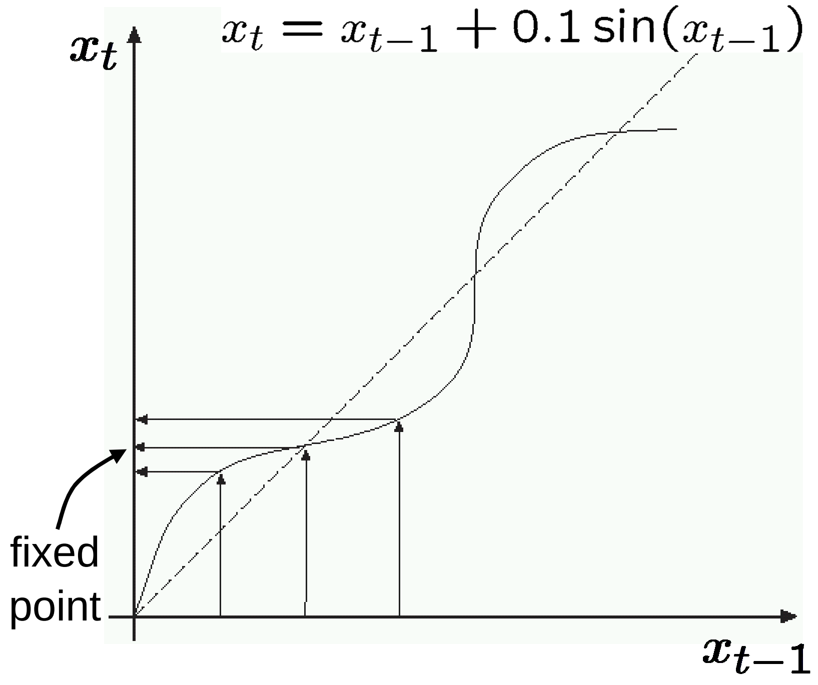
Data association: Select a subset of “measurements” from the entire collection of observations, i.e., those

- that are related to the object being tracked,
- for which simpler measurement models can be applied.

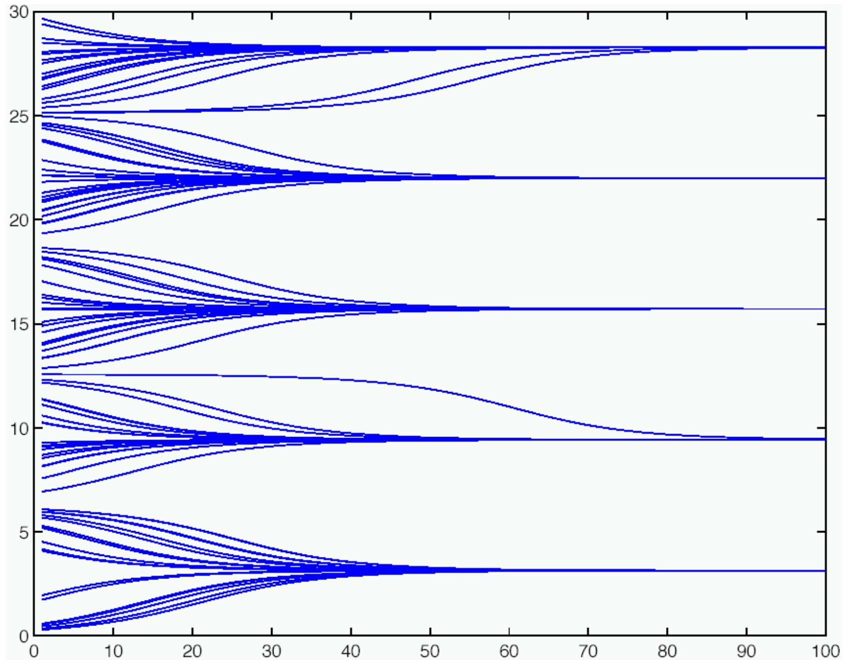
Probabilistic data association: let measurement be a weighted sum of measurements (weights given by likelihood of each measurement given the current model guess).

Remarks: do this when you don’t know how (or can’t be bothered) to model the rest of the scene (or background)

Problems: Nonlinear Dynamics



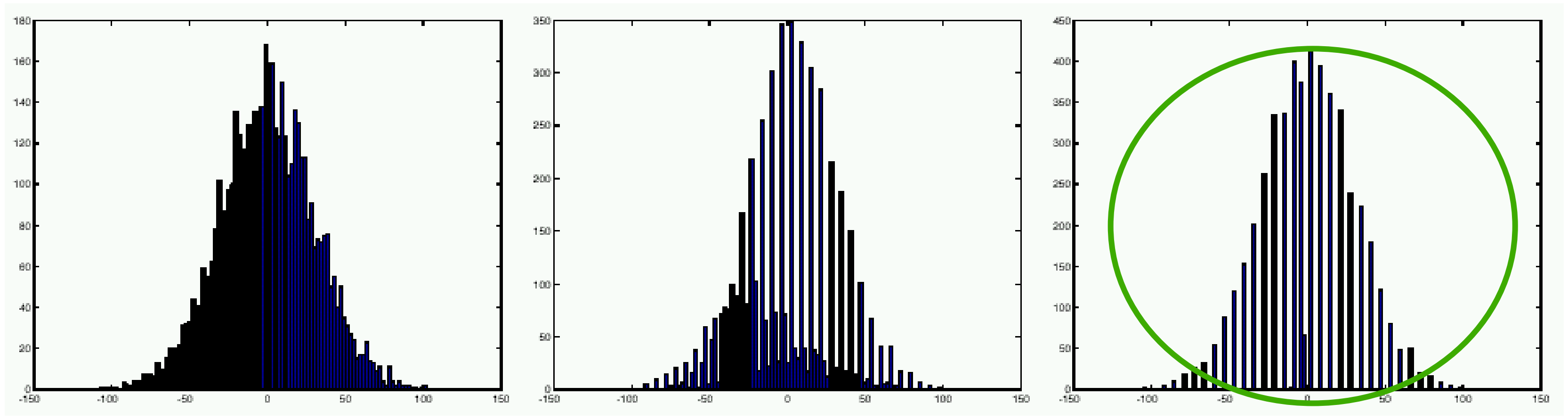
Dynamical model



State evolution over 100 time steps for different initial states

Problems: Nonlinear Dynamics

Multiple modes are poorly represented by a Gaussian



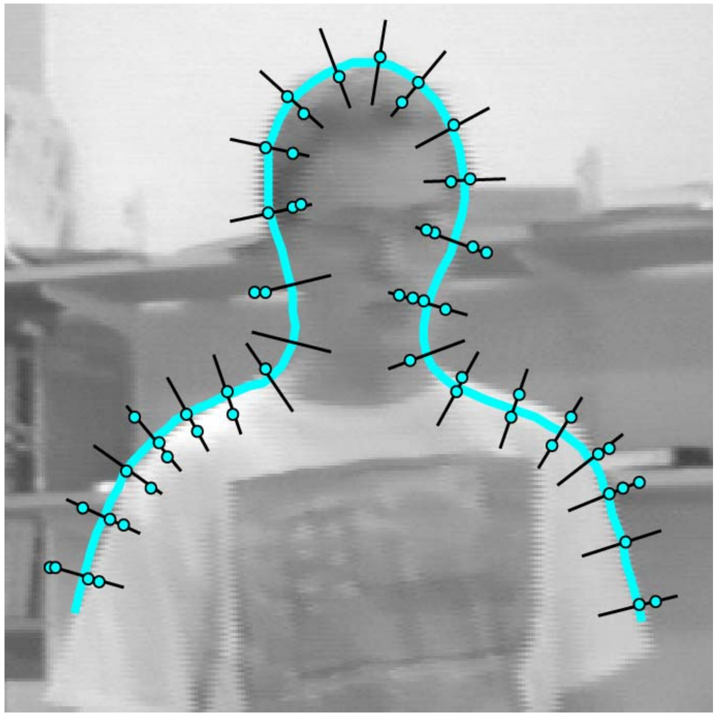
time = 0 time = 20 time = 70

Histograms of state values, as a function of time, given random Gaussian draws for the initial states

Problems: Multi-Modal Likelihoods

Except in specific domains, data association will not find only those measurements associated with the tracked object.

