Computer Vision 2
WS 2018/19

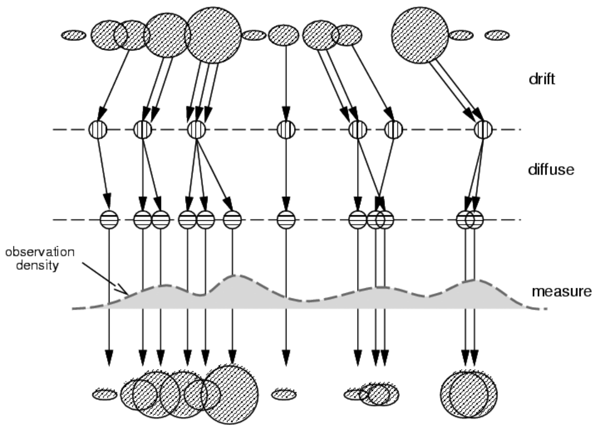
Part 8 – Beyond Kalman Filters
13.11.2018

Prof. Dr. Bastian Leibe

RWTH Aachen University, Computer Vision Group
http://www.vision.rwth-aachen.de

Course Outline

- Single-Object Tracking
- Bayesian Filtering
  - Kalman Filters, EKF
  - Particle Filters
- Multi-Object Tracking
- Visual Odometry
- Visual SLAM & 3D Reconstruction
- Deep Learning for Video Analysis



Today: Beyond Gaussian Error Models

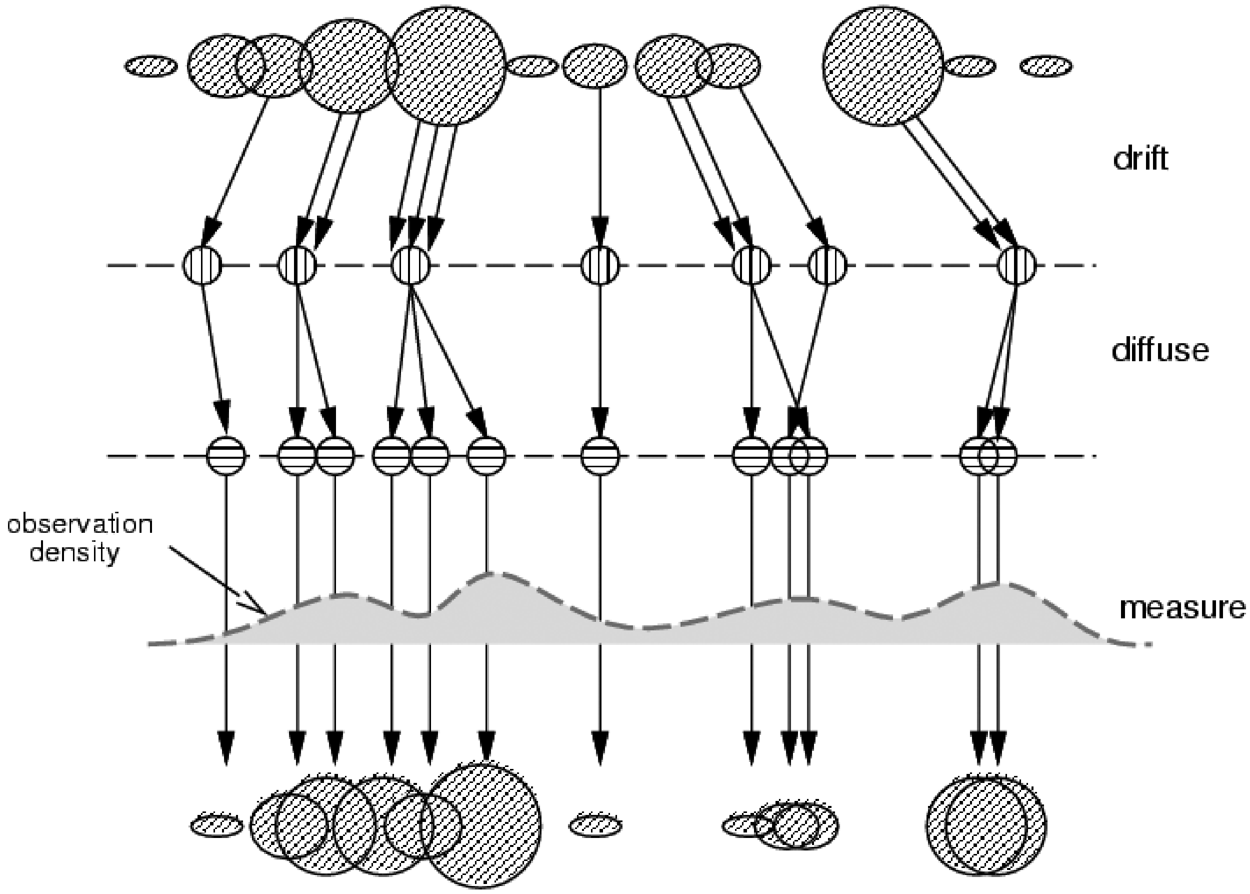


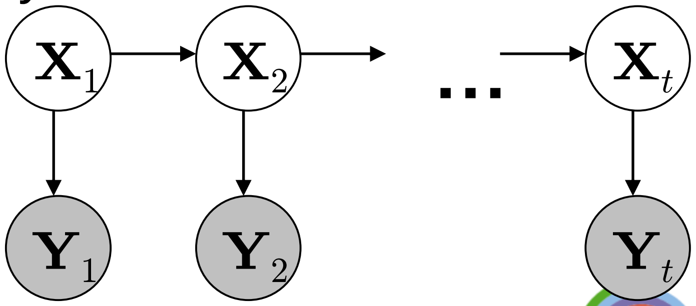
Figure from Isard & Blake

Topics of This Lecture

- Recap: Kalman Filter
  - Basic ideas
  - Kalman filter for 1D state
  - General Kalman filter
  - Limitations
  - Extensions
- Particle Filters
  - Basic ideas
  - Propagation of general densities
  - Factored sampling

Recap: Tracking as Inference

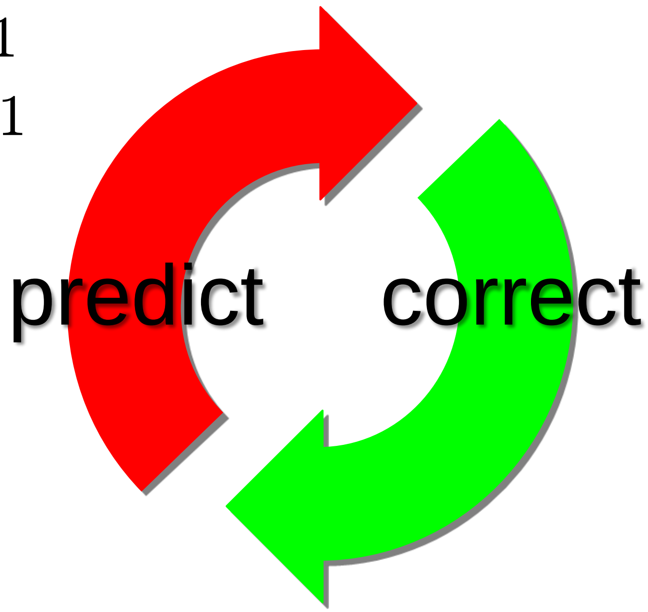
- Inference problem
  - The hidden state consists of the true parameters we care about, denoted $\mathbf{X}$.
  - The measurement is our noisy observation that results from the underlying state, denoted $\mathbf{Y}$.
  - At each time step, state changes (from $\mathbf{X}_{t\text{-}1}$ to $\mathbf{X}_t$) and we get a new observation $\mathbf{Y}_t$.
- Our goal: recover most likely state $\mathbf{X}_t$ given
  - All observations seen so far.
  - Knowledge about dynamics of state transitions.



