

Computer Vision Week 11

Tracking

Tracking

- “Generating an inference about the motion of an object given a sequence of images.
- Applications:
 - Motion capture
 - Recognition from motion
 - Surveillance
 - Target tracking

Tracking

- Very general model:
 - We assume there are moving objects, which have an underlying state X
 - There are measurements Y , some of which are functions of this state
 - There is a clock
 - at each tick, the state changes
 - at each tick, we get a new observation

Examples

- object is ball, state is 3D position+velocity, measurements are stereo pairs



- object is person, state is body configuration, measurements are frames, clock is in camera (30 fps)



Problem formulation

- X : state; not directly observable
- Y : output (measurement); we observe this
- $p(X)$ Prior probability of state vector; summarizes prior domain knowledge by independent measurements
- $p(Y)$ Probability of measuring Y ; fixed for any given image
- $p(Y|X)$ Probability of measuring Y given that the state is X ; compares image to expectation based on state
- $p(X|Y)$ Probability of X given that measurement Y has occurred; called state posterior
- X_i, Y_i discrete model; measurements taken at discrete instants of time (clock ticks)

Motion and measurement model “Linear dynamic model”

- $X_{t+1} = A X_t + B W_t$
- $Y_{t+1} = C X_{t+1} + D N_{t+1}$

Three main steps

- **Prediction:** we have seen $\mathbf{y}_0, \dots, \mathbf{y}_{i-1}$ — what state does this set of measurements predict for the i 'th frame? to solve this problem, we need to obtain a representation of $P(\mathbf{X}_i | \mathbf{Y}_0 = \mathbf{y}_0, \dots, \mathbf{Y}_{i-1} = \mathbf{y}_{i-1})$.
- **Data association:** Some of the measurements obtained from the i -th frame may tell us about the object's state. Typically, we use $P(\mathbf{X}_i | \mathbf{Y}_0 = \mathbf{y}_0, \dots, \mathbf{Y}_{i-1} = \mathbf{y}_{i-1})$ to identify these measurements.
- **Correction:** now that we have \mathbf{y}_i — the relevant measurements — we need to compute a representation of $P(\mathbf{X}_i | \mathbf{Y}_0 = \mathbf{y}_0, \dots, \mathbf{Y}_i = \mathbf{y}_i)$.

Simplifying Assumptions

- **Only the immediate past matters:** formally, we require

$$P(\mathbf{X}_i | \mathbf{X}_1, \dots, \mathbf{X}_{i-1}) = P(\mathbf{X}_i | \mathbf{X}_{i-1})$$

This assumption hugely simplifies the design of algorithms, as we shall see; furthermore, it isn't terribly restrictive if we're clever about interpreting \mathbf{X}_i as we shall show in the next section.

- **Measurements depend only on the current state:** we assume that \mathbf{Y}_i is conditionally independent of all other measurements given \mathbf{X}_i . This means that

$$P(\mathbf{Y}_i, \mathbf{Y}_j, \dots, \mathbf{Y}_k | \mathbf{X}_i) = P(\mathbf{Y}_i | \mathbf{X}_i) P(\mathbf{Y}_j, \dots, \mathbf{Y}_k | \mathbf{X}_i)$$

Again, this isn't a particularly restrictive or controversial assumption, but it yields important simplifications.

Tracking as induction

- Assume data association is done
 - Linear dynamic model
- Do correction for the 0'th frame
- Assume we have corrected estimate for i 'th frame
 - show we can do prediction for $i+1$, correction for $i+1$

Base case

Firstly, we assume that we have $P(\mathbf{X}_0)$

$$\begin{aligned} P(\mathbf{X}_0 | \mathbf{Y}_0 = \mathbf{y}_0) &= \frac{P(\mathbf{y}_0 | \mathbf{X}_0) P(\mathbf{X}_0)}{P(\mathbf{y}_0)} \\ &= \frac{P(\mathbf{y}_0 | \mathbf{X}_0) P(\mathbf{X}_0)}{\int P(\mathbf{y}_0 | \mathbf{X}_0) P(\mathbf{X}_0) d\mathbf{X}_0} \\ &\propto P(\mathbf{y}_0 | \mathbf{X}_0) P(\mathbf{X}_0) \end{aligned}$$

Induction step

Given $P(\mathbf{X}_{i-1} | \mathbf{y}_0, \dots, \mathbf{y}_{i-1})$.

