6.869 Computer Vision

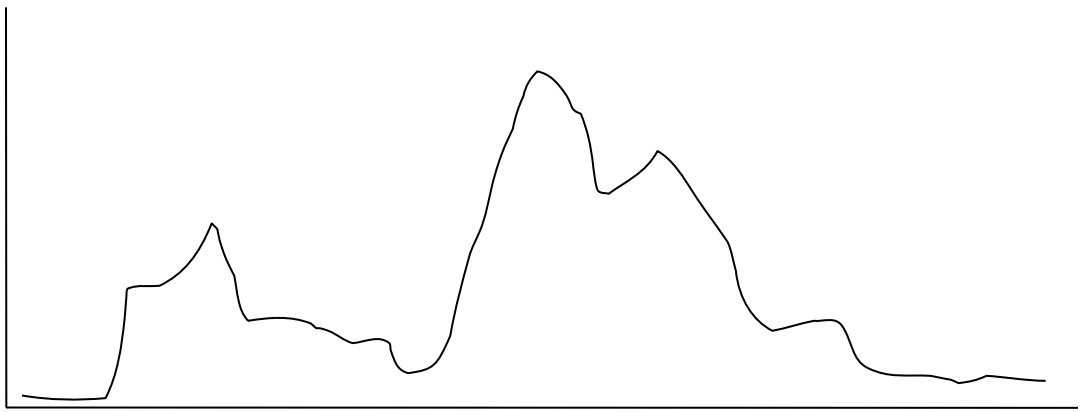
Recursive filtering for tracking--Kalman filtering and particle filtering.

April 11, 2011

Bill Freeman and Antonio Torralba

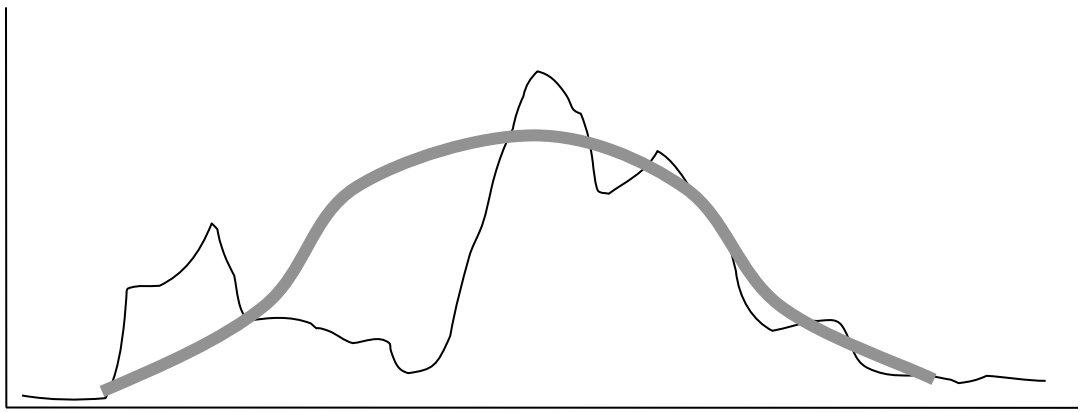
- See slide notes for material to be presented on board.

Representing non-linear Distributions



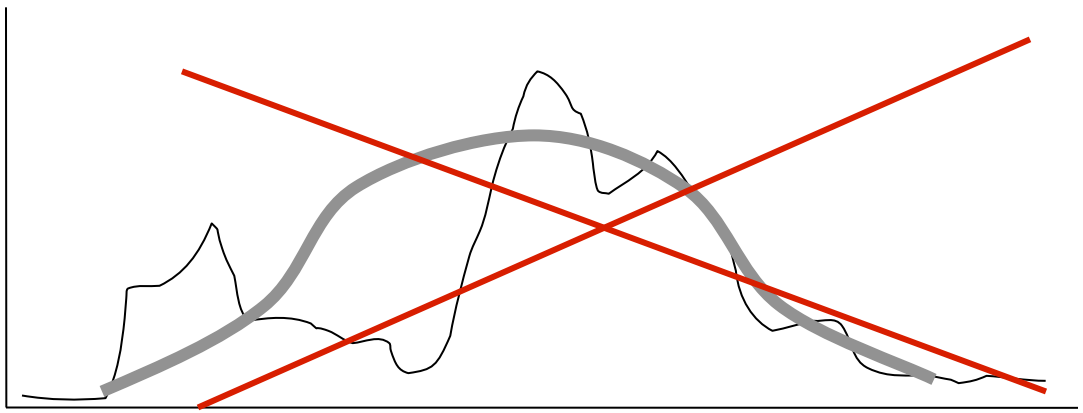
Representing non-linear Distributions

Unimodal parametric models fail to capture real-world densities…



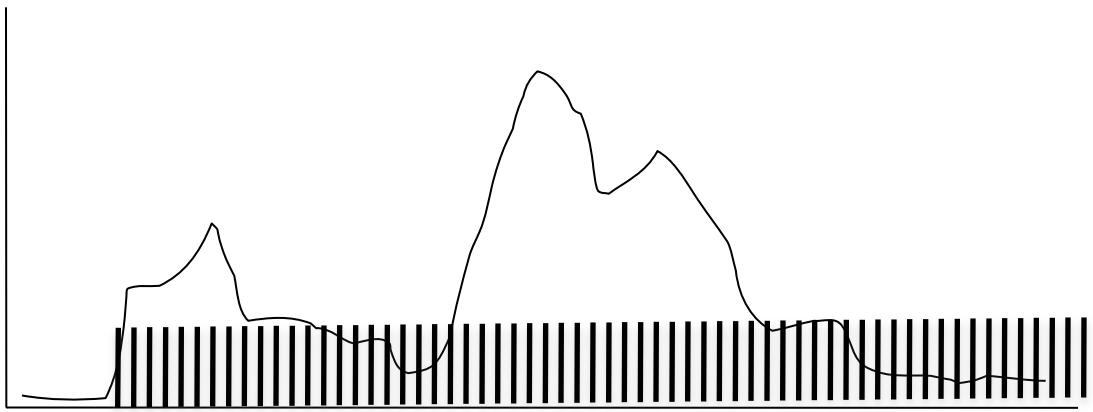
Representing non-linear Distributions

Unimodal parametric models fail to capture real-world densities…



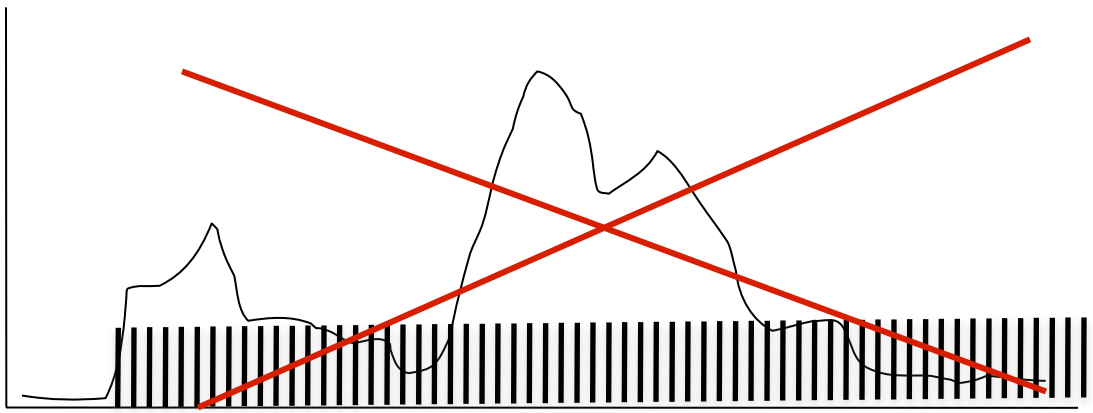
Discretize by evenly sampling over the entire state space

Tractable for 1-d problems like stereo, but not for high-dimensional problems.