Slide credit: Kristen Grauman

Recap: Tracking as Induction

- Base case:
  - Assume we have initial prior that predicts state in absence of any evidence: $P(\mathbf{X}_0)$
  - At the first frame, *correct* this given the value of $\mathbf{Y}_0 = \mathbf{y}_0$
- Given corrected estimate for frame t:
  - Predict for frame $t+1$
  - Correct for frame $t+1$



Slide credit: Svetlana Lazebnik

Recap: Prediction and Correction

- Prediction:

$$P(X_t \mid y_0, \ldots, y_{t-1}) = \int \underbrace{P(X_t \mid X_{t-1})}_{\text{Dynamics model}} \underbrace{P(X_{t-1} \mid y_0, \ldots, y_{t-1})}_{\text{Corrected estimate from previous step}} dX_{t-1}$$

- Correction:

$$P(X_t \mid y_0, \ldots, y_t) = \frac{\overbrace{P(y_t \mid X_t)}^{\text{Observation model}} \overbrace{P(X_t \mid y_0, \ldots, y_{t-1})}^{\text{Predicted estimate}}}{\int P(y_t \mid X_t) P(X_t \mid y_0, \ldots, y_{t-1}) dX_t}$$

Slide credit: Svetlana Lazebnik

Recap: Linear Dynamic Models

- Dynamics model
  - State undergoes linear transformation D_t plus Gaussian noise

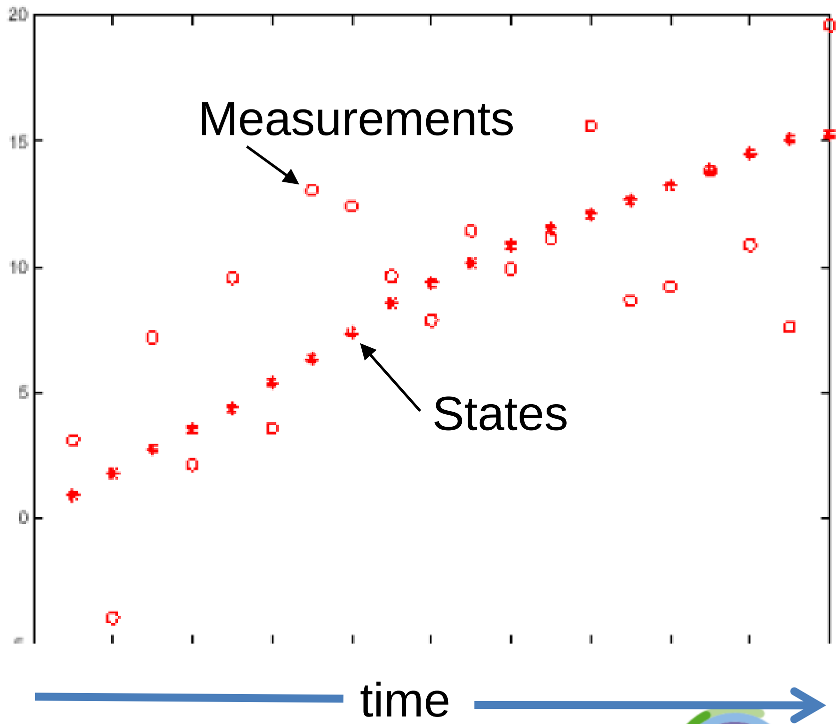
$$\boldsymbol{x}_t \sim N\left(\boldsymbol{D}_t \boldsymbol{x}_{t-1}, \Sigma_{d_t}\right)$$

- Observation model
  - Measurement is linearly transformed state plus Gaussian noise

$$\boldsymbol{y}_t \sim N\left(\boldsymbol{M}_t \boldsymbol{x}_t, \Sigma_{m_t}\right)$$

Slide credit: Svetlana Lazebnik, Kristen Grauman

Example: Constant Velocity (1D Points)



Slide credit: Kristen Grauman

Figure from Forsyth & Ponce

Example: Constant Velocity (1D Points)

- State vector: position p and velocity v

$$x_t = \begin{bmatrix} p_t \\ v_t \end{bmatrix} \qquad \begin{aligned} p_t &= \\ v_t &= \end{aligned}$$

(greek letters denote noise terms)

$$x_t = D_t x_{t-1} + noise =$$

- Measurement is position only

$$y_t = M x_t + noise =$$

Slide credit: Svetlana Lazebnik, Kristen Grauman

Example: Constant Velocity (1D Points)

- State vector: position p and velocity v

$$x_t = \begin{bmatrix} p_t \\ v_t \end{bmatrix} \qquad \begin{aligned} p_t &= p_{t-1} + (\Delta t) v_{t-1} + \varepsilon \\ v_t &= v_{t-1} + \xi \end{aligned}$$

(greek letters denote noise terms)

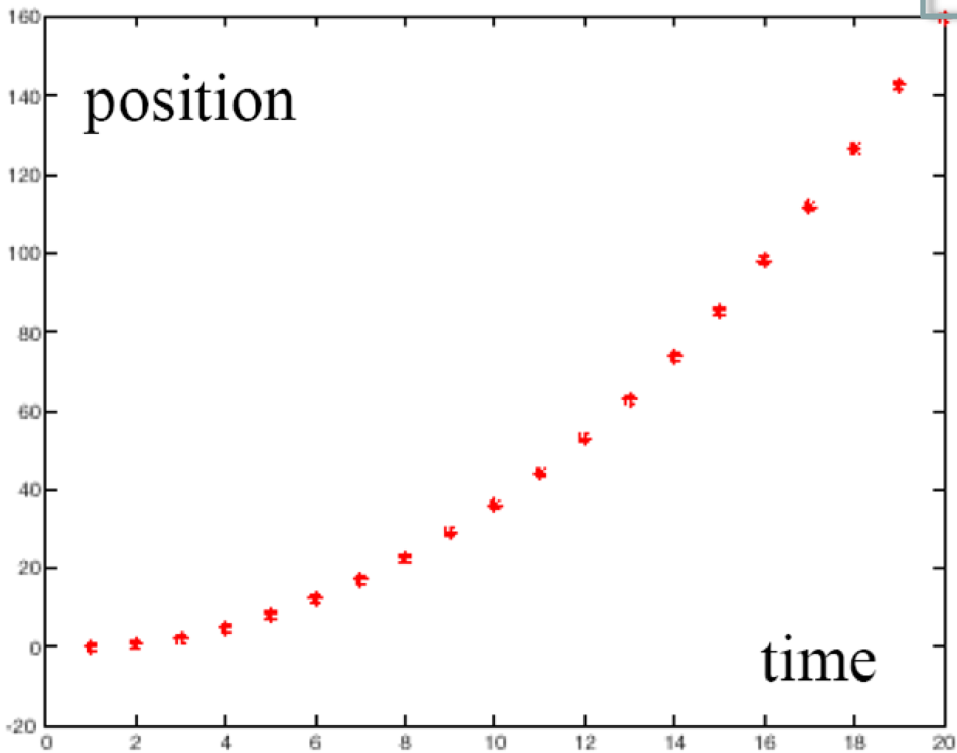
$$x_t = D_t x_{t-1} + noise = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p_{t-1} \\ v_{t-1} \end{bmatrix} + noise$$