Measurement clutter in natural images causes likelihood functions to have multiple, local maxima.



Hill-Climbing (ML / MAP) Trackers

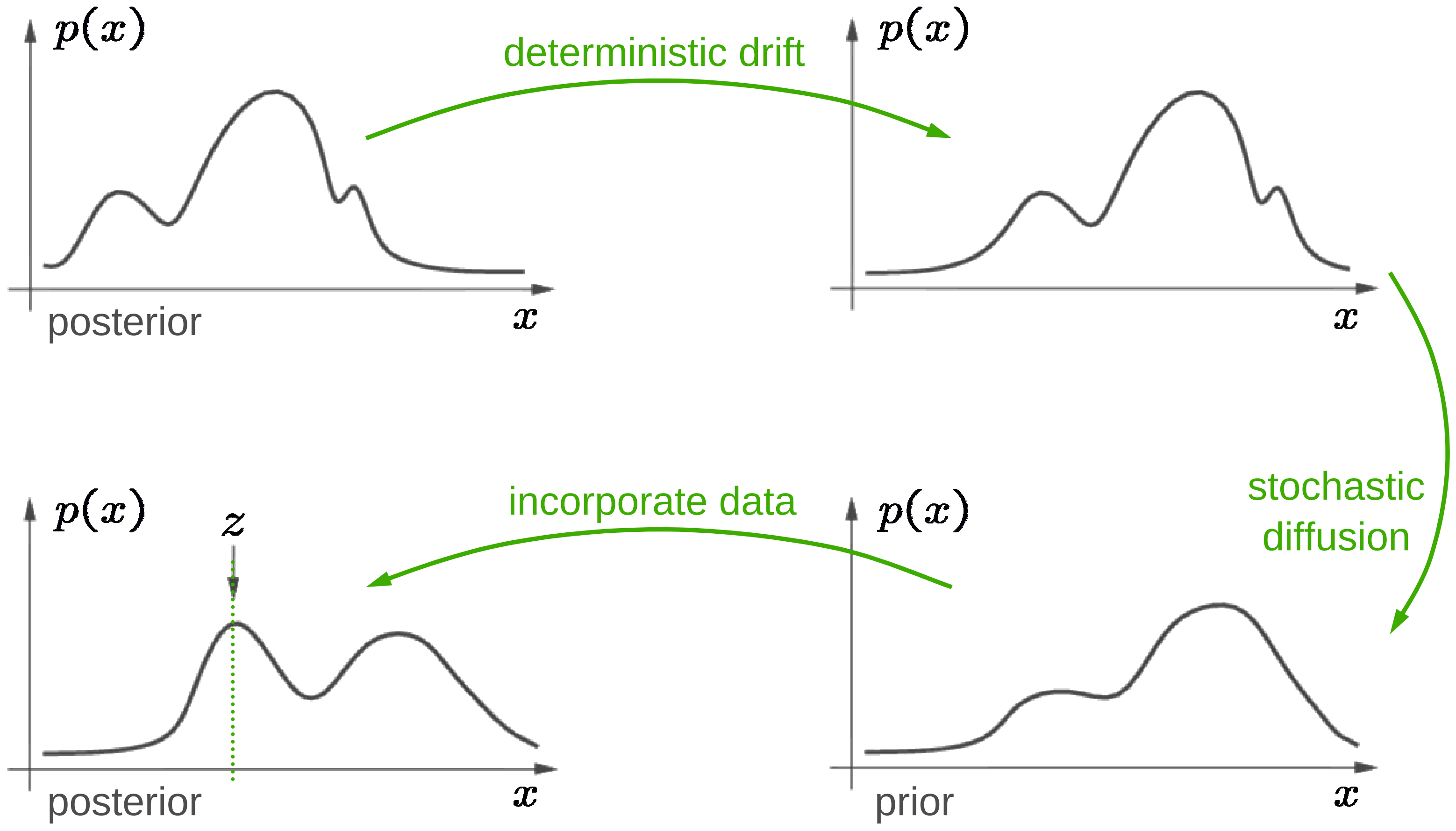
With non-Gaussian likelihoods, or nonlinear dynamics, one might forego computation of the full filtering distribution.

Instead, just estimate a maximal likelihood or posterior state; i.e.,

- do not propagate uncertainty
- use noiseless dynamical model to propagate the "optimal" state from the previous time to the current time
- then use iterative optimization (e.g. gradient ascent) from the initial guess to find local maxima of the likelihood or an approximate posterior

Remarks: works great when you are always close enough to the optimal (ML or MAP) state. This may be suitable for many apps, but tracking diagnostics and restarts may be required.

Bayesian Filtering with General Distributions



Non-Parametric Approximate Inference

Approximate the filtering distribution using point samples:

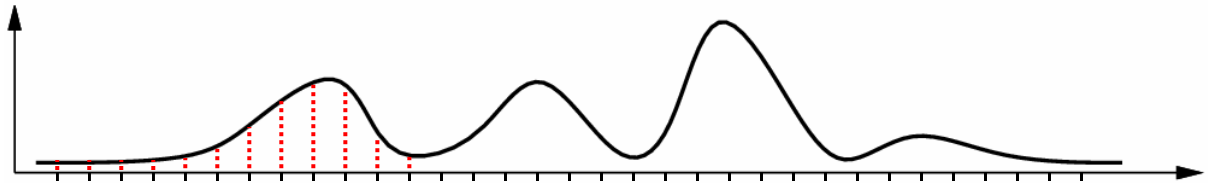
- By drawing a set of random samples from the filtering distribution, we could use samples statistics to approximate expectations

  Let $\mathcal{S} = \{\vec{\mathbf{x}}^{(j)}\}$ be a set of N fair samples from distribution $\mathcal{P}(\vec{\mathbf{x}})$, then for functions $f(\vec{\mathbf{x}})$

$$E_{\mathcal{S}}[f(\vec{\mathbf{x}})] \equiv \frac{1}{N}\sum_{j=1}^{N} f(\vec{\mathbf{x}}^{(j)}) \xrightarrow{N\to\infty} E_{\mathcal{P}}[f(\vec{\mathbf{x}})]$$