Prediction

Prediction involves representing

$$P(\mathbf{X}_i | \mathbf{y}_0, \dots, \mathbf{y}_{i-1})$$

Our independence assumptions make it possible to write

$$\begin{aligned} P(\mathbf{X}_i | \mathbf{y}_0, \dots, \mathbf{y}_{i-1}) &= \int P(\mathbf{X}_i, \mathbf{X}_{i-1} | \mathbf{y}_0, \dots, \mathbf{y}_{i-1}) d\mathbf{X}_{i-1} \\ &= \int P(\mathbf{X}_i | \mathbf{X}_{i-1}, \mathbf{y}_0, \dots, \mathbf{y}_{i-1}) P(\mathbf{X}_{i-1} | \mathbf{y}_0, \dots, \mathbf{y}_{i-1}) d\mathbf{X}_{i-1} \\ &= \int P(\mathbf{X}_i | \mathbf{X}_{i-1}) P(\mathbf{X}_{i-1} | \mathbf{y}_0, \dots, \mathbf{y}_{i-1}) d\mathbf{X}_{i-1} \end{aligned}$$

Induction step

Correction

Correction involves obtaining a representation of

$$P(\mathbf{X}_i | \mathbf{y}_0, \dots, \mathbf{y}_i)$$

Our independence assumptions make it possible to write

$$\begin{aligned} P(\mathbf{X}_i | \mathbf{y}_0, \dots, \mathbf{y}_i) &= \frac{P(\mathbf{X}_i, \mathbf{y}_0, \dots, \mathbf{y}_i)}{P(\mathbf{y}_0, \dots, \mathbf{y}_i)} \\ &= \frac{P(\mathbf{y}_i | \mathbf{X}_i, \mathbf{y}_0, \dots, \mathbf{y}_{i-1}) P(\mathbf{X}_i | \mathbf{y}_0, \dots, \mathbf{y}_{i-1}) P(\mathbf{y}_0, \dots, \mathbf{y}_{i-1})}{P(\mathbf{y}_0, \dots, \mathbf{y}_i)} \\ &= P(\mathbf{y}_i | \mathbf{X}_i) P(\mathbf{X}_i | \mathbf{y}_0, \dots, \mathbf{y}_{i-1}) \frac{P(\mathbf{y}_0, \dots, \mathbf{y}_{i-1})}{P(\mathbf{y}_0, \dots, \mathbf{y}_i)} \\ &= \frac{P(\mathbf{y}_i | \mathbf{X}_i) P(\mathbf{X}_i | \mathbf{y}_0, \dots, \mathbf{y}_{i-1})}{\int P(\mathbf{y}_i | \mathbf{X}_i) P(\mathbf{X}_i | \mathbf{y}_0, \dots, \mathbf{y}_{i-1}) d\mathbf{X}_i} \end{aligned}$$

Linear dynamic models

- Use notation \sim to mean "has the pdf of", $N(a, b)$ is a normal distribution with mean a and covariance b .
- Then a linear dynamic model has the form

$$\mathbf{x}_i = N(\mathbf{D}_{i-1}\mathbf{x}_{i-1}; \Sigma_{d_i})$$

$$\mathbf{y}_i = N(\mathbf{M}_i\mathbf{x}_i; \Sigma_{m_i})$$

- This is much, much more general than it looks, and extremely powerful

Examples

- Drifting points
 - we assume that the new position of the point is the old one, plus noise.
 - For the measurement model, we may not need to observe the whole state of the object
 - e.g. a point moving in 3D, at the 3k'th tick we see x , 3k+1'th tick we see y , 3k+2'th tick we see z
 - in this case, we can still make decent estimates of **all three** coordinates at each tick.
 - This property, which does not apply to every model, is called **Observability**

Examples

- Points moving with constant velocity
- Periodic motion
- Points moving with constant acceleration

Points moving with constant velocity

- We have

$$u_i = u_{i-1} + \Delta t v_{i-1} + \epsilon_i$$

$$v_i = v_{i-1} + \zeta_i$$

- (the Greek letters denote noise terms)
- Stack (u, v) into a single state vector

$$\begin{pmatrix} u \\ v \end{pmatrix}_i = \begin{pmatrix} 1 & \Delta t \\ 0 & 1 \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix}_{i-1} + \text{noise}$$

- which is the form we had above

Points moving with constant acceleration

- We have

$$u_i = u_{i-1} + \Delta t v_{i-1} + \epsilon_i$$

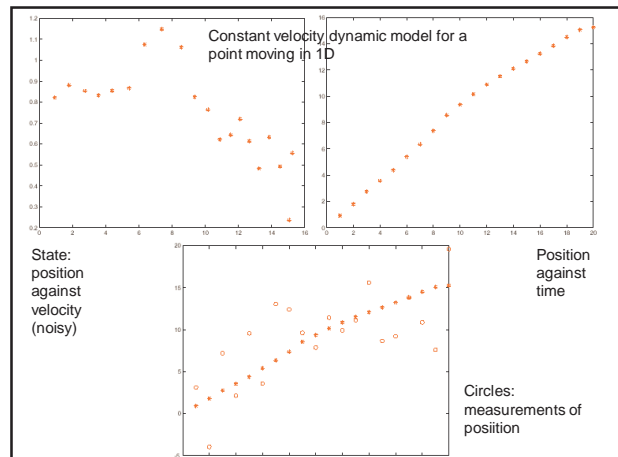
$$v_i = v_{i-1} + \Delta t a_{i-1} + \zeta_i$$