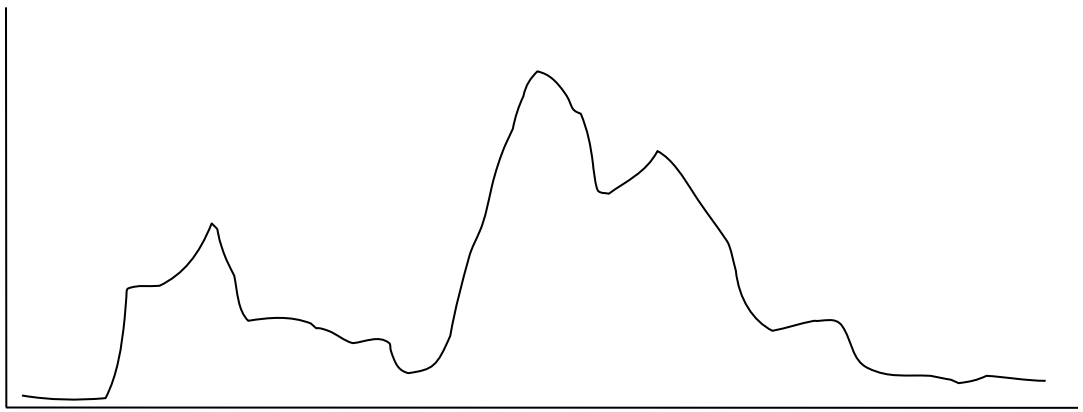
Discretize by evenly sampling over the entire state space

Tractable for 1-d problems like stereo, but not for high-dimensional problems.



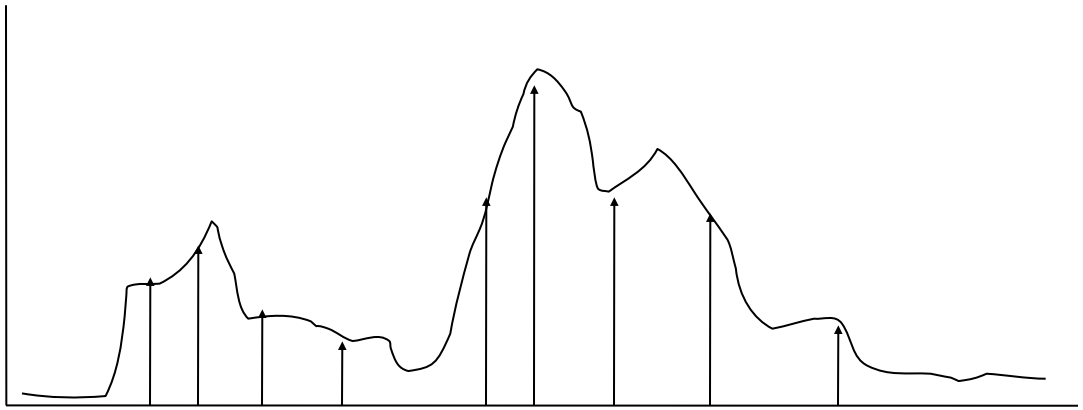
Representing Distributions using Weighted Samples

Rather than a parametric form, use a set of samples to represent a density:



Representing Distributions using Weighted Samples

Rather than a parametric form, use a set of samples to represent a density:



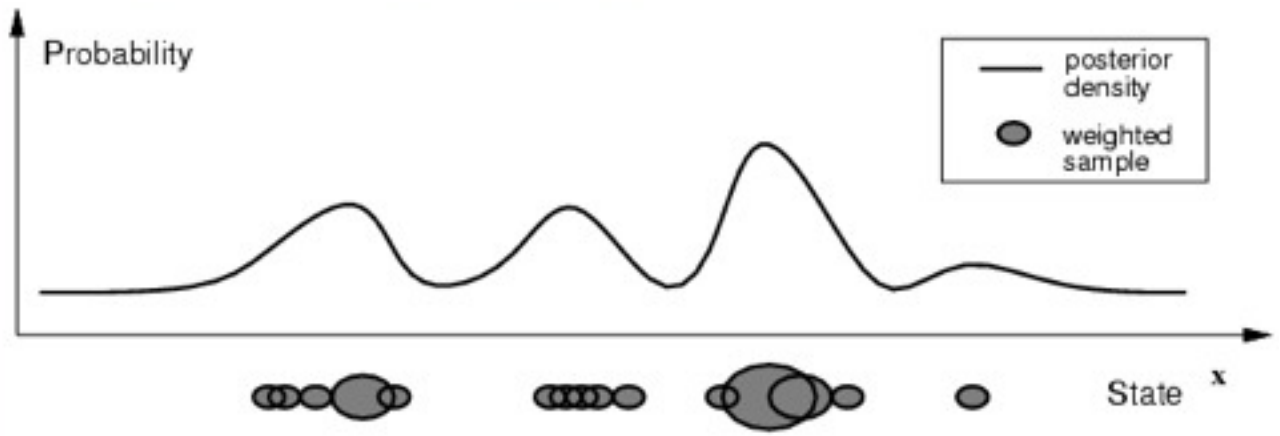
$$\{(\boldsymbol{u}^i, w^i)\}$$

Sample positions

Probability mass at each sample

This gives us two knobs to adjust when representing a probability density by samples: the locations of the samples, and the probability weight on each sample.

Representing distributions using weighted samples, another picture



[Isard 1998]

Sampled representation of a probability distribution

Represent a probability distribution

$$p_f(\boldsymbol{X}) = \frac{f(\boldsymbol{X})}{\int f(\boldsymbol{U}) d\boldsymbol{U}}$$

by a set of N weighted samples

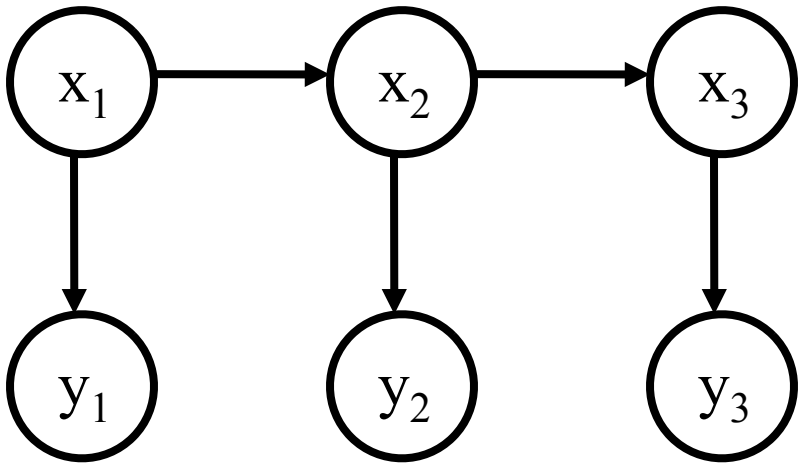
$$\{(\boldsymbol{u}^i, w^i)\}$$

where $\boldsymbol{u}^i \sim s(\boldsymbol{u})$ and $w^i = f(\boldsymbol{u}^i)/s(\boldsymbol{u}^i)$.