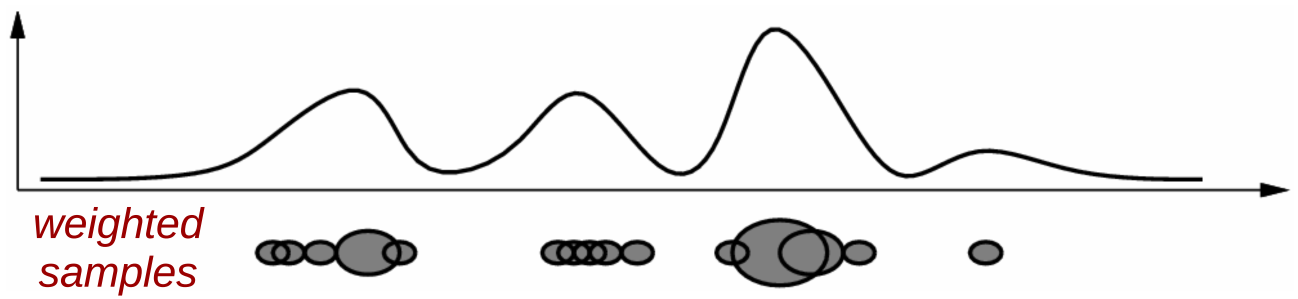
  Problem: we don't know how to draw samples from $p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{z}}_{1:t})$

- We could sample at regular intervals



  Problem: most samples have low probability – wasted computation

Importance Sampling



Weighted samples $\mathcal{S} = \{\vec{\mathbf{x}}^{(j)},\, w^{(j)}\}$

- Draw samples $\vec{\mathbf{x}}^{(j)}$ from a *proposal distribution* $\mathcal{Q}(\vec{\mathbf{x}})$
- Find weights $w^{(j)}$ so that the linearly weighted sample statistics approximate expectations under the desired distribution $\mathcal{P}(\vec{\mathbf{x}})$

$$E_{\mathcal{S}}[f(\vec{\mathbf{x}})] \equiv \sum_{j=1}^{N} w^{(j)} f(\vec{\mathbf{x}}^{(j)}) \xrightarrow{N\to\infty} E_{\mathcal{Q}}[w(\vec{\mathbf{x}})\, f(\vec{\mathbf{x}})]$$

$$= \int w(\vec{\mathbf{x}})\, f(\vec{\mathbf{x}})\, \mathcal{Q}(\vec{\mathbf{x}})\, d\vec{\mathbf{x}}$$

$$= \int f(\vec{\mathbf{x}})\, \mathcal{P}(\vec{\mathbf{x}})\, d\vec{\mathbf{x}}$$

$$= E_{\mathcal{P}}[f(\vec{\mathbf{x}})]$$

$$\boxed{\text{i.e. } w(\vec{\mathbf{x}}) = \frac{\mathcal{P}(\vec{\mathbf{x}})}{\mathcal{Q}(\vec{\mathbf{x}})}}$$

Particle Filter

Sequential Monte Carlo methods draw samples at each time step and then re-weight them to approximate the filtering distribution:

$$p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{z}}_{1:t}) \;=\; c\, p(\vec{\mathbf{z}}_t \,|\, \vec{\mathbf{x}}_t)\, p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{z}}_{1:t-1})$$

Simple particle filter:

- draw samples from the prediction distribution $p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{z}}_{1:t-1})$
- weights are proportional to the ratio of posterior and prediction distributions, i.e. the normalized likelihood $c\, p(\vec{\mathbf{z}}_t \,|\, \vec{\mathbf{x}}_t)$

$$\begin{tikzcd}
\underset{\text{posterior}}{p(\vec{\mathbf{x}}_{t-1} \,|\, \vec{\mathbf{z}}_{1:t-1})} \arrow[r, "\text{sample}"] & \underset{\text{temporal dynamics}}{p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{x}}_{t-1})} \arrow[r, "\text{sample}"] & \underset{\text{likelihood}}{p(\vec{\mathbf{z}}_t \,|\, \vec{\mathbf{x}}_t)} \arrow[r, "\text{normalize}"] & \underset{\text{posterior}}{p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{z}}_{1:t})}
\end{tikzcd}$$

[Gordon et al ’93; Isard & Blake ’98; Liu & Chen ’98, **…***]*

Particle Filter

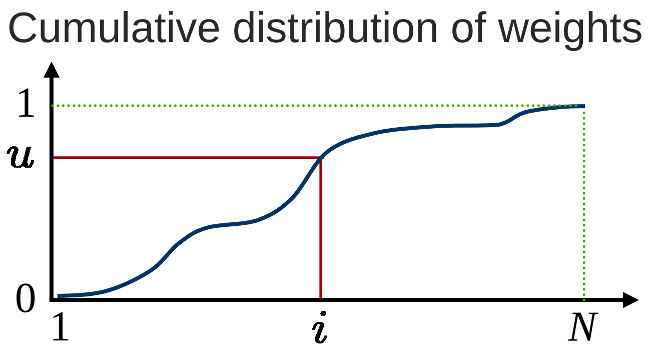
Given a weighted sample set $\mathcal{S}_{t-1} = \{\vec{\mathbf{x}}_{t-1}^{(j)}, w_{t-1}^{(j)}\}$

- the prediction distribution becomes a linear mixture model

$$p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{z}}_{1:t-1}) \;=\; \sum_{j=1}^{N} w^{(j)}\, p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{x}}_{t-1}^{(j)})$$

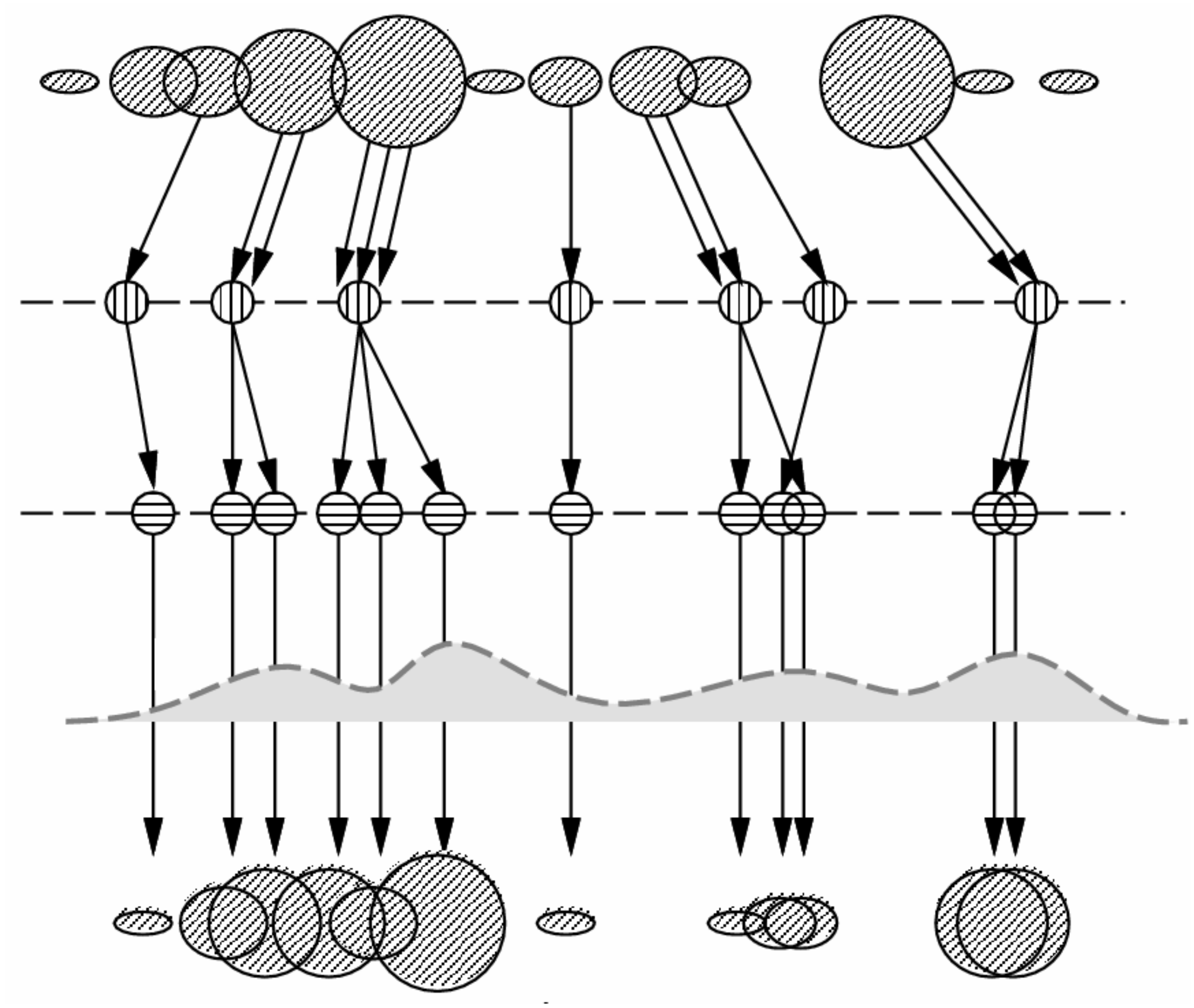
  - sample a component of the mixture by the treating weights as mixing probabilities