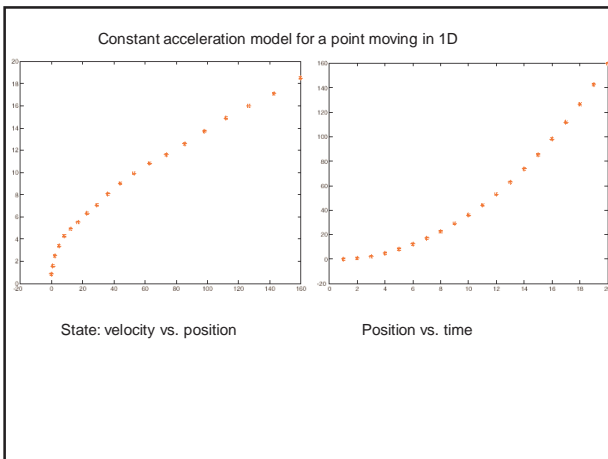
$$a_i = a_{i-1} + \xi_i$$

- (the Greek letters denote noise terms)
- Stack (u, v) into a single state vector

$$\begin{pmatrix} u \\ v \\ a \end{pmatrix}_i = \begin{pmatrix} 1 & \Delta t & 0 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} u \\ v \\ a \end{pmatrix}_{i-1} + \text{noise}$$

- which is the form we had above





The Kalman Filter

- Key ideas:
 - Linear models interact uniquely well with Gaussian noise - make the prior Gaussian, everything else Gaussian and the calculations are easy
 - Gaussians are really easy to represent --- once you know the mean and covariance, you're done

The Kalman Filter in 1D

- Dynamic Model

$$x_i \sim N(d_i x_{i-1}, \sigma_{d_i}^2)$$

$$y_i \sim N(m_i x_i, \sigma_{m_i}^2)$$
- Notation
 - mean of $P(X_i|y_0, \dots, y_{i-1})$ as \bar{X}_i^- Predicted mean
 - Corrected mean — mean of $P(X_i|y_0, \dots, y_i)$ as \bar{X}_i^+
 - the standard deviation of $P(X_i|y_0, \dots, y_{i-1})$ as σ_i^-
 - of $P(X_i|y_0, \dots, y_i)$ as σ_i^+

Prediction for 1D Kalman filter

- The new state is obtained by
 - multiplying old state by known constant
 - adding zero-mean noise
- Therefore, predicted mean for new state is
 - constant times mean for old state
- Predicted variance is
 - sum of constant² times old state variance and noise variance

Because:
old state is normal random variable, multiplying normal rv by constant implies mean is multiplied by a constant variance by square of constant, adding zero mean noise adds zero to the mean, adding rv's adds variance

Dynamic Model:

$$x_i \sim N(d_i x_{i-1}, \sigma_{d_i}^2)$$

$$y_i \sim N(m_i x_i, \sigma_{m_i}^2)$$

Start Assumptions: \bar{x}_0^- and σ_0^- are known
Update Equations: Prediction

$$\bar{x}_i^- = d_i \bar{x}_{i-1}^-$$

$$\sigma_i^- = \sqrt{\sigma_{d_i}^2 + (d_i \sigma_{i-1}^-)^2}$$

Update Equations: Correction

$$\bar{x}_i^+ = \left(\frac{\bar{x}_i^- \sigma_{m_i}^2 + m_i y_i (\sigma_i^-)^2}{\sigma_{m_i}^2 + m_i^2 (\sigma_i^-)^2} \right)$$

$$\sigma_i^+ = \sqrt{\left(\frac{\sigma_{m_i}^2 (\sigma_i^-)^2}{(\sigma_{m_i}^2 + m_i^2 (\sigma_i^-)^2)} \right)}$$

Correction for 1D Kalman filter

- Pattern match to identities given in book
 - basically, guess the integrals, get:

$$\bar{x}_i^+ = \left(\frac{\bar{x}_i^- \sigma_{m_i}^2 + m_i y_i (\sigma_i^-)^2}{\sigma_{m_i}^2 + m_i^2 (\sigma_i^-)^2} \right)$$

$$\sigma_i^+ = \sqrt{\left(\frac{\sigma_{m_i}^2 (\sigma_i^-)^2}{(\sigma_{m_i}^2 + m_i^2 (\sigma_i^-)^2)} \right)}$$
- Notice:
 - if measurement noise is small, we rely mainly on the measurement, if it's large, mainly on the prediction

In higher dimensions, derivation follows the same lines, but isn't as easy.