- Measurement is position only

$$y_t = M x_t + noise = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} p_t \\ v_t \end{bmatrix} + noise$$

Slide credit: Svetlana Lazebnik, Kristen Grauman

Example: Constant Acceleration (1D Points)



Slide credit: Kristen Grauman

Figure from Forsyth & Ponce

Example: Constant Acceleration (1D Points)

- State vector: position p, velocity v, and acceleration a.

$$x_t = \begin{bmatrix} p_t \\ v_t \\ a_t \end{bmatrix} \qquad \begin{aligned} p_t &= p_{t-1} + (\Delta t) v_{t-1} + \tfrac{1}{2}(\Delta t)^2 a_{t-1} + \varepsilon \\ v_t &= v_{t-1} + (\Delta t) a_{t-1} + \xi \\ a_t &= a_{t-1} + \zeta \end{aligned}$$

(greek letters denote noise terms)

$$x_t = D_t x_{t-1} + noise = \begin{bmatrix} 1 & \Delta t & \frac{1}{2}(\Delta t)^2 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_{t-1} \\ v_{t-1} \\ a_{t-1} \end{bmatrix} + noise$$

- Measurement is position only

$$y_t = M x_t + noise = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_t \\ v_t \\ a_t \end{bmatrix} + noise$$

Slide credit: Svetlana Lazebnik, Kristen Grauman

General Motion Models

- Assuming we have differential equations for the motion
  - E.g. for (undampened) periodic motion of a linear spring

$$\frac{d^2 p}{dt^2} = -p$$

- Substitute variables to transform this into linear system

$$p_1 = p \qquad p_2 = \frac{dp}{dt} \qquad p_3 = \frac{d^2 p}{dt^2}$$

- Then we have

$$x_t = \begin{bmatrix} p_{1,t} \\ p_{2,t} \\ p_{3,t} \end{bmatrix} \qquad \begin{aligned} p_{1,t} &= p_{1,t-1} + (\Delta t) p_{2,t-1} + \tfrac{1}{2} (\Delta t)^2 p_{3,t-1} + \varepsilon \\ p_{2,t} &= p_{2,t-1} + (\Delta t) p_{3,t-1} + \xi \\ p_{3,t} &= -p_{1,t-1} + \zeta \end{aligned} \qquad D_t = \begin{bmatrix} 1 & \Delta t & \tfrac{1}{2}(\Delta t)^2 \\ 0 & 1 & \Delta t \\ -1 & 0 & 0 \end{bmatrix}$$

The Kalman Filter

- Kalman filter
  - Method for tracking linear dynamical models in Gaussian noise
- The predicted/corrected state distributions are Gaussian
  - You only need to maintain the mean and covariance.
  - The calculations are easy (all the integrals can be done in closed form).

Slide credit: Svetlana Lazebnik

The Kalman Filter

Know corrected state from previous time step, and all measurements up to the current one
→ Predict distribution over next state.

Receive measurement

Know prediction of state, and next measurement
→Update distribution over current state.

Time update ("Predict")	Measurement update ("Correct")

$$P\left(X_t \middle| y_0, \ldots, y_{t-1}\right)$$

$$P\left(X_t \middle| y_0, \ldots, y_t\right)$$

Mean and std. dev. of predicted state:

$$\mu_t^-, \sigma_t^-$$

Time advances: t++

Mean and std. dev. of corrected state:

$$\mu_t^+, \sigma_t^+$$

Slide credit: Kristen Grauman

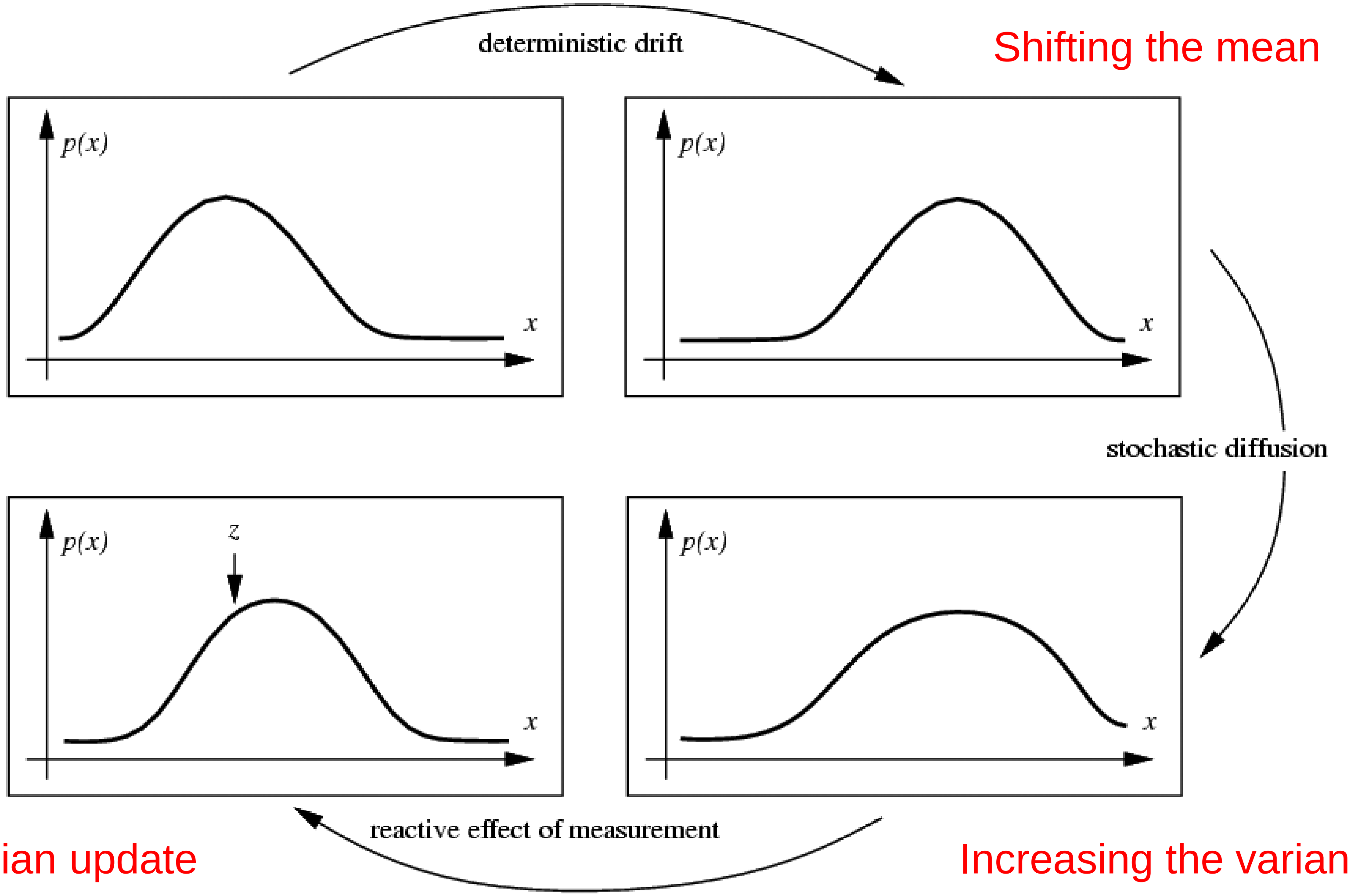
Kalman Filter for 1D State

- Want to represent and update

$$P\left(x_t \middle| y_0, \ldots, y_{t-1}\right) = N\left(\mu_t^-, (\sigma_t^-)^2\right)$$

$$P\left(x_t \middle| y_0, \ldots, y_t\right) = N\left(\mu_t^+, (\sigma_t^+)^2\right)$$