sample
$u \sim U(0,1)$



  - then sample from the associated dynamics pdf $p(\vec{\mathbf{x}}_t \,|\, \vec{\mathbf{x}}_{t-1}^{(i)})$

Particle Filter



weighted sample set

re-sample & drift

diffuse & re-sample

compute likelihoods

weighted sample set

Particle Filter

Some advantages of particle filters

- handle non-Gaussian and multi-modal distributions
- easy to implement
- manage computational cost by adjusting numbers of samples

Some hazards

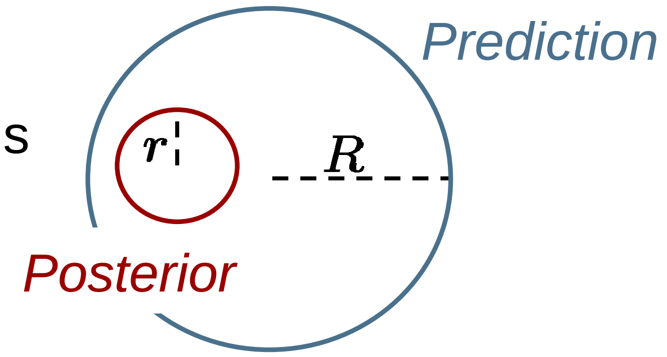
- proposal distribution should be non-zero wherever the posterior distribution is non-zero (usually heavy-tailed)
- proposals should be as good as possible, exploiting current observations in addition to prediction distribution
- sampling variability can be a problem
  - must have enough samples in regions of high probability for normalization to be useful
  - too many samples needed for high dimensional problems.
  - samples tend to migrate to a single mode

Particle Filter

Sample Size

- sample size depends on 'volumes' of prediction and posterior distributions
- so, number of particles should be exponential in state dimension, D:

$$N = (R/r)^D$$

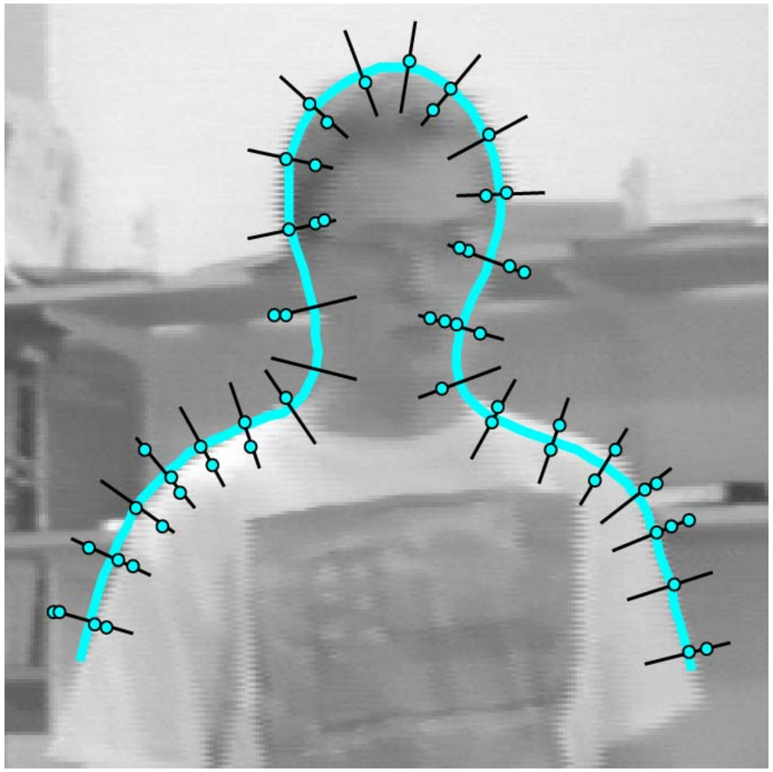


- effective number of 'independent' samples: $N_e = 1/\sum_j (w^{(j)})^2$

Sample Variability and Bias

- estimator variance can be a problem due to sampling
- re-sampling from poor approximations introduces bias over time (no need to re-sample every frame, eg. re-sample if N_e is small)
- better proposals minimize variance