Dynamic Model:

$$\mathbf{x}_i \sim N(\mathcal{D}_i \mathbf{x}_{i-1}, \Sigma_{d_i})$$

$$\mathbf{y}_i \sim N(\mathcal{M}_i \mathbf{x}_i, \Sigma_{m_i})$$

Start Assumptions: $\bar{\mathbf{x}}_0$ and Σ_0 are known
 Update Equations: Prediction

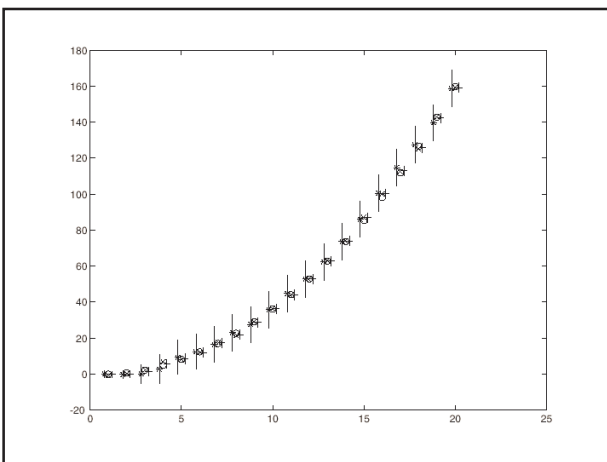
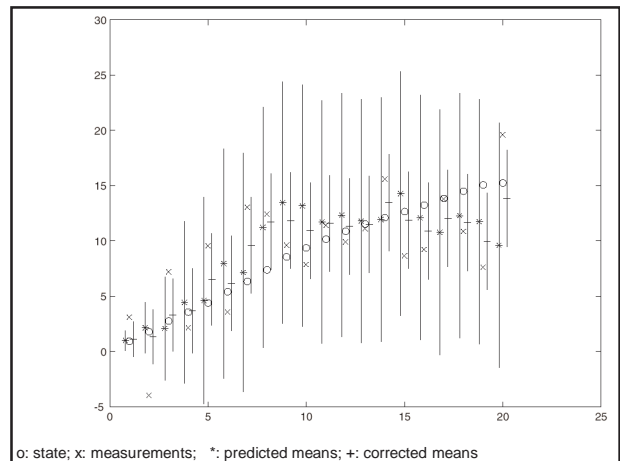
$$\bar{\mathbf{x}}_i^- = \mathcal{D}_i \bar{\mathbf{x}}_{i-1}^+$$

$$\Sigma_i^- = \Sigma_{d_i} + \mathcal{D}_i \sigma_{i-1}^+ \mathcal{D}_i$$

Update Equations: Correction

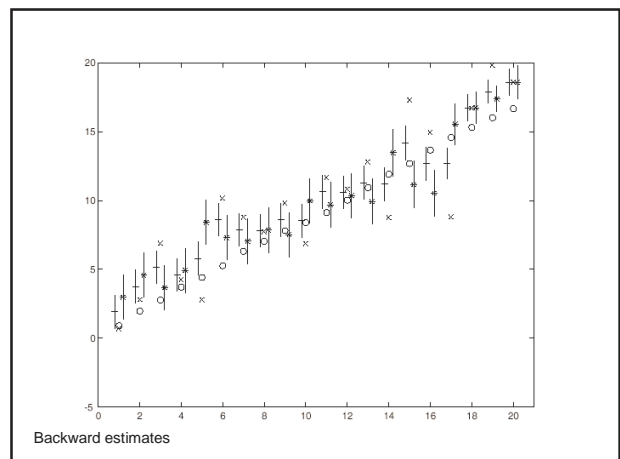
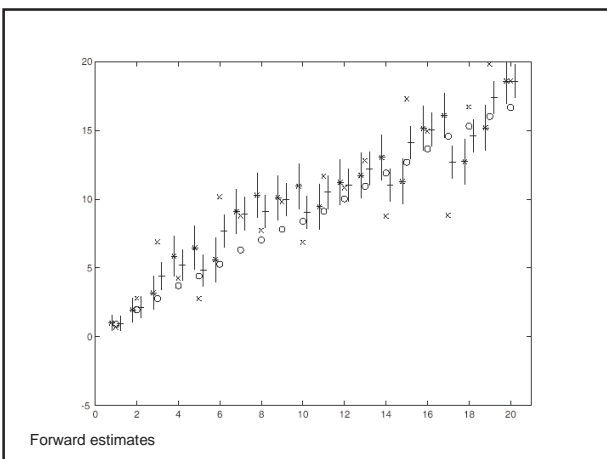
$$\mathcal{K}_i = \Sigma_i^- \mathcal{M}_i^T [\mathcal{M}_i \Sigma_i^- \mathcal{M}_i^T + \Sigma_{m_i}]^{-1}$$

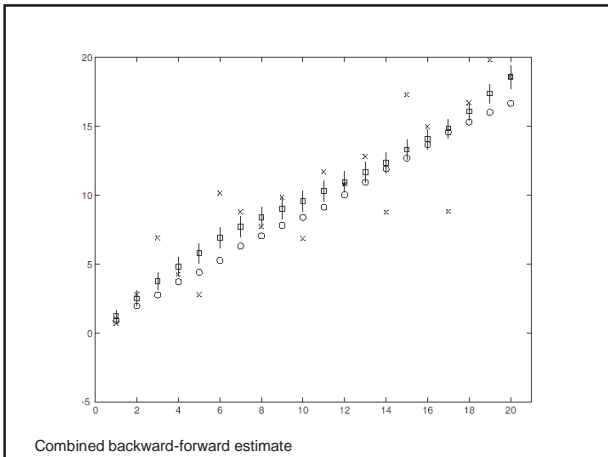
$$\bar{\mathbf{x}}_i^+ = \bar{\mathbf{x}}_i^- + \mathcal{K}_i [\mathbf{y}_i - \mathcal{M}_i \bar{\mathbf{x}}_i^-]$$

$$\Sigma_i^+ = [I_d - \mathcal{K}_i \mathcal{M}_i] \Sigma_i^-$$


Smoothing

- Idea
 - We don't have the best estimate of state - what about the future?
 - Run two filters, one moving forward, the other backward in time.
 - Now combine state estimates
 - The crucial point here is that we can obtain a smoothed estimate by viewing the backward filter's prediction as yet another measurement for the forward filter
 - so we've already done the equations