Propagation of Gaussian densities



Shifting the mean

Increasing the variance

Bayesian update

1D Kalman Filter: Prediction

- Have linear dynamic model defining predicted state evolution, with noise

$$X_t \sim N\left(dx_{t-1}, \sigma_d^2\right)$$

- Want to estimate predicted distribution for next state

$$P\left(X_t \middle| y_0, \ldots, y_{t-1}\right) = N\left(\mu_t^-, (\sigma_t^-)^2\right)$$

- Update the mean:

$$\mu_t^- = d\mu_{t-1}^+$$

for derivations, see F&P Chapter 17.3

- Update the variance:

$$(\sigma_t^-)^2 = \sigma_d^2 + (d\sigma_{t-1}^+)^2$$

Slide credit: Kristen Grauman

1D Kalman Filter: Correction

- Have linear model defining the mapping of state to measurements:

$$Y_t \sim N\left(mx_t, \sigma_m^2\right)$$

- Want to estimate corrected distribution given latest measurement:

$$P\left(X_t \middle| y_0, \ldots, y_t\right) = N\left(\mu_t^+, (\sigma_t^+)^2\right)$$

- Update the mean:

$$\mu_t^+ = \frac{\mu_t^- \sigma_m^2 + m y_t (\sigma_t^-)^2}{\sigma_m^2 + m^2 (\sigma_t^-)^2}$$

- Update the variance:

$$(\sigma_t^+)^2 = \frac{\sigma_m^2 (\sigma_t^-)^2}{\sigma_m^2 + m^2 (\sigma_t^-)^2}$$

Slide credit: Kristen Grauman

Derivations: F&P Chapter 17.3

Prediction vs. Correction

$$\mu_t^+ = \frac{\mu_t^- \sigma_m^2 + m y_t (\sigma_t^-)^2}{\sigma_m^2 + m^2 (\sigma_t^-)^2} \qquad (\sigma_t^+)^2 = \frac{\sigma_m^2 (\sigma_t^-)^2}{\sigma_m^2 + m^2 (\sigma_t^-)^2}$$

- What if there is no prediction uncertainty $(\sigma_t^- = 0)$?

$$\mu_t^+ = \mu_t^- \qquad (\sigma_t^+)^2 = 0$$

The measurement is ignored!

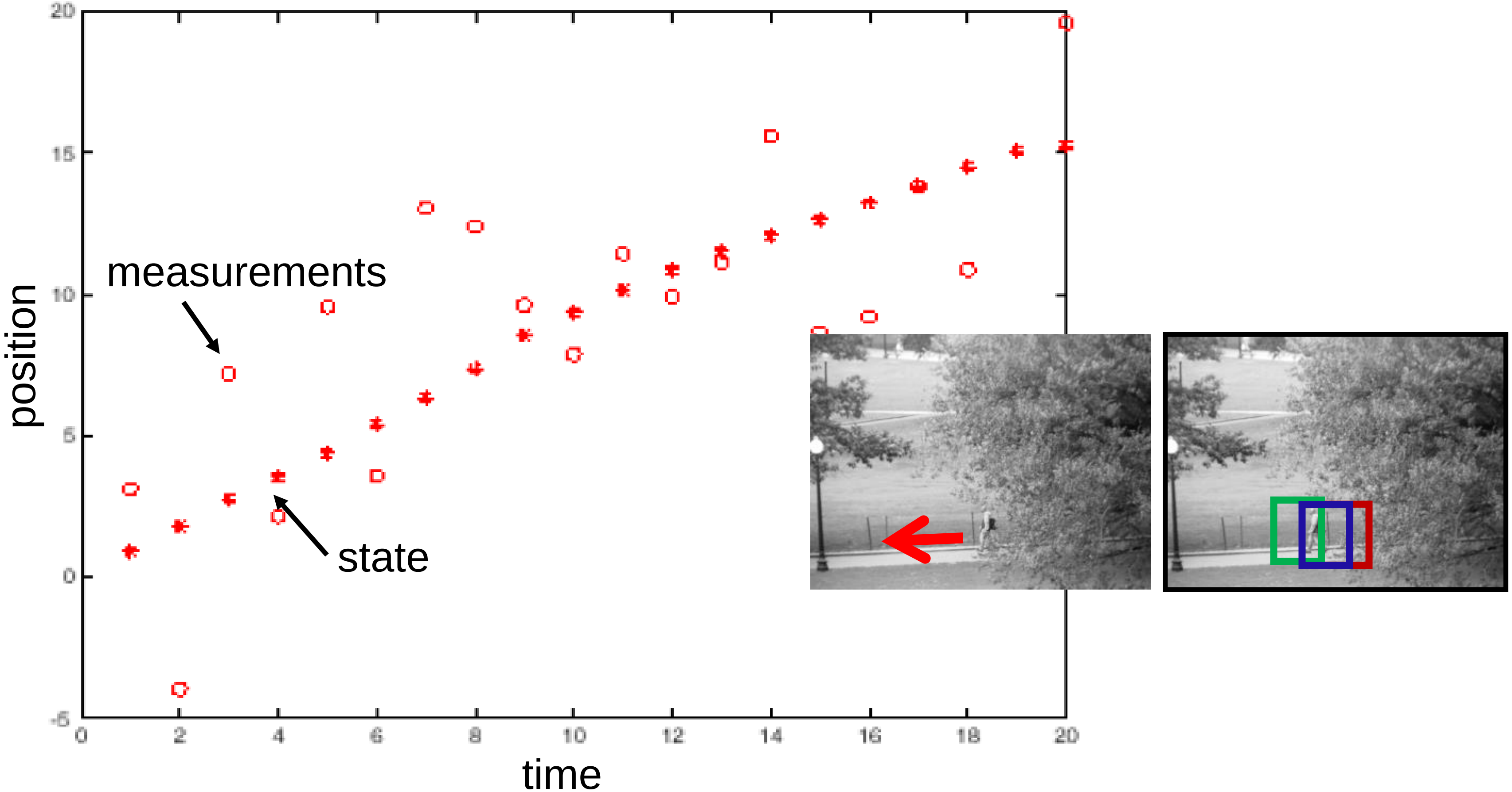
- What if there is no measurement uncertainty $(\sigma_m = 0)$?

$$\mu_t^+ = \frac{y_t}{m} \qquad (\sigma_t^+)^2 = 0$$

The prediction is ignored!