Contour Tracking



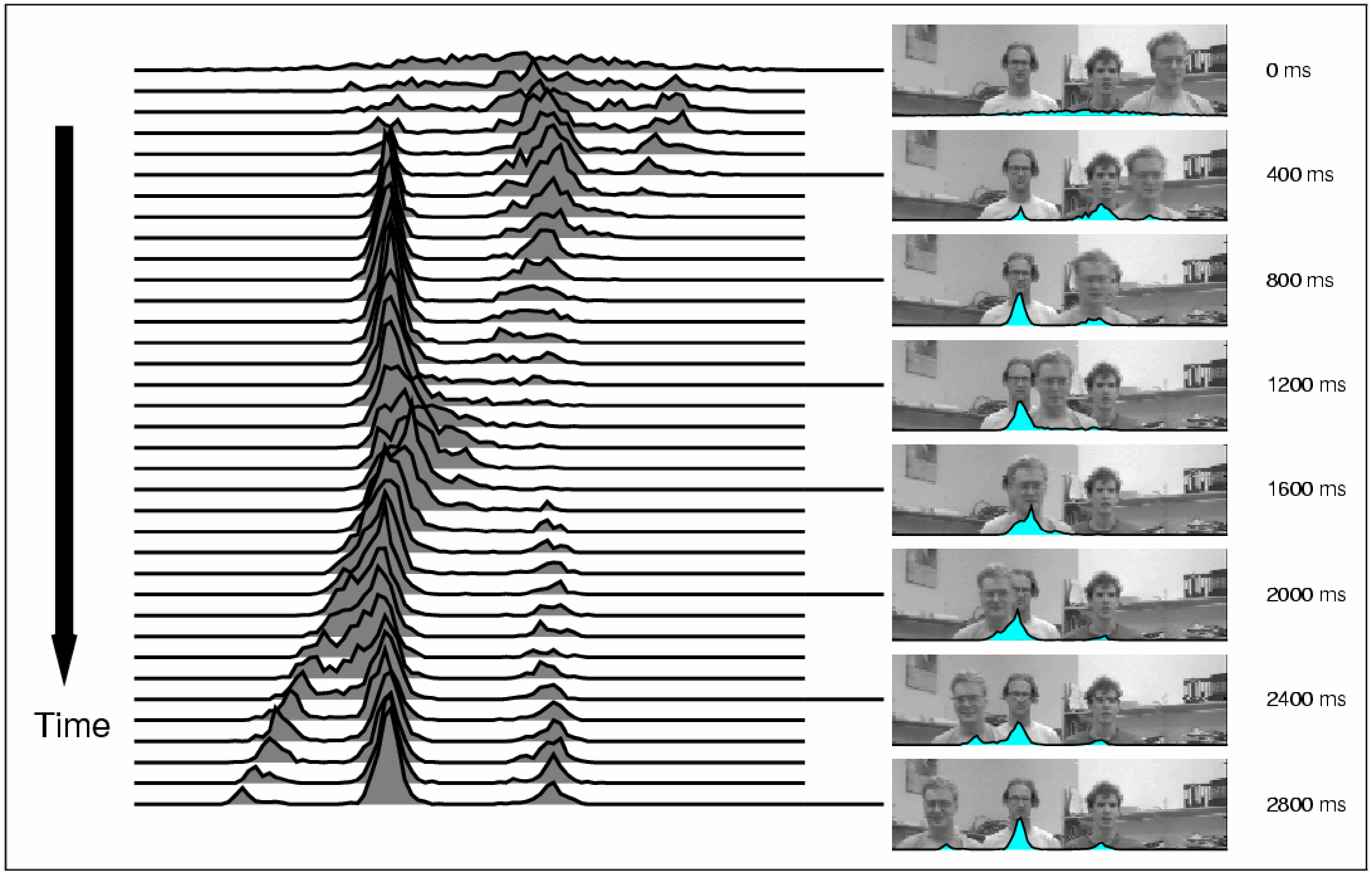
State: control points of spline-based contour representation

Measurements: edge strength perpendicular to contour

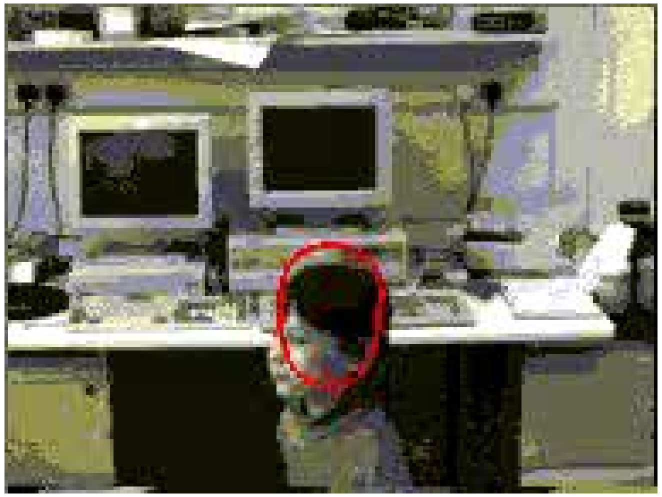
Dynamics: 2nd–order Markov (often learned)

[Isard & Blake, "Condensation - conditional density propagation for visual tracking." IJCV, 1998]

Contour Tracking



Contour Tracking



Tracking 3D People

3D Articulated Shape

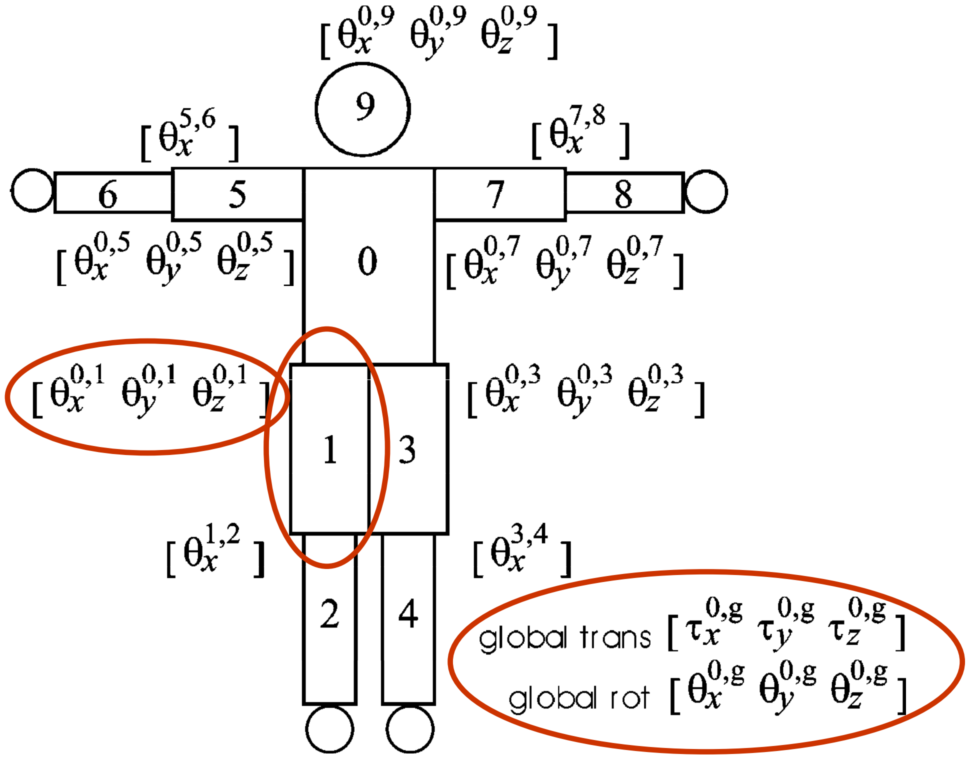
- ~10 parts (tapered cylinders)
- ~19 joint angles
- 6 pose/position variables

Parameters:

ϕ – shape vector $(\tau_x^{i,j}, \theta_x^{i,j})$

$\mathbf{V}$ – motion vector

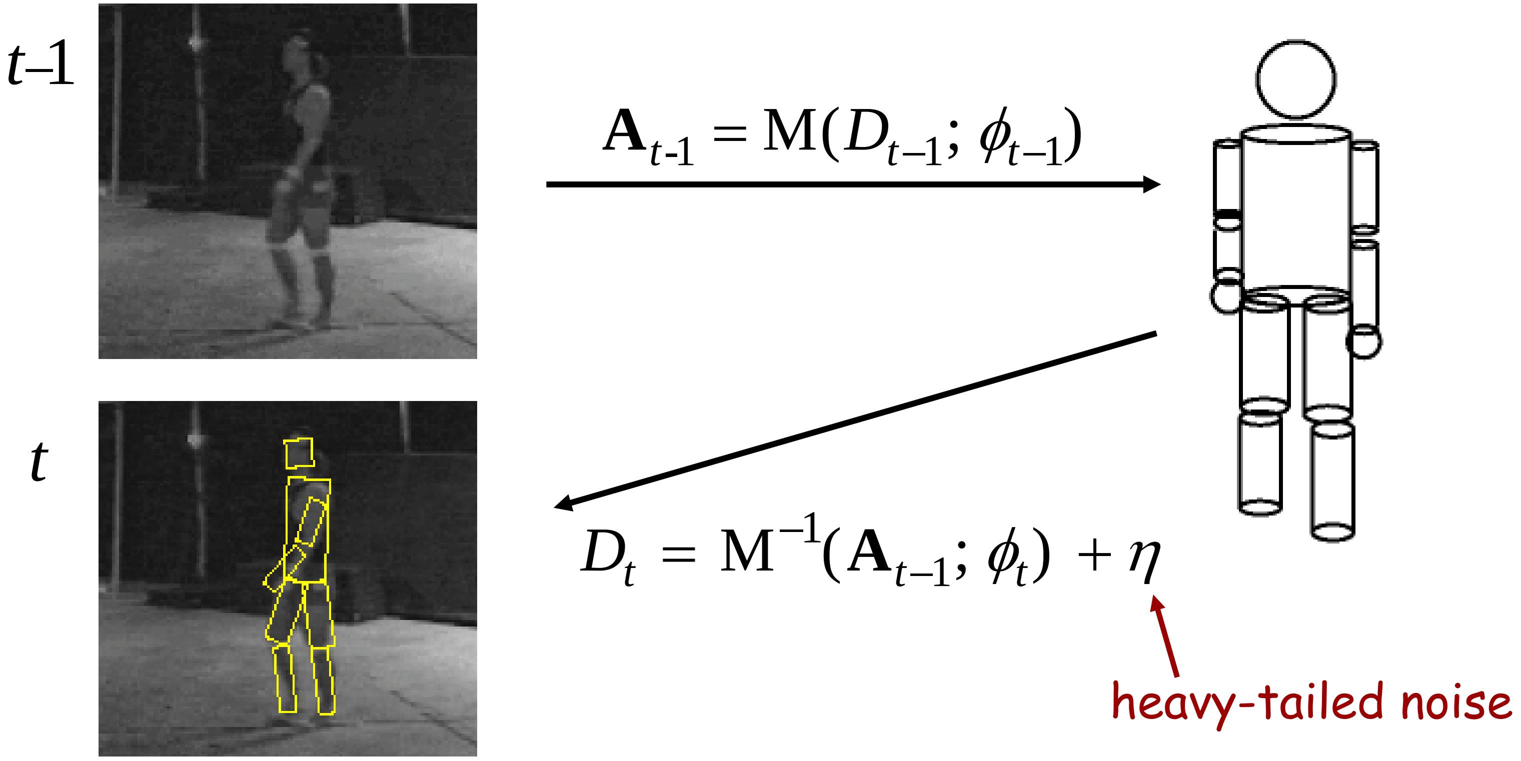
$\mathbf{A}$ – appearance model



$[\theta_x^{0,9}\ \theta_y^{0,9}\ \theta_z^{0,9}]$
9
$[\theta_x^{5,6}]$
$[\theta_x^{7,8}]$
6
5
7
8
0
$[\theta_x^{0,5}\ \theta_y^{0,5}\ \theta_z^{0,5}]$
$[\theta_x^{0,7}\ \theta_y^{0,7}\ \theta_z^{0,7}]$
$[\theta_x^{0,1}\ \theta_y^{0,1}\ \theta_z^{0,1}]$
$[\theta_x^{0,3}\ \theta_y^{0,3}\ \theta_z^{0,3}]$
1
3
$[\theta_x^{1,2}]$
$[\theta_x^{3,4}]$
2
4
global trans $[\tau_x^{0,g}\ \tau_y^{0,g}\ \tau_z^{0,g}]$
global rot $[\theta_x^{0,g}\ \theta_y^{0,g}\ \theta_z^{0,g}]$

[Sidenbladh, Black, & Fleet, 2000; Sidenbladh & Black, 2001; Poon & Fleet, 2002]

Motion Likelihood



$$\mathbf{A}_{t\text{-}1} = \mathrm{M}(D_{t-1};\, \phi_{t-1})$$

$$D_t = \mathrm{M}^{-1}(\mathbf{A}_{t-1};\, \phi_t) + \eta$$

Image formation: perspective projection of texture-mapped 3D shape (assumes brightness constancy and additive noise)

Motion Likelihood

Likelihood for visible limb *j*, is given by

$$L_j = \prod_{\boxed{\mathbf{x}\in\Re_j}} p_\eta\left(\boxed{D_t(\mathbf{x}) - \mathrm{M}^{-1}(\mathbf{A}_{t-1};\, \phi_t,\, \mathbf{x})},\ \sigma(\mathbf{x})\right)$$

To deal with hidden limbs, the limb likelihood is

$$p_j = \begin{cases} L_j & \text{if not occluded} \\ L_{occlude} & \text{if occluded} \end{cases}$$

where $L_{occlude}$ is constant

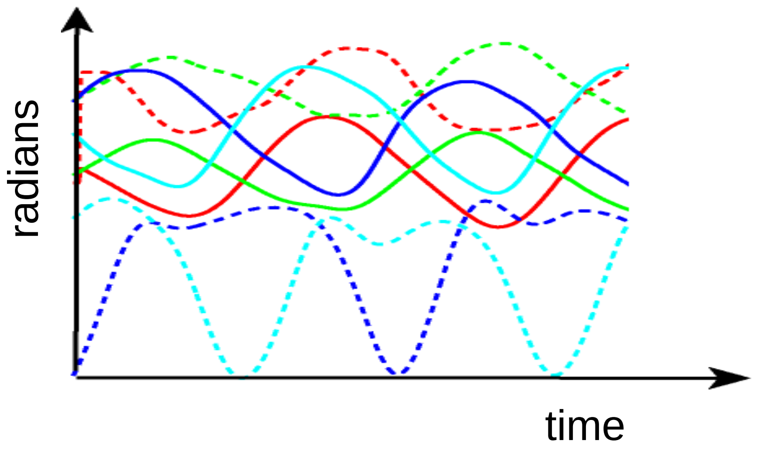
Independence of limb observations yields likelihood:

$$p(D_t \mid \phi_t, \mathbf{A}_{t-1}) = \prod_j p_j$$

Dynamics: Learned Walking Model

Activity-specific dynamics:

- constrain posterior to valid 3D human motions
- lower-dimensional state space simplifies estimation



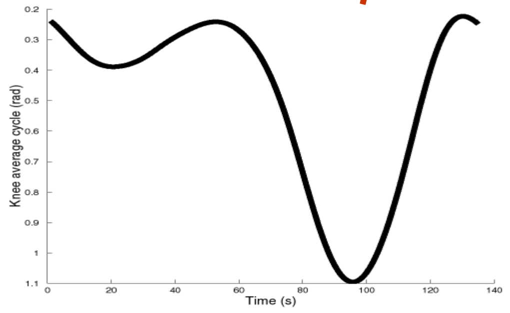
Joint angles versus time, for walking person obtained from a 3D motion-capture system

Dynamics: Learned Walking Model

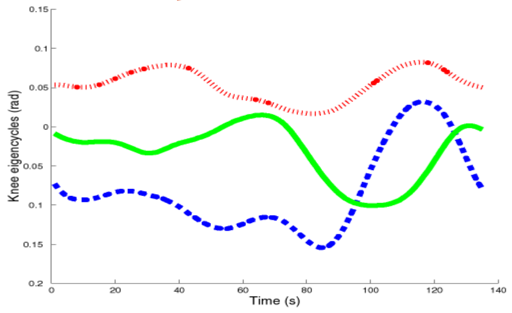
Joint angle curves are segmented and scaled to yield data curves $\{\theta_i(\psi)\}_{i=1}^{N}$ where $\psi \in [0,\ 1)$ is the phase the walking cycle.