You can also think of this as a sum of dirac delta functions, each of weight w:

$$p_f(x) = \sum_i w^i \delta(x - u^i)$$

Tracking, in particle filter representation



$$p_f(x) = \sum_i w^i \delta(x - u^i)$$

$$P(x_n \mid y_1 \ldots y_n) = k\, P(y_n \mid x_n) \int dx_{n-1} P(x_n \mid x_{n-1}) P(x_{n-1} \mid y_1 \ldots y_{n-1})$$

Update step

Prediction step

Particle filter

- Let's apply this sampled probability density machinery to generalize the Kalman filtering framework.
- More general probability density representation than uni-modal Gaussian.
- Allows for general state dynamics, f(x) + noise

Sampled Prediction

$$P(\boldsymbol{x}_i|\boldsymbol{y}_0,\ldots,\boldsymbol{y}_{i-1}) \;=\; ?$$

Sampled Prediction

$$P(\boldsymbol{x}_i|\boldsymbol{y}_0,\ldots,\boldsymbol{y}_{i-1}) \;=\; ?$$

$$p(\boldsymbol{X}_{i-1}|\boldsymbol{y}_0,\ldots,\boldsymbol{y}_{i-1})$$

$$\left\{(\boldsymbol{u}_{i-1}^k, w_{i-1}^k)\right\} \longrightarrow \boldsymbol{x}_i = \boldsymbol{f}(\boldsymbol{x}_{i-1}) + \xi_i \longrightarrow$$

$$\left\{((f(\boldsymbol{u}_{i-1}^k) + \xi_i^l, \boldsymbol{u}_{i-1}^k), w_{i-1}^k)\right\}$$

$$p(\boldsymbol{X}_i, \boldsymbol{X}_{i-1}|\boldsymbol{y}_0,\ldots,\boldsymbol{y}_{i-1})$$

Sampled Prediction

$$P(\boldsymbol{x}_i|\boldsymbol{y}_0,\ldots,\boldsymbol{y}_{i-1}) \;=\; ?$$

$$p(\boldsymbol{X}_{i-1}|\boldsymbol{y}_0,\ldots,\boldsymbol{y}_{i-1})$$

$$\left\{(\boldsymbol{u}_{i-1}^k, w_{i-1}^k)\right\} \longrightarrow \boldsymbol{x}_i = \boldsymbol{f}(\boldsymbol{x}_{i-1}) + \xi_i \longrightarrow$$

$$\left\{((f(\boldsymbol{u}_{i-1}^k) + \xi_i^l, \boldsymbol{u}_{i-1}^k), w_{i-1}^k)\right\}$$

$$p(\boldsymbol{X}_i, \boldsymbol{X}_{i-1}|\boldsymbol{y}_0,\ldots,\boldsymbol{y}_{i-1})$$

Drop elements to marginalize to get

$$P(\boldsymbol{x}_i|\boldsymbol{y}_0,\ldots,\boldsymbol{y}_{i-1}) \;\sim=$$

$$\left\{(f(\boldsymbol{u}_{i-1}^k) + \xi_i^l, w_{i-1}^k)\right\}$$

Sampled Correction (Bayes rule)

Prior → posterior

Reweight every sample with the likelihood of the observations, given that sample:

$$p(\boldsymbol{Y}_i = \boldsymbol{y}_i | \boldsymbol{X}_i = \boldsymbol{s}_i^{k,-}) w_i^{k,-}$$

yielding a set of samples describing the probability distribution after the correction (update) step:

$$\left\{ (\boldsymbol{s}_i^{k,-}, p(\boldsymbol{Y}_i = \boldsymbol{y}_i | \boldsymbol{X}_i = \boldsymbol{s}_i^{k,-}) w_i^{k,-}) \right\}$$

Naïve PF Tracking

- Start with samples from something simple (Gaussian)
- Repeat

Naïve PF Tracking

- Start with samples from something simple (Gaussian)
- Repeat
  - Correct

Naïve PF Tracking

- Start with samples from something simple (Gaussian)
- Repeat
  - Correct

$$\left\{(\boldsymbol{s}_i^{k,-}, p(\boldsymbol{Y}_i = \boldsymbol{y}_i | \boldsymbol{X}_i = \boldsymbol{s}_i^{k,-}) w_i^{k,-})\right\}$$

Naïve PF Tracking

- Start with samples from something simple (Gaussian)
- Repeat
  - Correct

Take each particle from the prediction step and modify the old weight by multiplying by the new likelihood

$$\left\{(\boldsymbol{s}_i^{k,-}, p(\boldsymbol{Y}_i = \boldsymbol{y}_i | \boldsymbol{X}_i = \boldsymbol{s}_i^{k,-}) w_i^{k,-})\right\}$$

Naïve PF Tracking

- Start with samples from something simple (Gaussian)
- Repeat
  - Correct

Take each particle from the prediction step and modify the old weight by multiplying by the new likelihood

$$\left\{(\boldsymbol{s}_i^{k,-}, p(\boldsymbol{Y}_i = \boldsymbol{y}_i | \boldsymbol{X}_i = \boldsymbol{s}_i^{k,-}) w_i^{k,-})\right\}$$

  - Predict

Naïve PF Tracking

- Start with samples from something simple (Gaussian)
- Repeat
  - Correct

Take each particle from the prediction step and modify the old weight by multiplying by the new likelihood

$$\left\{(\boldsymbol{s}_i^{k,-}, p(\boldsymbol{Y}_i = \boldsymbol{y}_i | \boldsymbol{X}_i = \boldsymbol{s}_i^{k,-}) w_i^{k,-})\right\}$$

  - Predict

$$\left\{(f(\mathbf{s}_{i-1}^{k}) + \xi_i^l, w_{i-1}^{k})\right\}$$

Naïve PF Tracking

- Start with samples from something simple (Gaussian)
- Repeat
  - Correct

Take each particle from the prediction step and modify the old weight by multiplying by the new likelihood

$$\left\{(\boldsymbol{s}_i^{k,-}, p(\boldsymbol{Y}_i = \boldsymbol{y}_i | \boldsymbol{X}_i = \boldsymbol{s}_i^{k,-}) w_i^{k,-})\right\}$$

  - Predict

$$\left\{(f(\mathbf{s}_{i-1}^{k}) + \xi_i^l, w_{i-1}^k)\right\}$$

Run every particle through the dynamics function and add noise.

Naïve PF Tracking

- Start with samples from something simple (Gaussian)
- Repeat
  - Correct

Take each particle from the prediction step and modify the old weight by multiplying by the new likelihood

$$\left\{(\boldsymbol{s}_i^{k,-}, p(\boldsymbol{Y}_i = \boldsymbol{y}_i | \boldsymbol{X}_i = \boldsymbol{s}_i^{k,-}) w_i^{k,-})\right\}$$

  - Predict

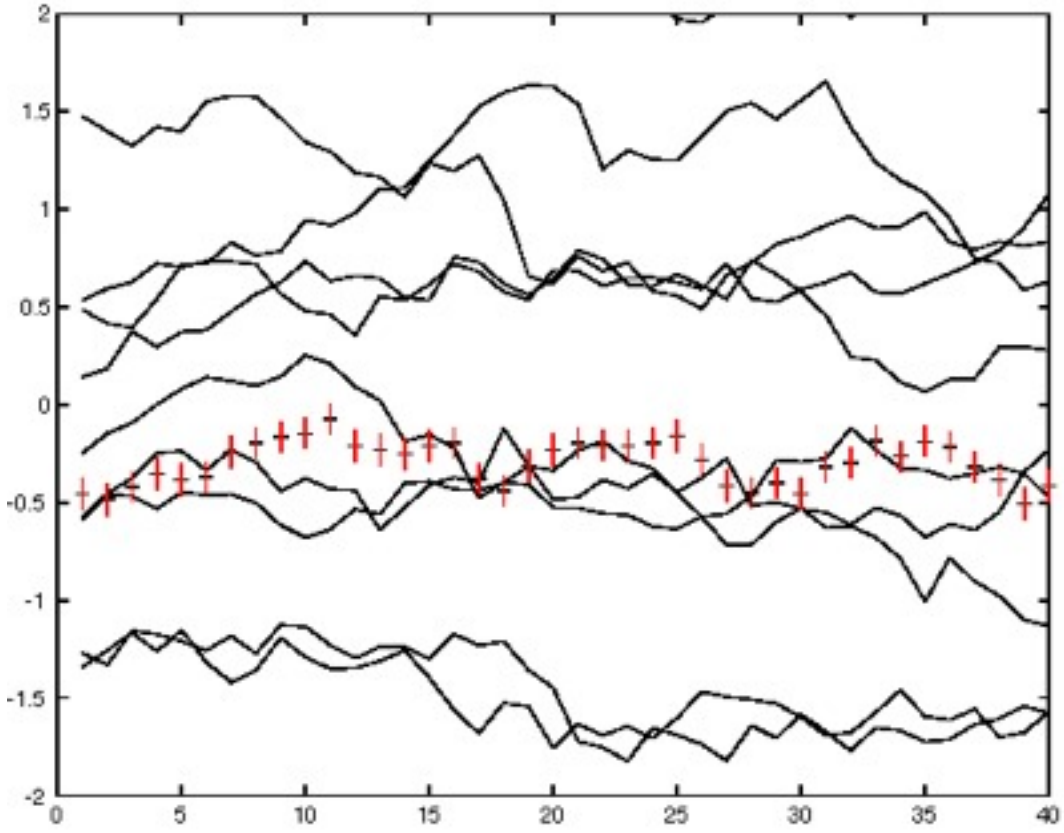
$$\left\{(f(\mathbf{s}_{i-1}^{k}) + \xi_i^l, w_{i-1}^k)\right\}$$

Run every particle through the dynamics function and add noise.