Slide credit: Kristen Grauman

Recall: Constant Velocity Example

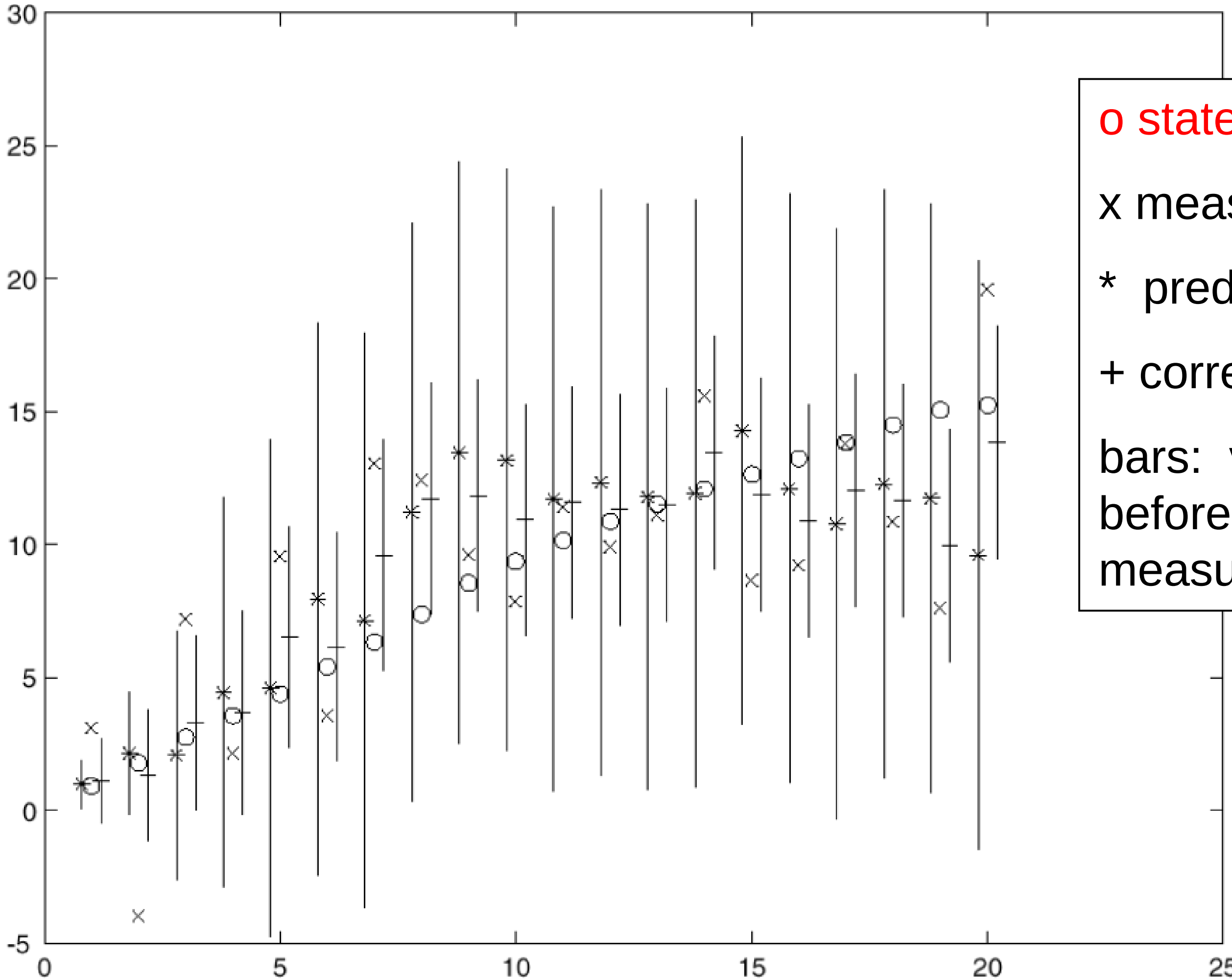


State is 2D: position + velocity
Measurement is 1D: position

Slide credit: Kristen Grauman

Figure from Forsyth & Ponce

Constant Velocity Model



o state

x measurement

* predicted mean estimate

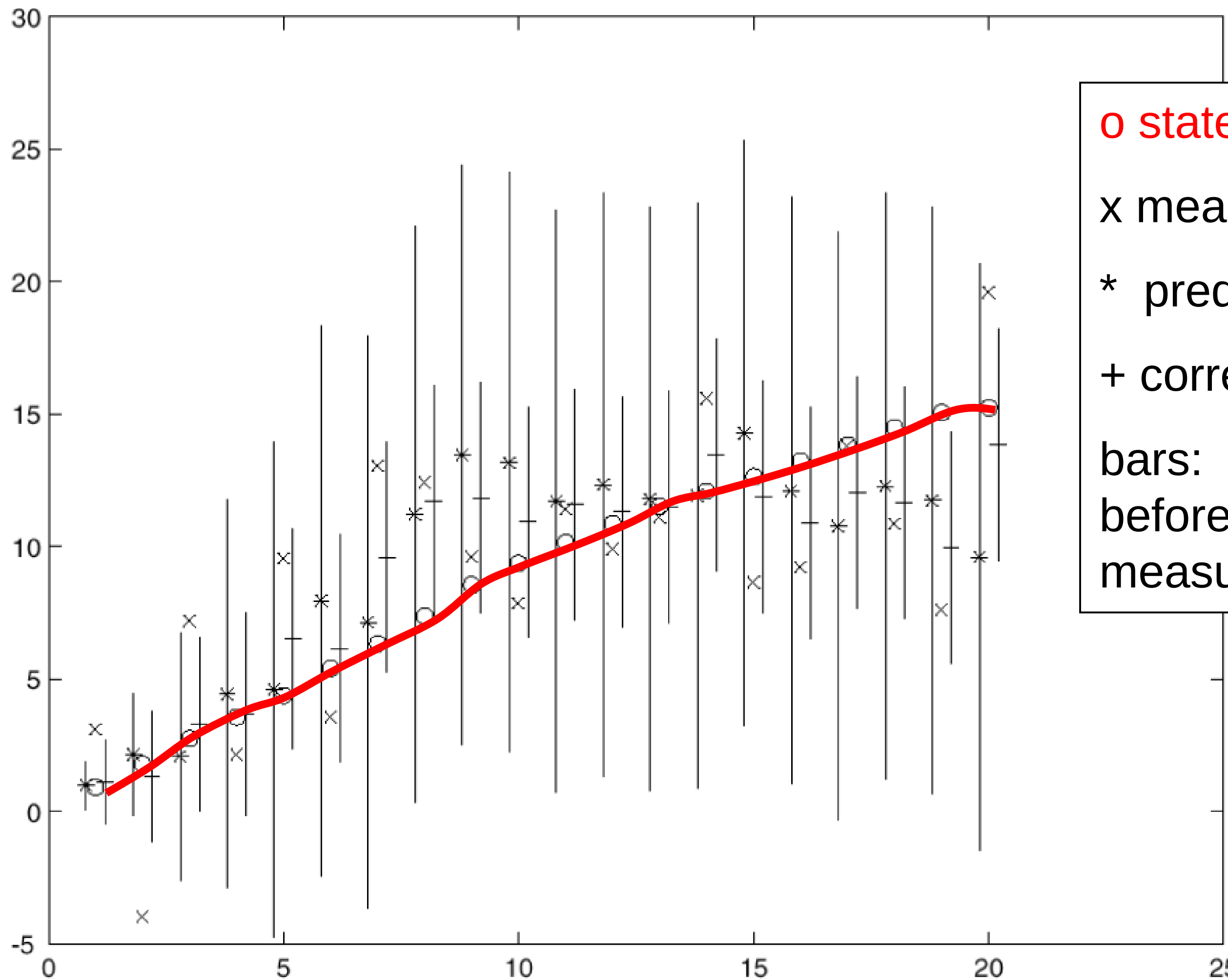
+ corrected mean estimate

bars: variance estimates before and after measurements

Slide credit: Kristen Grauman

Figure from Forsyth & Ponce

Constant Velocity Model



o state

x measurement

* predicted mean estimate

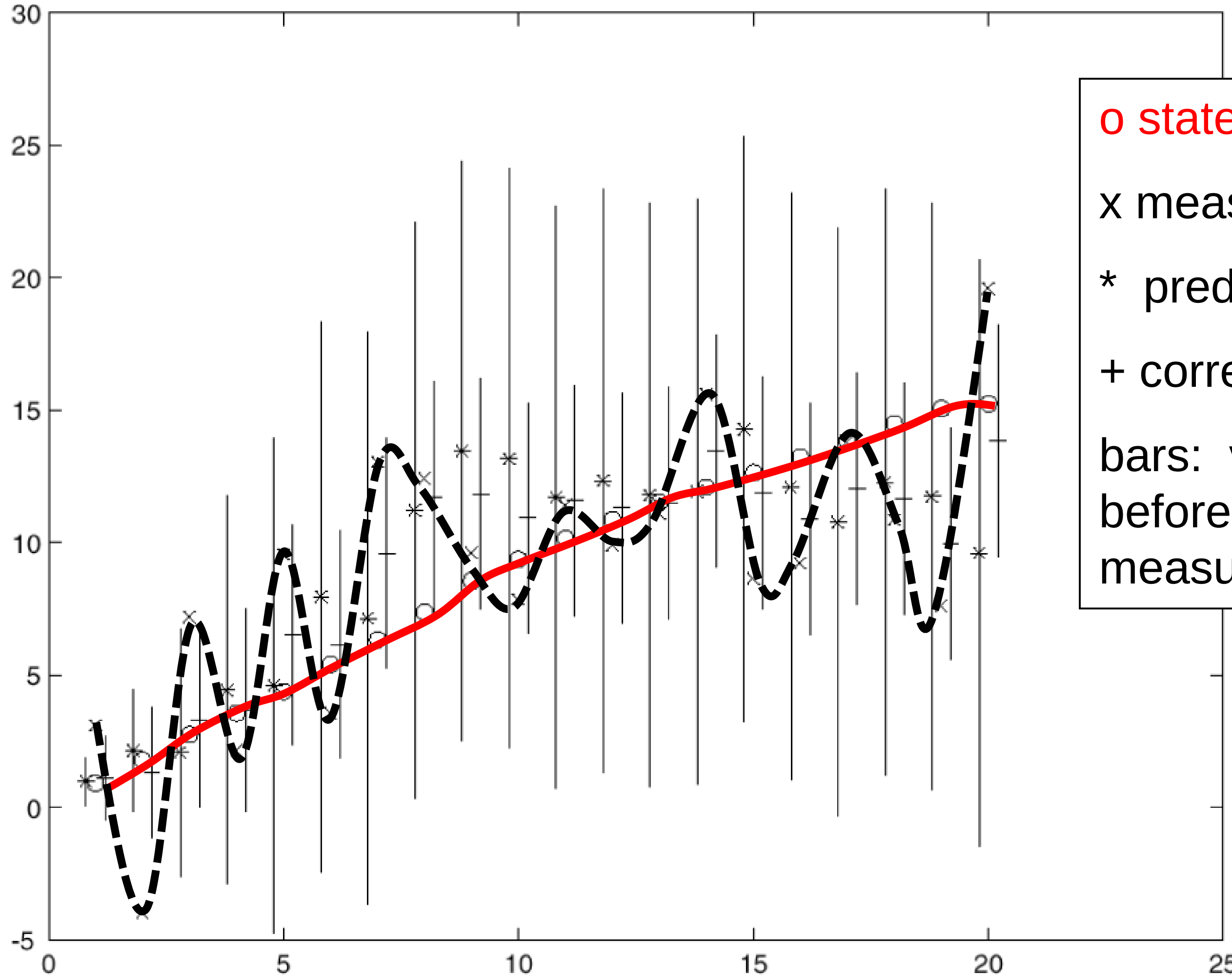
+ corrected mean estimate