Object State

- affine transformation parameters of a patch w.r.t. the first image [Shi 1994]
- parameters of a shape [Comaniciu 2003][Perez 2002]
- parameters of the contour [Isard 1998][Wu 2003]
- speed and acceleration [Vo 2003]
- parameters describing target appearance [Cootes 1998]

Measurement

- Template
- Single histogram
- Multi-part histograms
- Multiple features

Solving the Bayes equations

Gaussian and linear:

- Kalman filter [Kalman 1960]

Gaussian non-linear:

- Extended Kalman filter
 - first order Taylor expansion
- Unscented Kalman filter [Julier 1997]
 - non-linear propagation of the first two order moments

Problem: Gaussian assumption does not hold in real-world image sequences; posterior multimodality, non-Gaussian noise

Non-Gaussian non-linear

- Monte Carlo methods

Sequential Monte Carlo (Particle Filter)

- Monte Carlo approximation of the Bayes recursion
 - Approximate the posterior pdf

$$p(x_k | z_{1:k}) \approx \sum_{i=1}^L \omega_k^{(i)} \delta(x_k - x_k^{(i)})$$

- Samples propagated using state and observation equations

$$\omega_k^{(i)} \propto \omega_{k-1}^{(i)} \frac{g(z_k | x_k^{(i)}) f(x_k^{(i)} | x_{k-1}^{(i)})}{q(x_k^{(i)} | x_{k-1}^{(i)}, z_k)} \quad E[x_k | p] = \sum_{i=1}^L \omega_k^{(i)} x_k^{(i)}$$

CONDENSATION [Isard 1996]

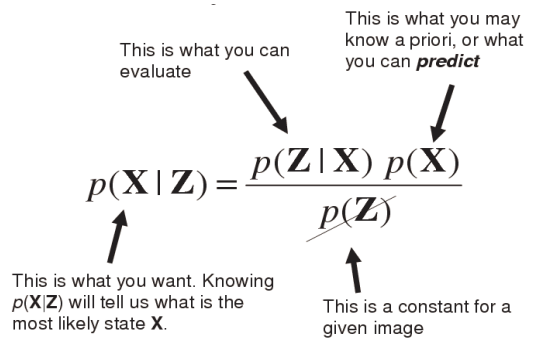
Notation

- X State vector (curve's position and orientation)
- Z Measurement Vector (image edge locations)
- $p(X)$ Prior probability of state vector; summarizes prior domain knowledge by independent measurements
- $p(Z)$ Possibility of measuring Z; fixed for any given image
- $p(Z|X)$ Possibility of measuring Z given that the state is X; compares image to expectation based on state
- $p(X|Z)$ Possibility of X given that measurement Z has occurred; called state posterior

Tracking as Estimation

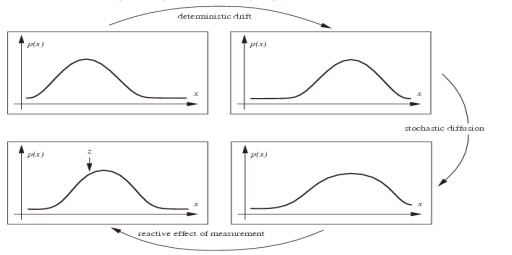
- Compute state posterior, $p(\mathbf{X}|\mathbf{Z})$, and select next state to be the one that maximizes this (Maximum a Posteriori estimate)
- Measurements are complex and noisy, so posterior cannot be evaluated in closed form
- Particle filter (iterative sampling) idea: Stochastically approximate the state posterior with a set of N weighted particles
- Use Bayes' rule to compute $p(\mathbf{X}|\mathbf{Z})$

Bayes' Rule



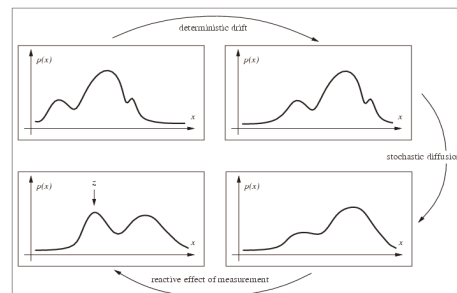
Temporal propagation of conditional densities

- The Kalman filter as a recursive linear estimator is a special case, applying only to Gaussian densities, of a more general probability density propagation process.
- In the simple Gaussian case, the diffusion is purely linear and the density function evolves as a Gaussian pulse that translates, spreads and is reinforced, remaining Gaussian throughout, as in figure, a process that is described analytically and exactly by the Kalman filter.