But doesn't work that well because of sample impoverishment…

Sample impoverishment

10 of the 100 particles, along with the true Kalman filter track, with variance:



Resample the prior

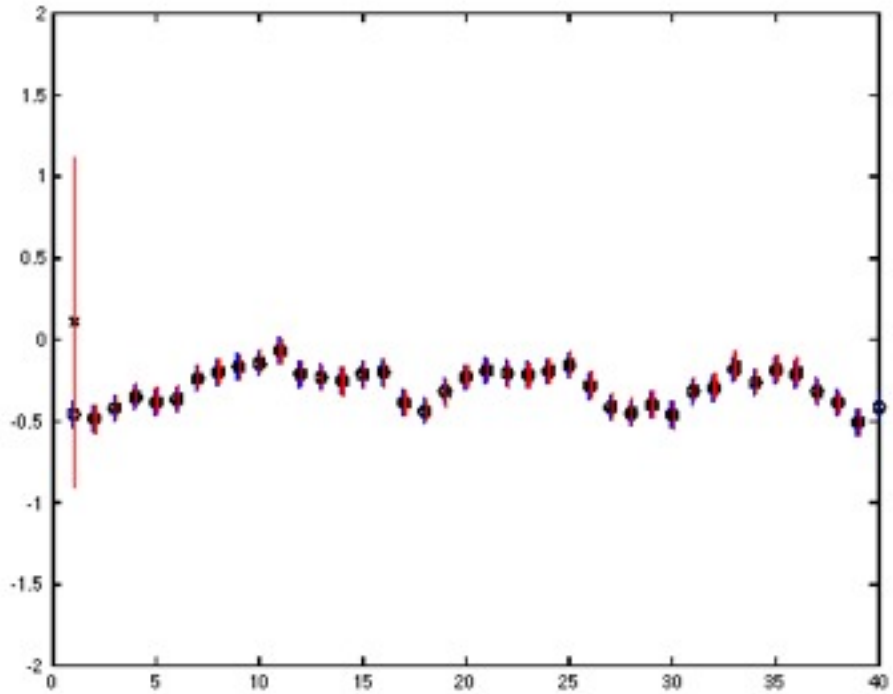
In a sampled density representation, the frequency of samples can be traded off against weight:

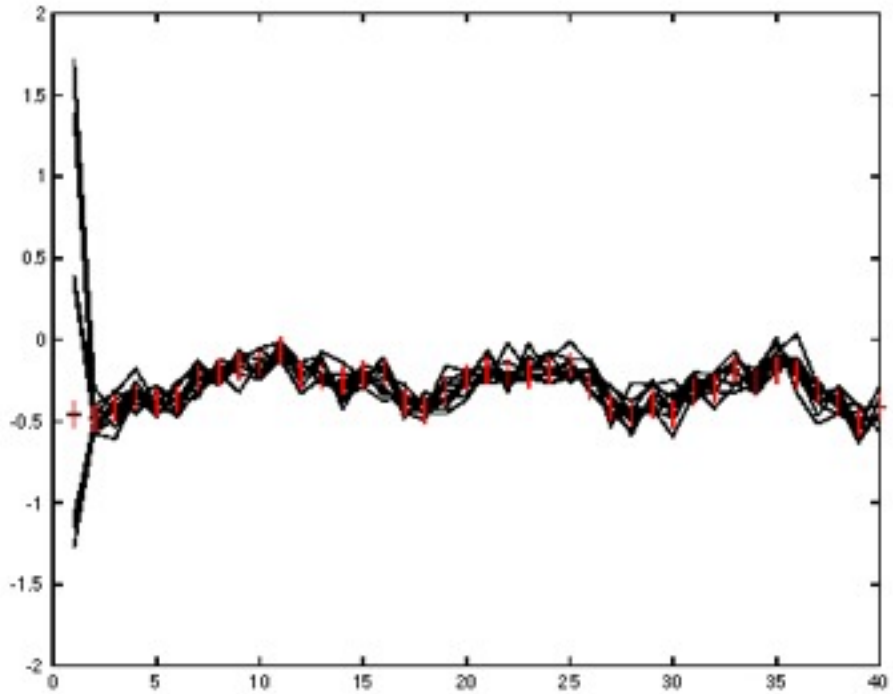
$$(\boldsymbol{s}_k, w_k) \longrightarrow \begin{matrix}(\boldsymbol{s}_k, 1)\\ (\boldsymbol{s}_k, 1)\\ (\boldsymbol{s}_k, 1)\\ \vdots\end{matrix} \; N_k \text{ copies} \quad \text{s.t.} \quad \frac{N_k}{\sum_k N_k} = w_k$$

These new samples are a representation of the same density.

I.e., make N draws with replacement from the original set of samples, using the weights as the probability of drawing a sample.

Resampling concentrates samples





A practical particle filter with resampling

Initialization: Represent $P(X_0)$ by a set of N samples

$$\left\{(s_0^{k,-}, w_0^{k,-})\right\}$$

where

$$s_0^{k,-} \sim P_s(S) \text{ and } w_0^{k,-} = P(s_0^{k,-})/P_s(S = s_0^{k,-})$$

Ideally, $P(X_0)$ has a simple form and $s_0^{k,-} \sim P(X_0)$ and $w_0^{k,-} = 1$.

Prediction: Represent $P(X_i|y_0, y_{i-1})$ by

$$\left\{(s_i^{k,-}, w_i^{k,-})\right\}$$

where

$$s_i^{k,-} = f(s_{i-1}^{k,+}) + \xi_i^k \text{ and } w_i^{k,-} = w_{i-1}^{k,+} \text{ and } \xi_i^k \sim N(0, \Sigma_{d_i})$$