bars: variance estimates before and after measurements

Slide credit: Kristen Grauman

Figure from Forsyth & Ponce

Constant Velocity Model



o state

x measurement

* predicted mean estimate

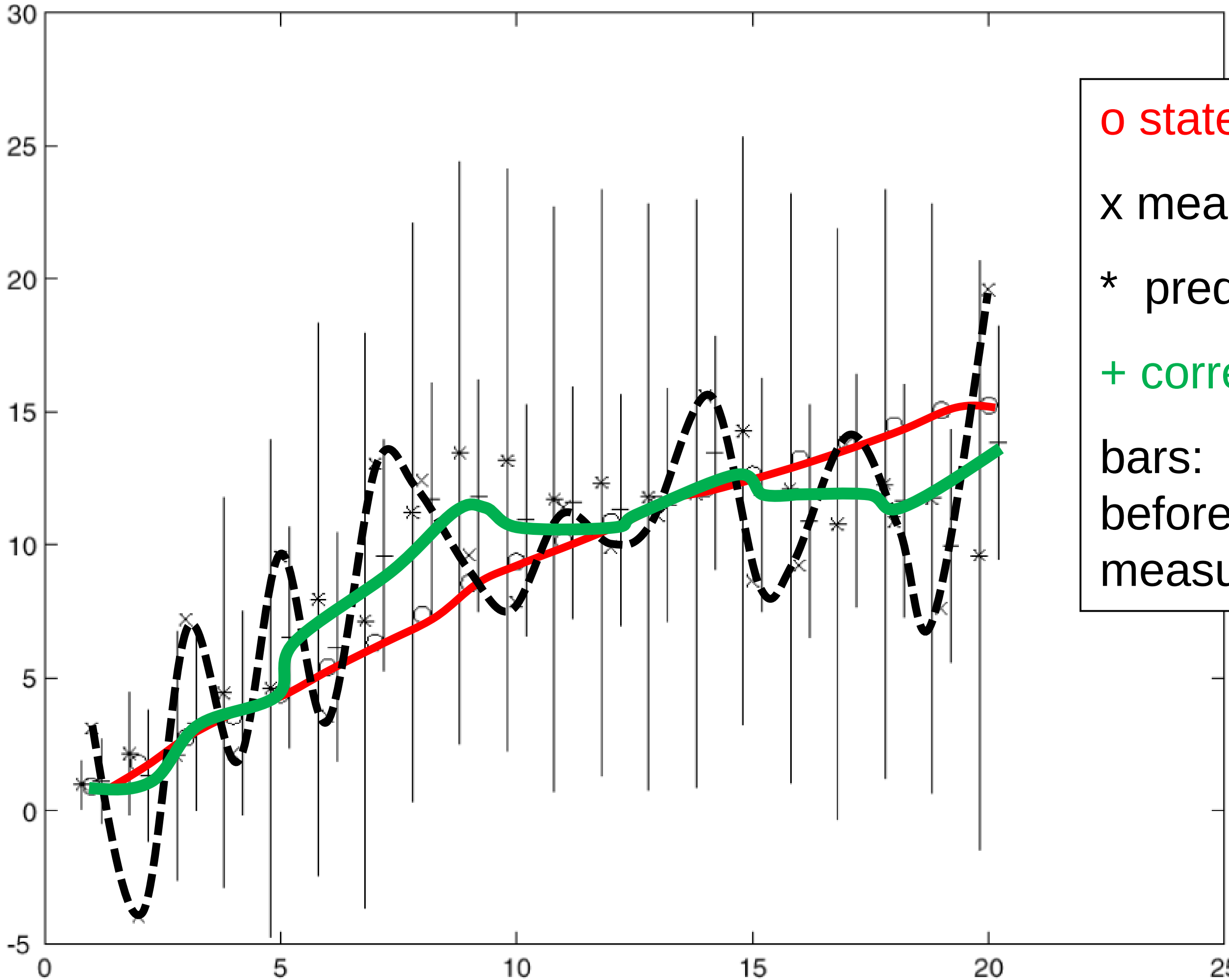
+ corrected mean estimate

bars: variance estimates before and after measurements

Slide credit: Kristen Grauman

Figure from Forsyth & Ponce

Constant Velocity Model



o state

x measurement

* predicted mean estimate

+ corrected mean estimate

bars: variance estimates before and after measurements

Slide credit: Kristen Grauman

Figure from Forsyth & Ponce

Kalman Filter: General Case (>1dim)