Temporal propagation of conditional densities

- The random component of the dynamical model leads to spreading - increasing uncertainty - while the deterministic component causes density function to drift.
- The effect of an external observation z_i is to superimpose a reactive effect on the diffusion in which the density tends to peak in the vicinity of observations.
- In clutter, there are typically several competing observations and these tend to encourage a non-Gaussian state-density.



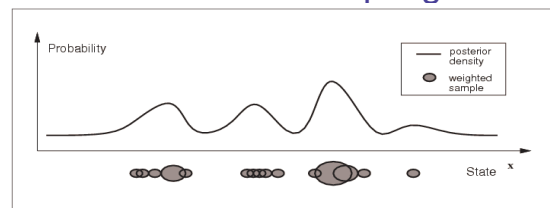
Factored Sampling

- Generate a set of samples that approximates the posterior $p(\mathbf{X}|\mathbf{Z})$
- Sample set $\mathbf{s}=\{\mathbf{s}^{(0)}, \dots, \mathbf{s}^{(N)}\}$ generated from $p(\mathbf{X})$; each sample has a weight

$$\pi_i = \frac{p_z(\mathbf{s}^{(i)})}{\sum_{j=1}^N p_z(\mathbf{s}^{(j)})}$$

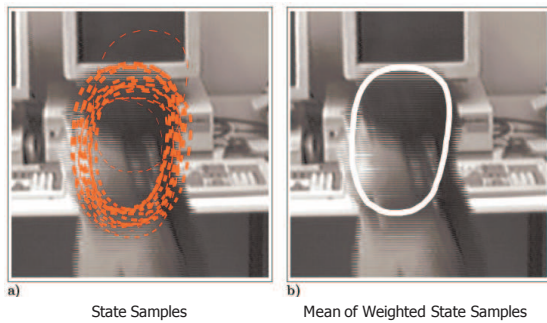
$$p_z(\mathbf{x}) = p(\mathbf{z}|\mathbf{x}),$$

Factored Sampling



A set of points, the centers of the blobs in the figure, is sampled randomly from a prior density $p(\mathbf{x})$. Each sample is assigned a weight (depicted by blob area) in proportion to the value of the observation density. The weighted point set then serves as a representation of the posterior density, suitable for sampling.

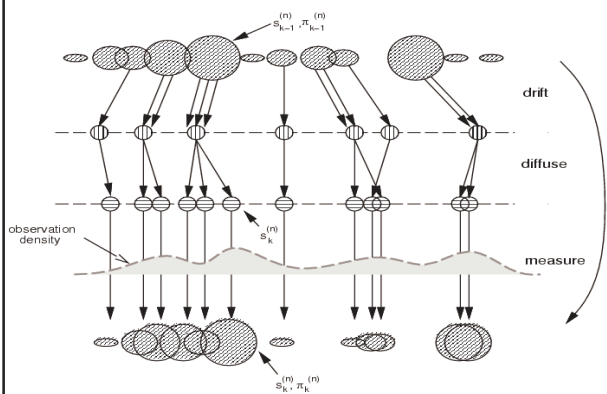
Estimating Target States



CONDENSATION Algorithm

- 1. **Select:** Randomly select N particles from $\{S_{t-1}^{(n)}\}$ based on weights $\pi_{t-1}^{(n)}$; same particle may be picked multiple times (factored sampling)
- 2. **Predict:** Move particles according to deterministic dynamics (drift), then perturb individually (diffuse)
- 3. **Measure:** Get a likelihood for each new sample by comparing it with the image's local appearance, i.e., based on $p(z_t|x_t)$; then update weight accordingly to obtain $\{(S_t^{(n)}, \pi_t^{(n)})\}$

One time step in Condensation



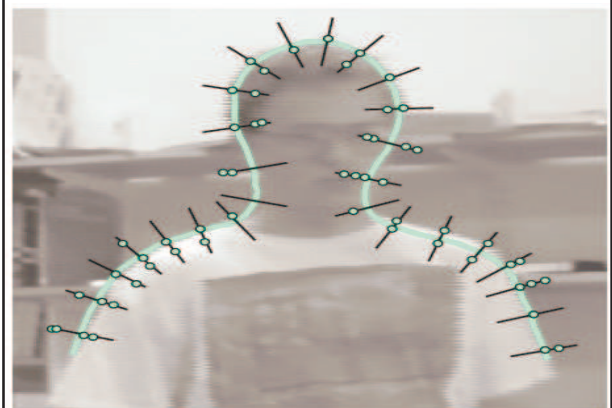
Notes on Updating

- Enforcing plausibility: Particles that represent impossible configurations are discarded.
- Diffusion is modeled with a Gaussian
- Likelihood function: Convert "goodness of prediction" score to pseudo-probability
 - More markings closer to predicted markings gives higher likelihood