Correction: Represent $P(X_i|y_0, y_i)$ by

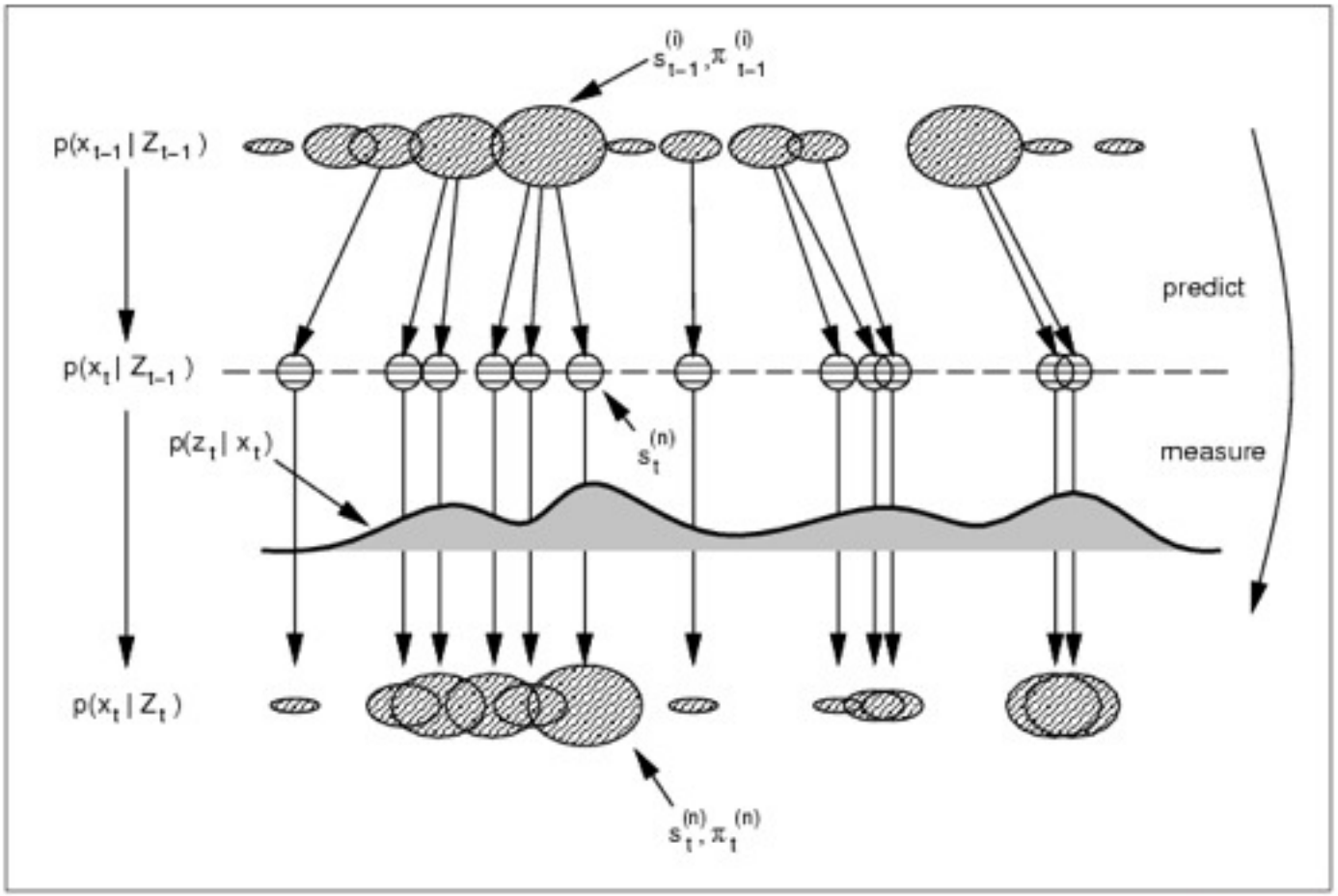
$$\left\{(s_i^{k,+}, w_i^{k,+})\right\}$$

where

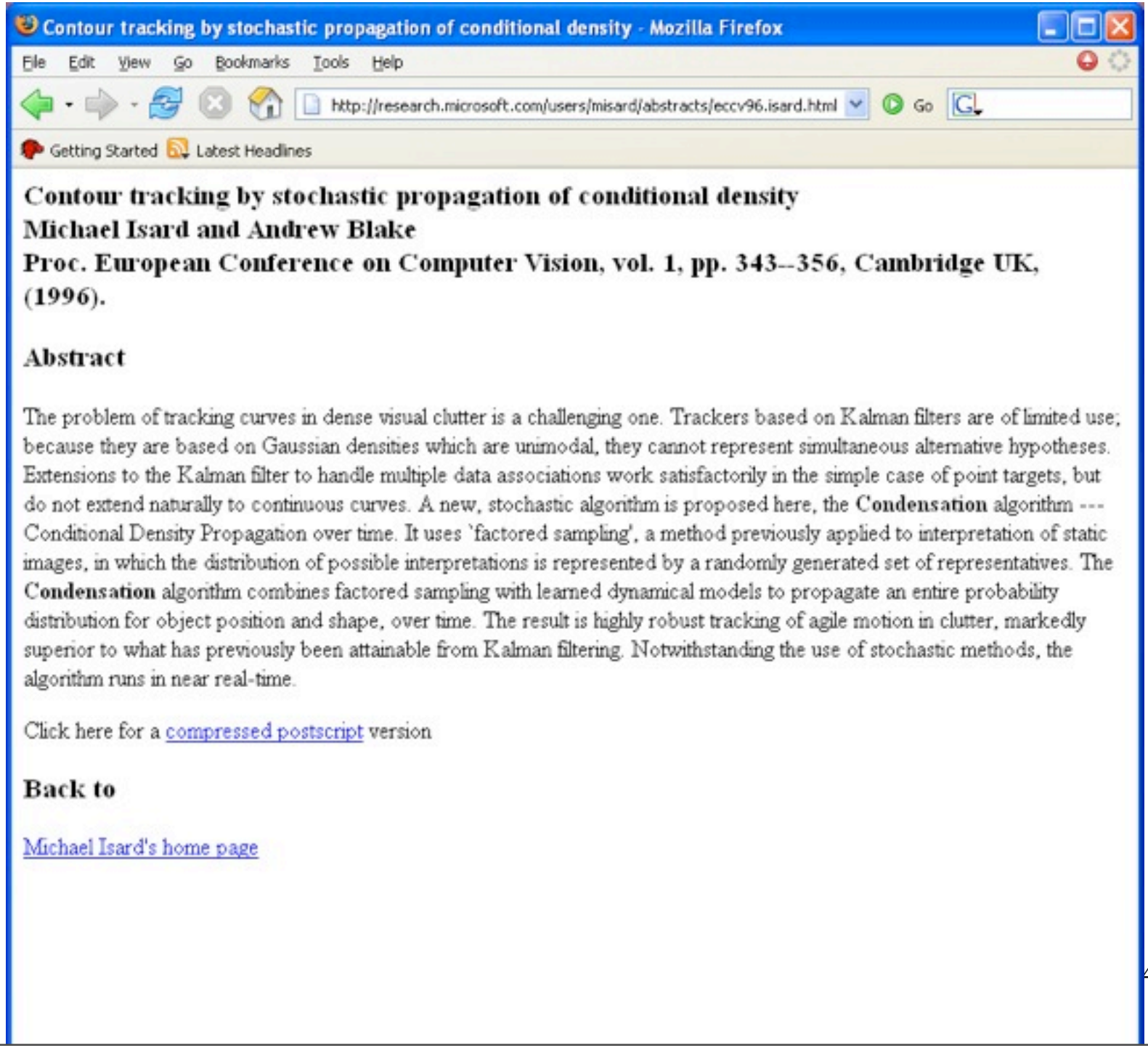
$$s_i^{k,+} = s_i^{k,-} \text{ and } w_i^{k,+} = P(Y_i = y_i|X_i = s_i^{k,-})w_i^{k,-}$$

Resampling: Normalise the weights so that $\sum_i w_i^{k,+} = 1$ and compute the variance of the normalised weights. If this variance exceeds some threshold, then construct a new set of samples by drawing, with replacement, N samples from the old set, using the weights as the probability that a sample will be drawn. The weight of each sample is now $1/N$.

Pictorial view



[Isard 1998]


Contour tracking by stochastic propagation of conditional density - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://research.microsoft.com/users/misard/abstracts/eccv96.isard.html
Go
Getting Started
Latest Headlines
Contour tracking by stochastic propagation of conditional density
Michael Isard and Andrew Blake
Proc. European Conference on Computer Vision, vol. 1, pp. 343--356, Cambridge UK, (1996).
Abstract
The problem of tracking curves in dense visual clutter is a challenging one. Trackers based on Kalman filters are of limited use; because they are based on Gaussian densities which are unimodal, they cannot represent simultaneous alternative hypotheses. Extensions to the Kalman filter to handle multiple data associations work satisfactorily in the simple case of point targets, but do not extend naturally to continuous curves. A new, stochastic algorithm is proposed here, the Condensation algorithm --- Conditional Density Propagation over time. It uses 'factored sampling', a method previously applied to interpretation of static images, in which the distribution of possible interpretations is represented by a randomly generated set of representatives. The Condensation algorithm combines factored sampling with learned dynamical models to propagate an entire probability distribution for object position and shape, over time. The result is highly robust tracking of agile motion in clutter, markedly superior to what has previously been attainable from Kalman filtering. Notwithstanding the use of stochastic methods, the algorithm runs in near real-time.
Click here for a compressed postscript version
Back to
Michael Isard's home page

Edit View History Bookmarks Tools Help

http://www.robots.ox.ac.uk/~misard/condensation.html

Getting Started Latest Headlines

west concord, ma 58&10 Search PageRank Check AutoLink AutoFill Subscribe Options west

Gmail - old conden... | Mozilla Firefox Sta... | west concord, ma ... | Olive Garden Italia... | Approve new CSA... | leafmv.mpg (video... | The Condens...