PCA provides a linear basis for the joint angles at phase ψ_t :

$$\vec{\phi}(\vec{\mathbf{c}};\ \psi) \;=\; \vec{\mu}(\psi) \;+\; \sum_{k=1}^{B} c_k\, \vec{\mathbf{b}}_k(\psi)$$

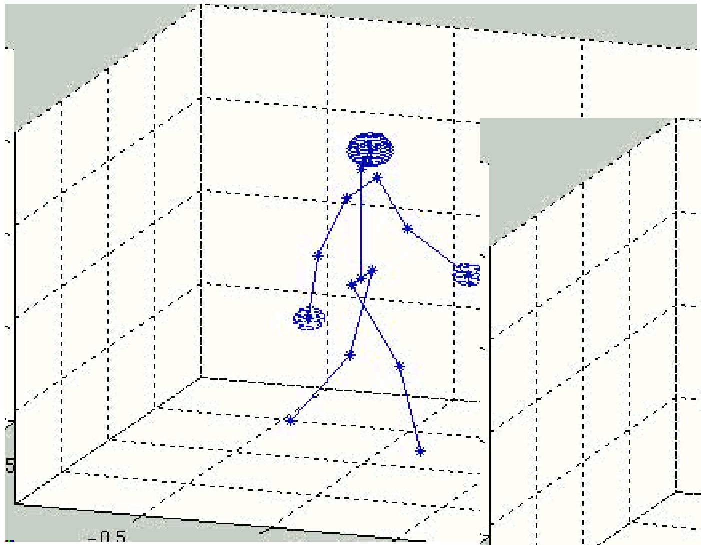


mean knee angle

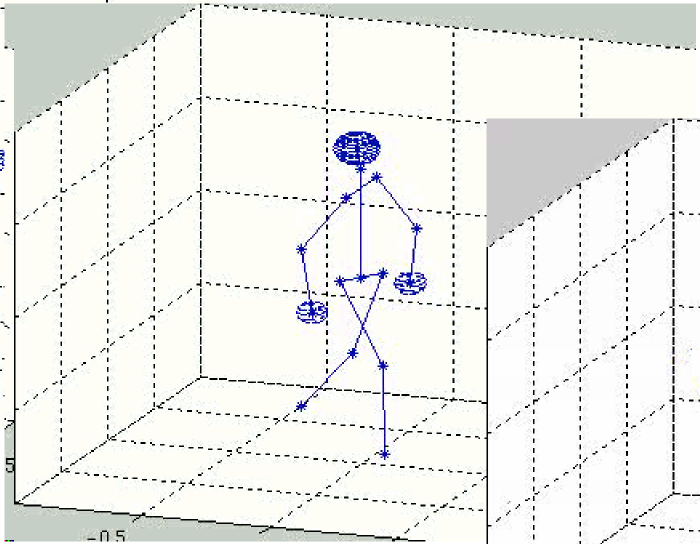


knee angle basis

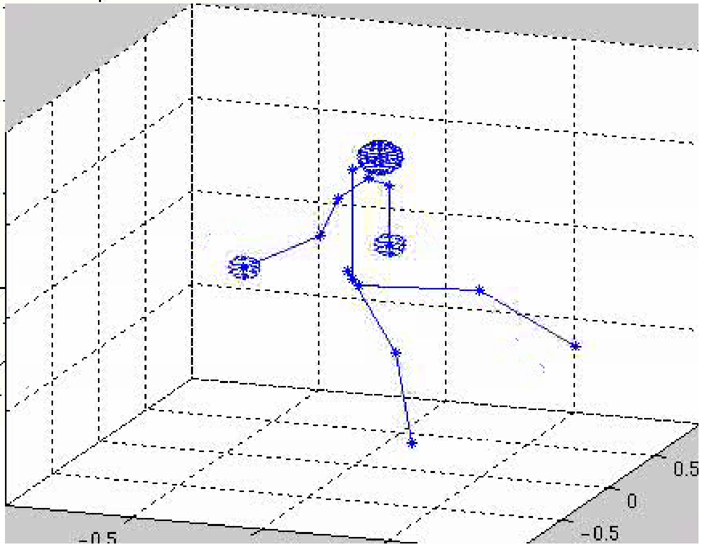
Temporal Dynamics: Walking Model



mean walking



mean walking plus
moderate noise



mean walking plus
large noise

Temporal Dynamics: Walking Model

Parameters of the generative model at time t :

$$\Big(\underset{\text{5 basis coefficients}}{\vec{\mathbf{c}}_t},\ \underset{\text{phase}}{\psi_t},\ \underset{\text{speed}}{\nu_t},\ \underbrace{\tau_t^g,\ \theta_t^g}_{\text{global pose}}\Big)$$

Smooth temporal dynamics (Gaussian process noise):

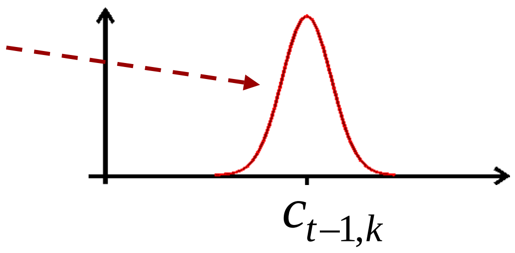
$$p(c_{t,k} \mid c_{t-1,k}) = G(c_{t,k} - c_{t-1,k}, \sigma_k^c)$$

$$p(v_t \mid v_{t-1}) = G(v_t - v_{t-1}, \sigma^v)$$

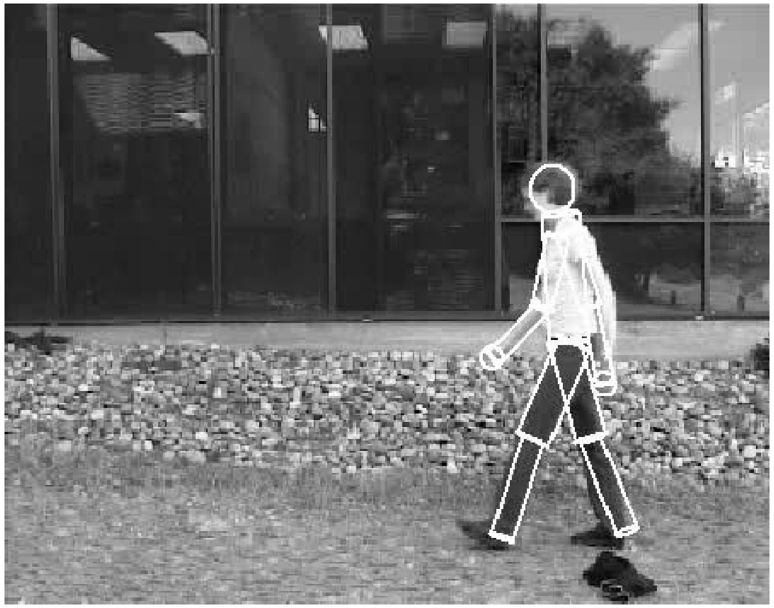
$$p(\psi_t \mid \psi_{t-1}) = G(\psi_t - \psi_{t-1} - v_{t-1}, \sigma^\psi)$$