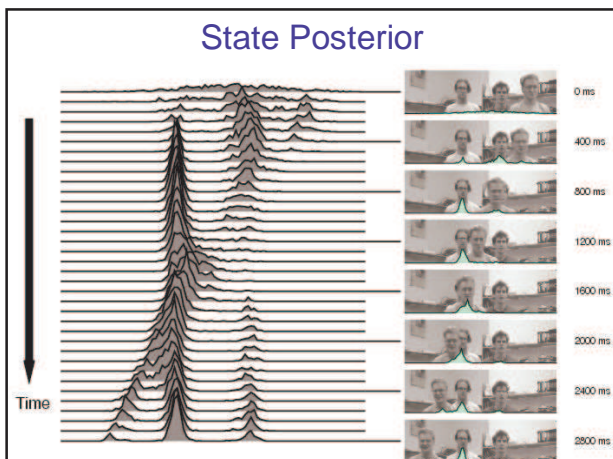
Object Motion Model

- For video tracking we need a way to propagate probability densities, so we need a "motion model" such as
 - $X_{t+1} = A X_t + B W_t$ where W is a noise term and A and B are state transition matrices that can be learned from training sequences
- The state, X , of an object, e.g., a B-spline curve, can be represented as a point in 6D state space of possible 2D affine transformations of the object