The Condensation Algorithm

Background

Tracking objects through highly cluttered scenes is difficult. We believe that for tracking to be robust when following agile moving objects, in the presence of dense background clutter, probabilistic algorithms are essential. Previous algorithms, for example the Kalman filter, have been limited in the range of probability distributions they represent. We have developed a new algorithm, the **Condensation** algorithm (**Con**ditional **Dens**ity Propa**gation**) which allows quite general representations of probability. Experimental results show that this increased generality does indeed lead to a marked improvement in tracking performance. In addition to permitting high-quality tracking in clutter, the simplicity of the **Condensation** algorithm also allows the use of non-linear motion models more complex than those commonly used in Kalman filters. We have implemented a mixed discrete/continuous tracker in the **Condensation** framework which switches between multiple continuous Auto-Regressive Process motion models according to a discrete transition matrix. Also, by using the statistical technique of *importance sampling* it is possible to build a **Condensation** tracker which runs in real time, and we have implemented a real-time hand-tracker on a low-end SGI workstation. My D.Phil. thesis gives a thorough description of the algorithm and some applications.

Sample Code

Download source code of a simple implementation of the **Condensation** algorithm.

Results

Here is an MPEG (2.3Mb) showing the **Condensation** algorithm tracking a leaf blowing in the wind, against a background of

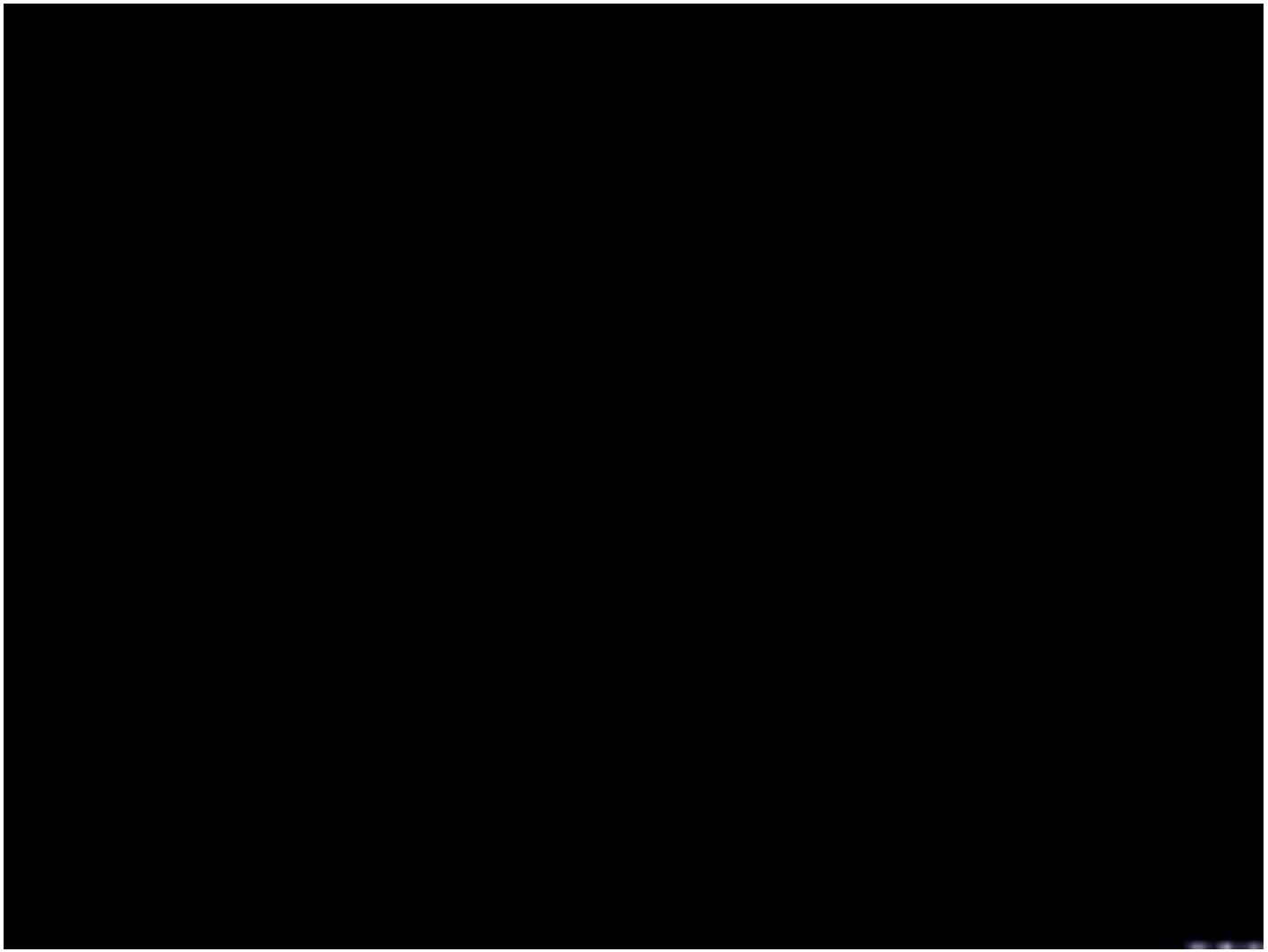
Find: Next Previous Highlight all Match case

Done

Animation of condensation algorithm

[Isard 1998]

Animation of condensation algorithm



[Isard 1998]

Applications

Applications

Tracking

Applications

Tracking

- hands

Applications

Tracking

- hands
- bodies

Applications

Tracking

- hands
- bodies
- Leaves

Applications

Tracking

- hands
- bodies
- Leaves

Applications

Tracking

- hands
- bodies
- Leaves

Applications

Tracking

- hands
- bodies
- Leaves

What might we expect?

Applications

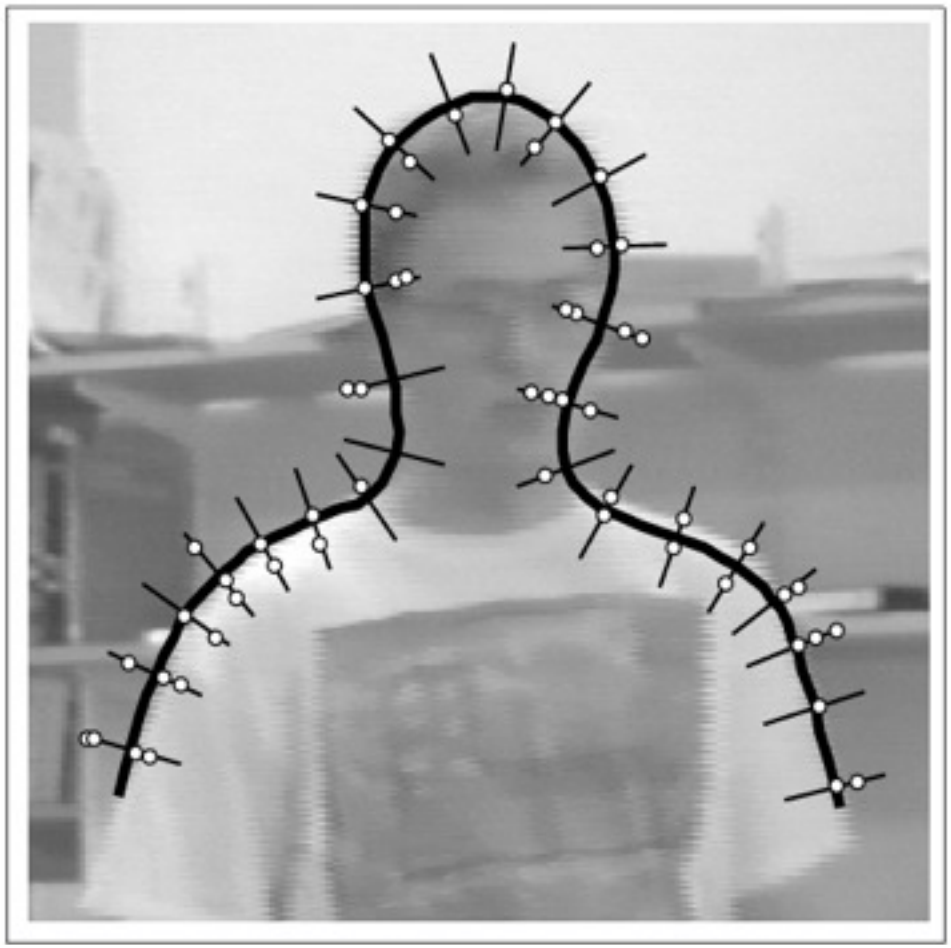
Tracking

- hands
- bodies
- Leaves

What might we expect?