PREDICT

$$x_t^- = D_t x_{t-1}^+$$

$$\Sigma_t^- = D_t \Sigma_{t-1}^+ D_t^T + \Sigma_{d_t}$$

CORRECT

$$K_t = \Sigma_t^- M_t^T \left(M_t \Sigma_t^- M_t^T + \Sigma_{m_t} \right)^{-1}$$

$$x_t^+ = x_t^- + K_t \left(y_t - M_t x_t^- \right)$$

"residual"

"Kalman gain"

$$\Sigma_t^+ = \left(I - K_t M_t \right) \Sigma_t^-$$

More weight on residual when measurement error covariance approaches 0.

Less weight on residual as a priori estimate error covariance approaches 0.

for derivations, see F&P Chapter 17.3

Slide credit: Kristen Grauman

Summary: Kalman Filter

- Pros:
  - Gaussian densities everywhere
  - Simple updates, compact and efficient
  - Very established method, very well understood
- Cons:
  - Unimodal distribution, only single hypothesis
  - Restricted class of motions defined by linear model

Slide adapted from Svetlana Lazebnik

Remarks

- Try it!
  - Not too hard to understand or program
- Start simple
  - Experiment in 1D
  - Make your own filter in Matlab, etc.
- Note: the Kalman filter "wants to work"
  - Debugging can be difficult
  - Errors can go un-noticed

Slide adapted from Greg Welch

Topics of This Lecture

- Recap: Kalman Filter
  - Basic ideas
  - Kalman filter for 1D state
  - General Kalman filter
  - Limitations
  - Extensions
- Particle Filters
  - Basic ideas
  - Propagation of general densities
  - Factored sampling

Extension: Extended Kalman Filter (EKF)

- Basic idea
  - State transition and observation model don't need to be linear functions of the state, but just need to be differentiable.

$$x_t = g(x_{t-1}, u_t) + \varepsilon$$

$$y_t = h(x_t) + \delta$$

  - The EKF essentially linearizes the nonlinearity around the current estimate by a Taylor expansion.
- Properties
  - Unlike the linear KF, the EKF is in general *not* an optimal estimator.
    - If the initial estimate is wrong, the filter may quickly diverge.
  - Still, it's the de-facto standard in many applications
    - Including navigation systems and GPS

Recap: Kalman Filter – Detailed Algorithm

- Algorithm summary
  - Assumption: linear model

$$\begin{aligned}\mathbf{x}_t &= \mathbf{D}_t\mathbf{x}_{t-1} + \varepsilon_t\\ \mathbf{y}_t &= \mathbf{M}_t\mathbf{x}_t + \delta_t\end{aligned}$$

  - Prediction step

$$\begin{aligned}\mathbf{x}_t^- &= \mathbf{D}_t\mathbf{x}_{t-1}^+\\ \mathbf{\Sigma}_t^- &= \mathbf{D}_t\mathbf{\Sigma}_{t-1}^+\mathbf{D}_t^T + \mathbf{\Sigma}_{d_t}\end{aligned}$$

  - Correction step

$$\begin{aligned}\mathbf{K}_t &= \mathbf{\Sigma}_t^-\mathbf{M}_t^T\left(\mathbf{M}_t\mathbf{\Sigma}_t^-\mathbf{M}_t^T + \mathbf{\Sigma}_{m_t}\right)^{-1}\\ \mathbf{x}_t^+ &= \mathbf{x}_t^- + \mathbf{K}_t\left(\mathbf{y}_t - \mathbf{M}_t\mathbf{x}_t^-\right)\\ \mathbf{\Sigma}_t^+ &= \left(\mathbf{I} - \mathbf{K}_t\mathbf{M}_t\right)\mathbf{\Sigma}_t^-\end{aligned}$$

Extended Kalman Filter (EKF)

- Algorithm summary
  - Nonlinear model

$$\mathbf{x}_t = \mathbf{g}\left(\mathbf{x}_{t-1}\right) + \varepsilon_t$$
$$\mathbf{y}_t = \mathbf{h}\left(\mathbf{x}_t\right) + \delta_t$$

with the Jacobians

  - Prediction step

$$\mathbf{x}_t^- = \mathbf{g}\left(\mathbf{x}_{t-1}^+\right)$$
$$\mathbf{\Sigma}_t^- = \mathbf{G}_t \mathbf{\Sigma}_{t-1}^+ \mathbf{G}_t^T + \mathbf{\Sigma}_{d_t}$$

$$\mathbf{G}_t = \left.\frac{\partial \mathbf{g}(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}_{t-1}^+}$$

  - Correction step

$$\mathbf{K}_t = \mathbf{\Sigma}_t^- \mathbf{H}_t^T \left(\mathbf{H}_t \mathbf{\Sigma}_t^- \mathbf{H}_t^T + \mathbf{\Sigma}_{m_t}\right)^{-1}$$
$$\mathbf{x}_t^+ = \mathbf{x}_t^- + \mathbf{K}_t \left(\mathbf{y}_t - \mathbf{h}\left(\mathbf{x}_t^-\right)\right)$$
$$\mathbf{\Sigma}_t^+ = \left(\mathbf{I} - \mathbf{K}_t \mathbf{H}_t\right) \mathbf{\Sigma}_t^-$$

$$\mathbf{H}_t = \left.\frac{\partial \mathbf{h}(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}_t^-}$$

Kalman Filter – Other Extensions

- Unscented Kalman Filter (UKF)
  - Used for models with highly nonlinear predict and update functions.
  - Here, the EKF can give very poor performance, since the covariance is propagated through linearization of the non-linear model.
  - Idea (UKF): Propagate just a few sample points ("*sigma points*") around the mean exactly, then recover the covariance from them.
  - More accurate results than the EKF's Taylor expansion approximation.
- Ensemble Kalman Filter (EnKF)
  - Represents the distribution of the system state using a collection (an *ensemble*) of state vectors.
  - Replace covariance matrix by *sample covariance* from ensemble.
  - Still basic assumption that all prob. distributions involved are Gaussian.
  - EnKFs are especially suitable for problems with a large number of variables.