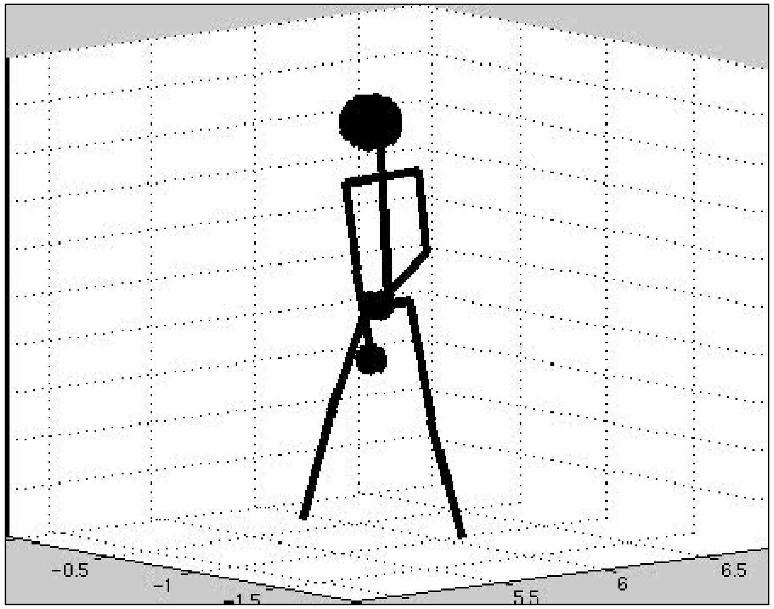
$$p(\tau_t^g \mid \tau_{t-1}^g, v_{t-1}^g) = G(\tau_t^g - \tau_{t-1}^g - v_{t-1}^g, \sigma^\tau)$$

$$p(\theta_t^g \mid \theta_{t-1}^g) = G(\theta_t^g - \theta_{t-1}^g, \sigma^\theta)$$



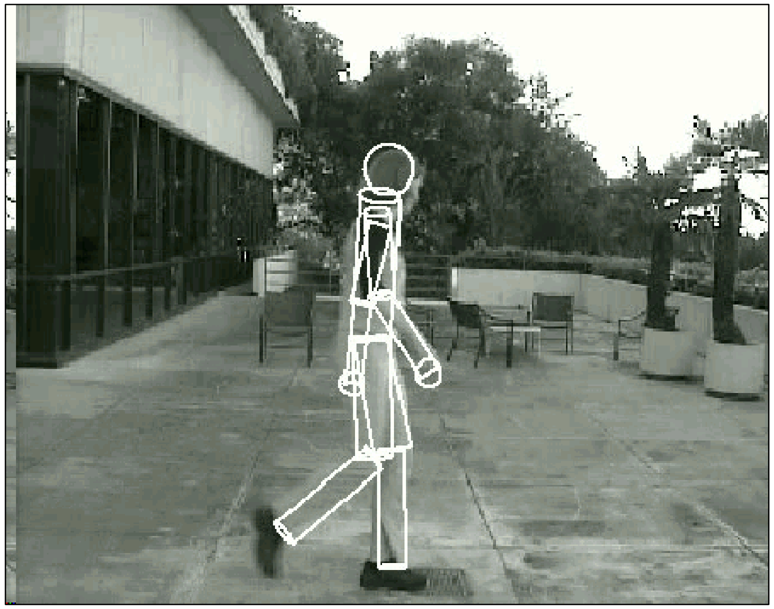
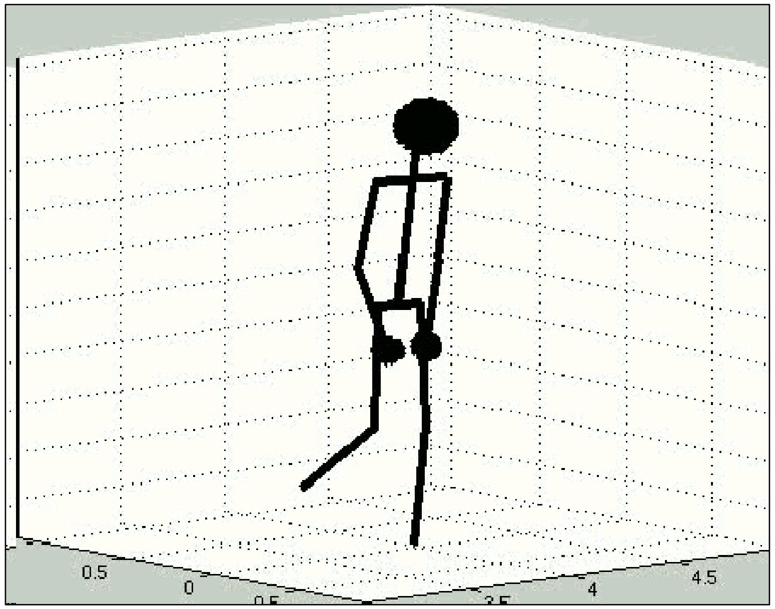
3D People Tracking





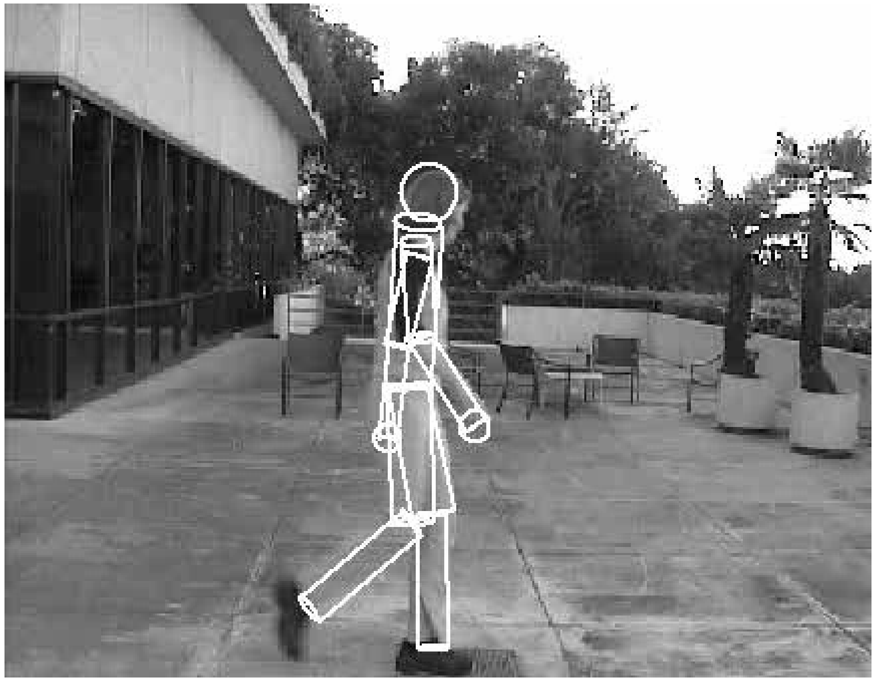
Mean posterior state shown from two viewpoints.
(manual initialization; 15,000 particles)

3D People Tracking

Mean posterior state shown from two viewpoints.
(note: significant change in orientation and depth)

3D People Tracking



Readings

- Forsyth & Ponce, Introduction to Computer Vision, Chapter, 17
- Isard & Blake, “Condensation: Conditional density propagation.” IJCV, 1998
- Shi & Tomasi, “Good features to track.” Proc. IEEE CVPR,1994
- Sidenbladh, Black & Fleet, “3D People tracking.” Proc ECCV, 2000
- Jepson, Fleet, and El-Maraghi, “Robust, on-line appearance models for visual tracking.” IEEE Trans on PAMI, 2003