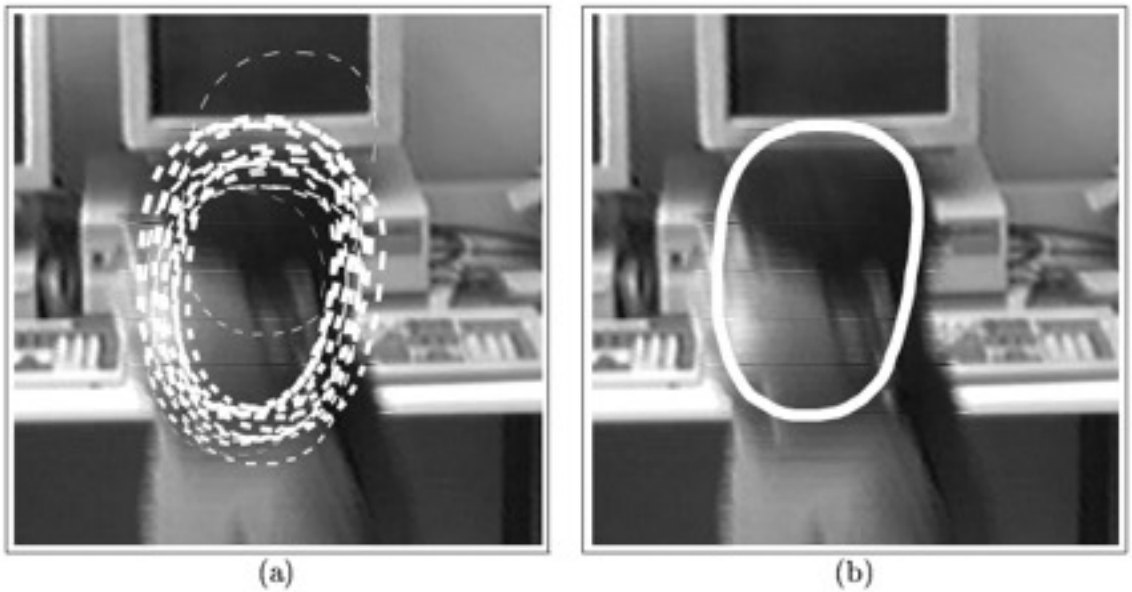
Reliable, robust, slow

Contour tracking



[Isard 1998]

Head tracking

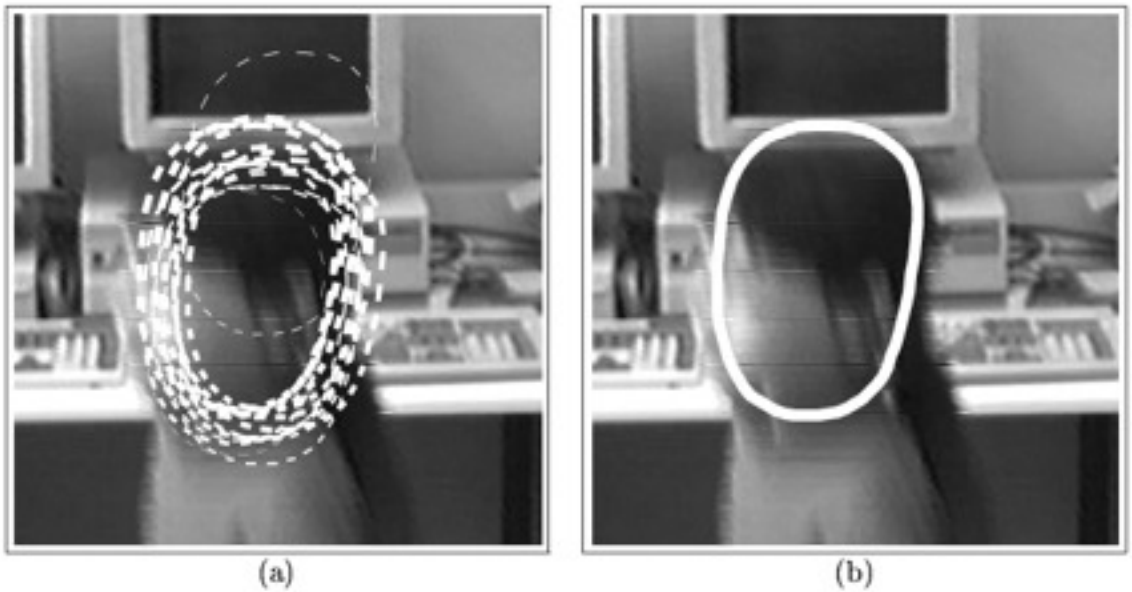


Picture of the states represented by the top weighted particles

The mean state

[Isard 1998]

Head tracking



Picture of the states represented by the top weighted particles

The mean state

[Isard 1998]

Leaf tracking

[Isard 1998]

Leaf tracking

[Isard 1998]

Hand tracking

[Isard 1998]

Hand tracking

[Isard 1998]

Probabilistic Tracking and Reconstruction of 3D Human Motion in Monocular Video Sequences

Presentation of the thesis work of:
Hedvig Sidenbladh, KTH

Thesis opponent: Prof. Bill Freeman, MIT

Thesis supervisors

- Prof. Jan-Olof Eklundh, KTH
- Prof. Michael Black, Brown University

Collaborators

- Dr. David Fleet, Xerox PARC
- Prof. Dirk Ormoneit, Stanford University

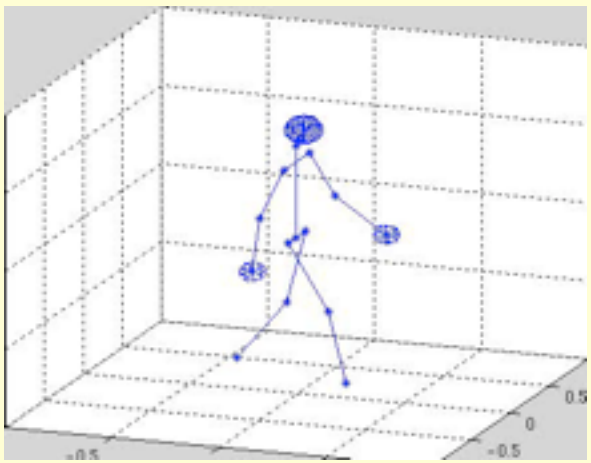
Models of Human Dynamics

- Action-specific model - Walking
  - Training data: 3D motion capture data
  - From training set, learn mean cycle and common modes of deviation (PCA)