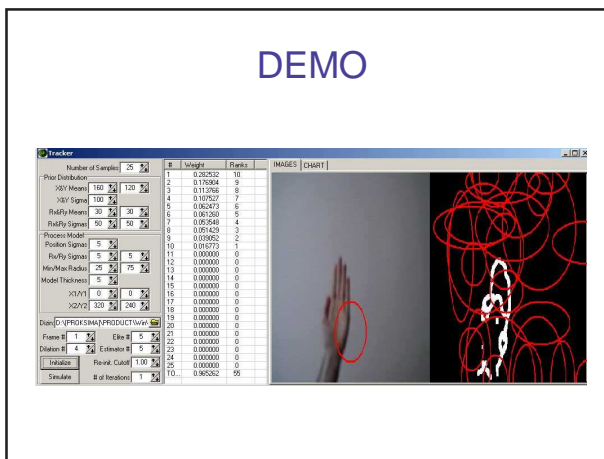
Observation Process



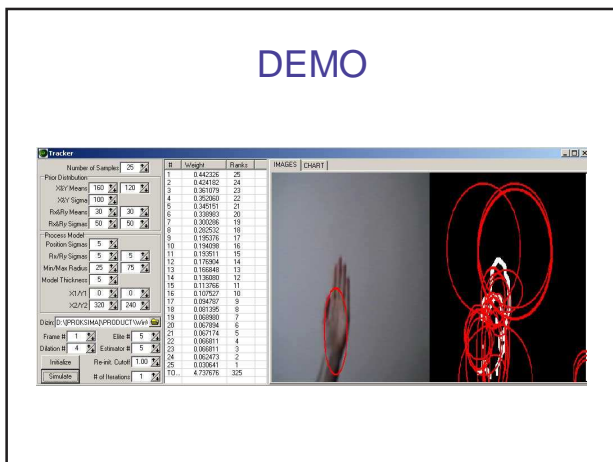
State Posterior



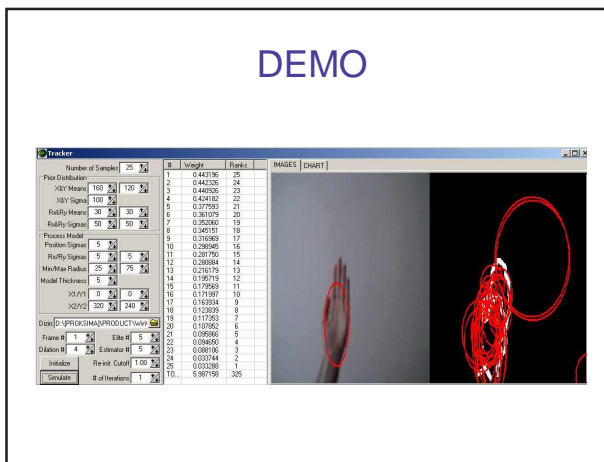
DEMO



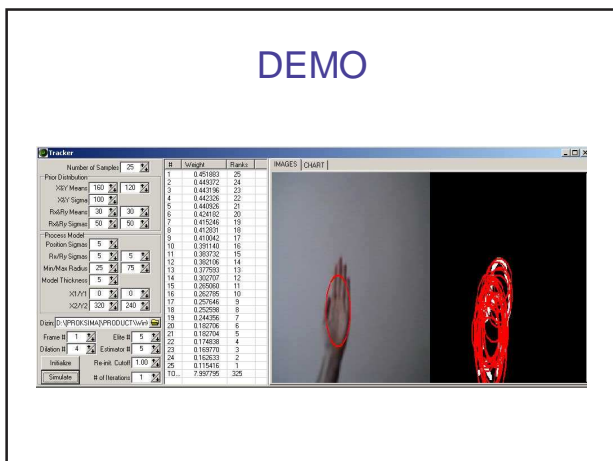
DEMO



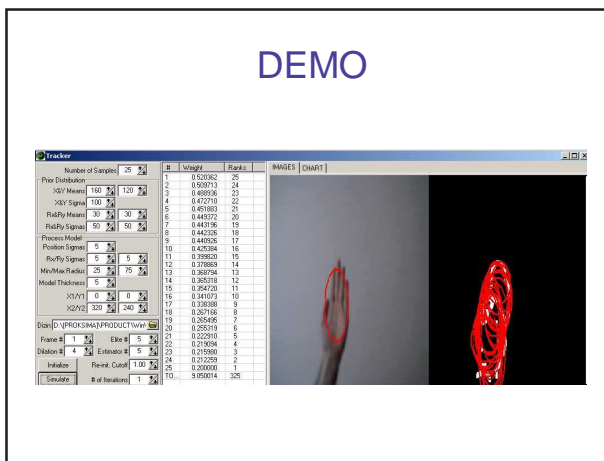
DEMO



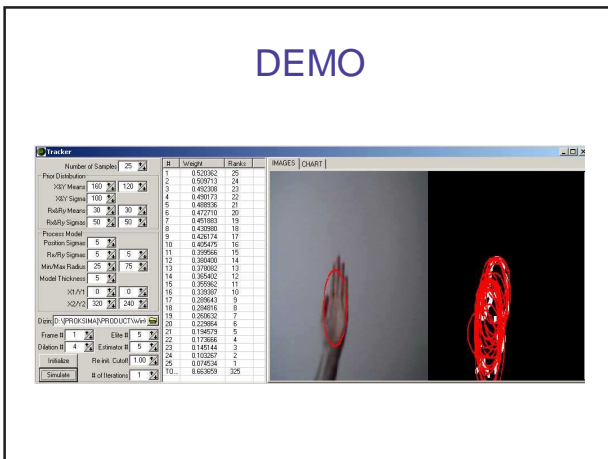
DEMO



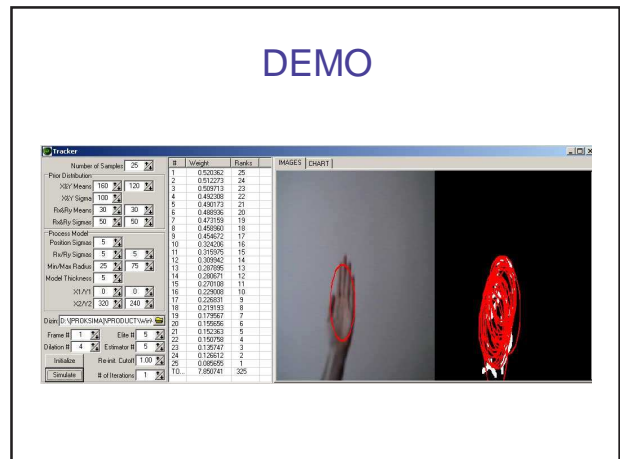
DEMO



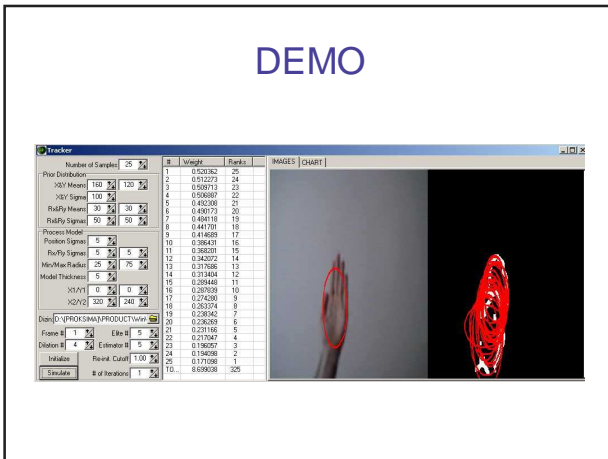
DEMO



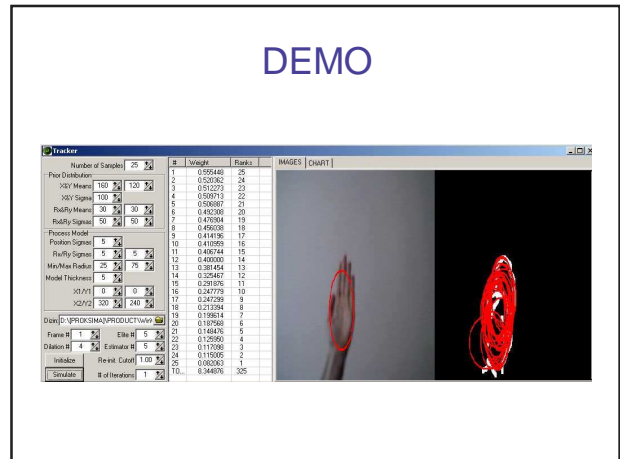
DEMO



DEMO



DEMO



DEMO