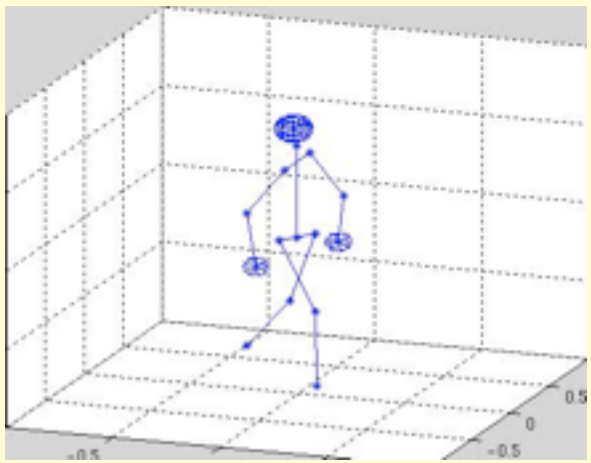
Mean cycle Small noise Large noise

Models of Human Dynamics

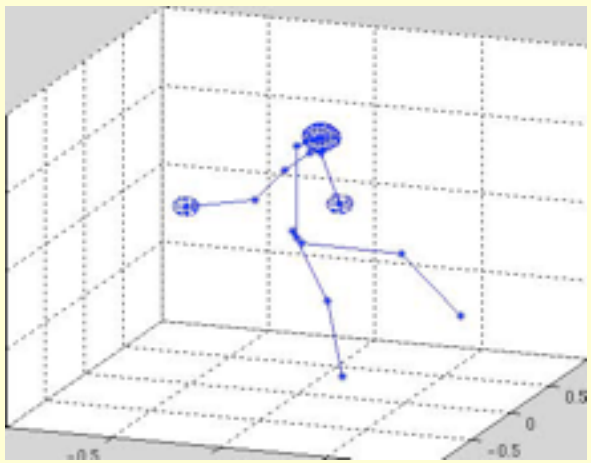
- Action-specific model - Walking
  - Training data: 3D motion capture data
  - From training set, learn mean cycle and common modes of deviation (PCA)



Mean cycle



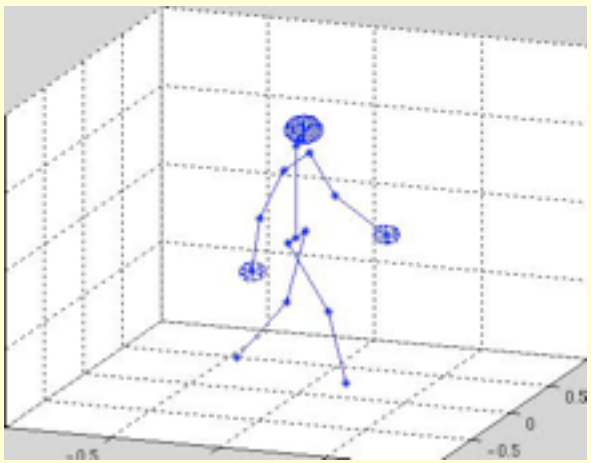
Small noise



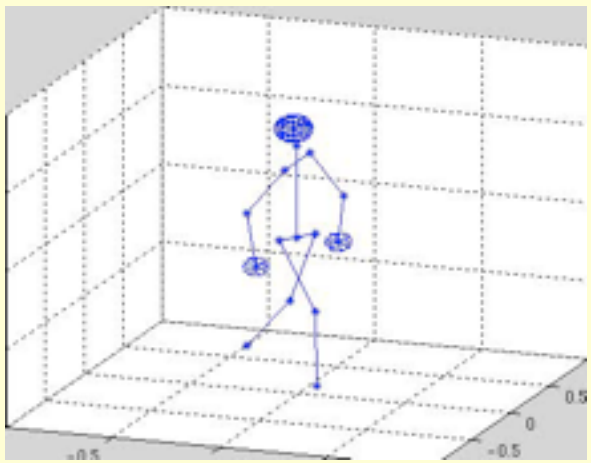
Large noise

Models of Human Dynamics

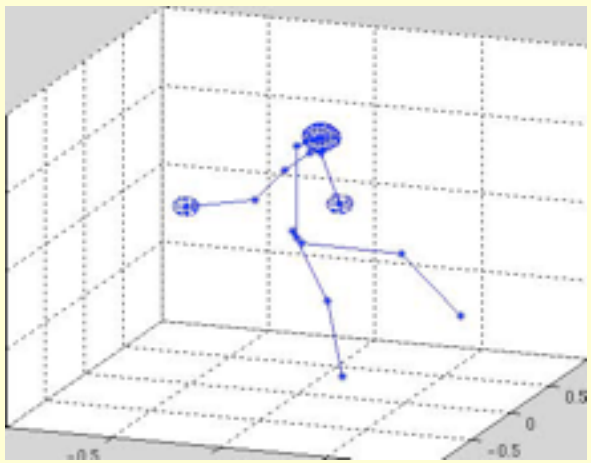
- Action-specific model - Walking
  - Training data: 3D motion capture data
  - From training set, learn mean cycle and common modes of deviation (PCA)



Mean cycle



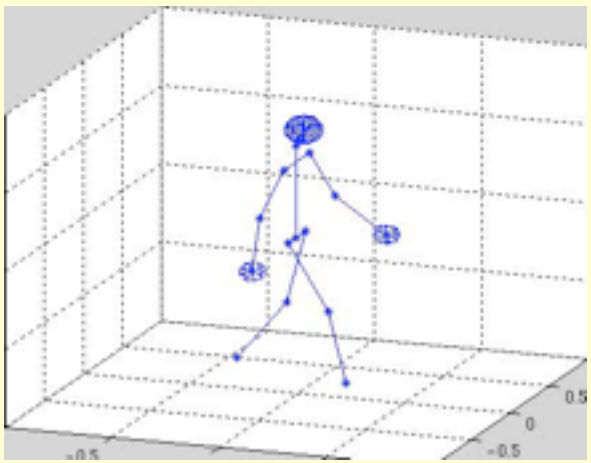
Small noise



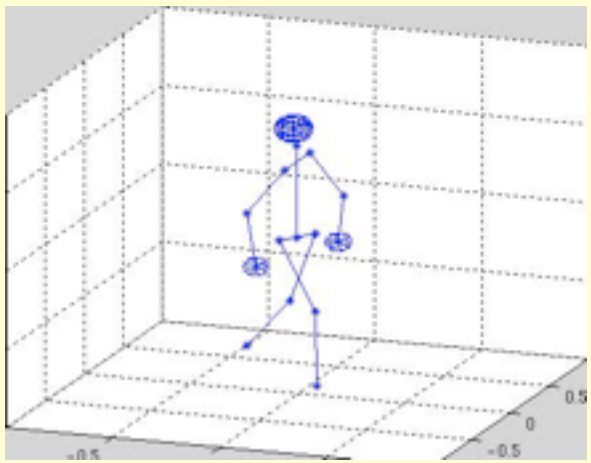
Large noise

Models of Human Dynamics

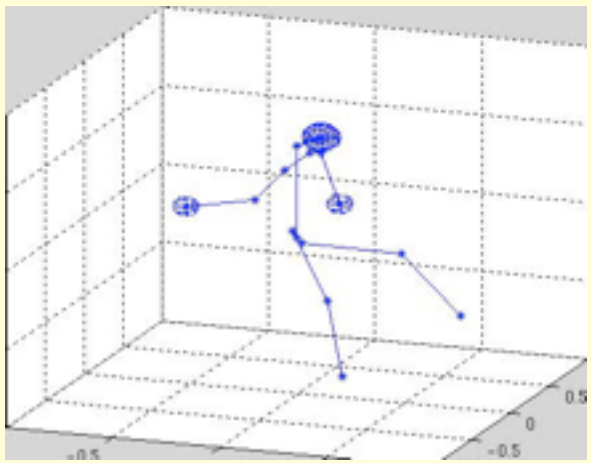
- Action-specific model - Walking
  - Training data: 3D motion capture data
  - From training set, learn mean cycle and common modes of deviation (PCA)



Mean cycle



Small noise



Large noise

Walking Person

#samples
from 15000
to 2500
by using the
learned
likelihood

2500 samples
~10 min/frame

Walking model

Walking Person

#samples
from 15000
to 2500
by using the
learned
likelihood

2500 samples
~10 min/frame

Walking model

No likelihood

* how strong is the walking prior?
  (or is our likelihood doing anything?)

No likelihood

* how strong is the walking prior?
  (or is our likelihood doing anything?)