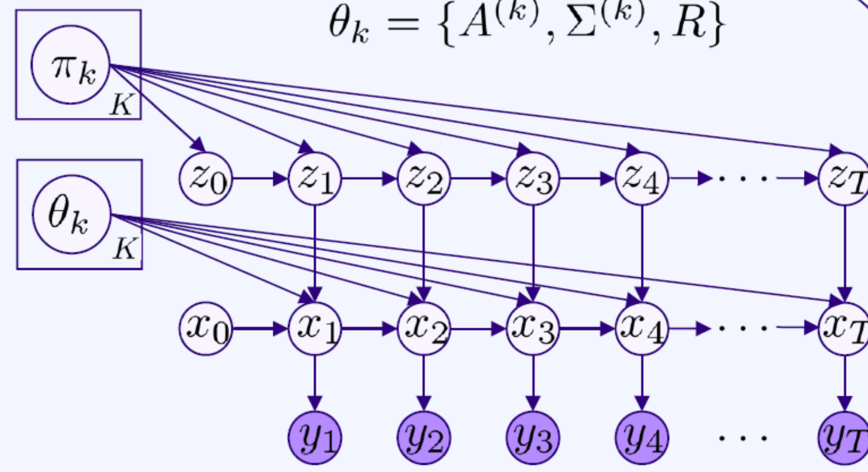
Even More Extensions

Switching linear dynamical system (SLDS):

$$z_t \sim \pi_{z_{t-1}}$$
$$x_t = A^{(z_t)} x_{t-1} + e_t(z_t)$$
$$y_t = C x_t + w_t$$

$$e_t \sim \mathcal{N}(0, \Sigma^{(z_t)}) \quad w_t \sim \mathcal{N}(0, R)$$

$$\theta_k = \{A^{(k)}, \Sigma^{(k)}, R\}$$



- Switching Linear Dynamic System (SLDS)
  - Use a set of k dynamic models $A^{(1)},\ldots,A^{(k)}$, each of which describes a different dynamic behavior.
  - Hidden variable z_t determines which model is active at time t.
  - A switching process can change z_t according to distribution $\pi_{z_{t-1}}$

Figure source: Erik Sudderth

Topics of This Lecture

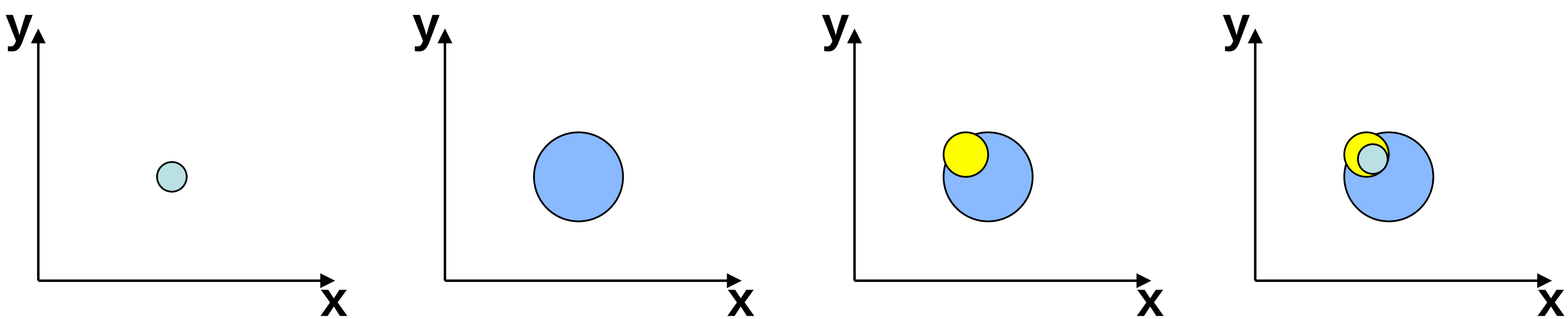
- Recap: Kalman Filter
  – Basic ideas
  – Kalman filter for 1D state
  – General Kalman filter
  – Limitations
  – Extensions
- Particle Filters
  – Basic ideas
  – Propagation of general densities
  – Factored sampling

Today: only main ideas

Formal introduction next lecture

When Is A Single Hypothesis Too Limiting?

Initial position | Prediction | Measurement | Update



- Consider this example: say we are tracking the face on the right using a skin color blob to get our measurement.

Video from Jojic & Frey

Slide credit: Kristen Grauman

Figure from Thrun & Kosecka

Propagation of General Densities

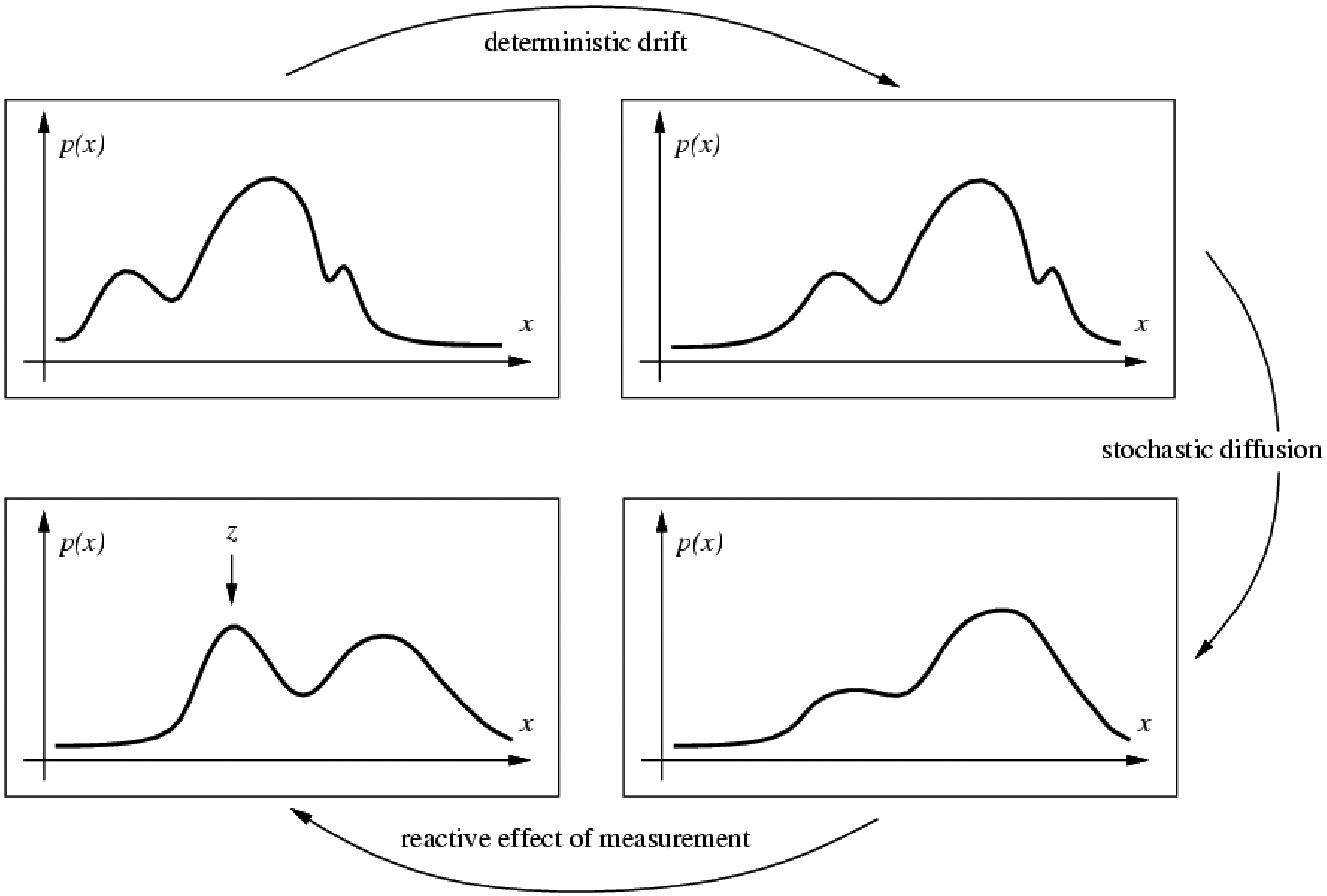
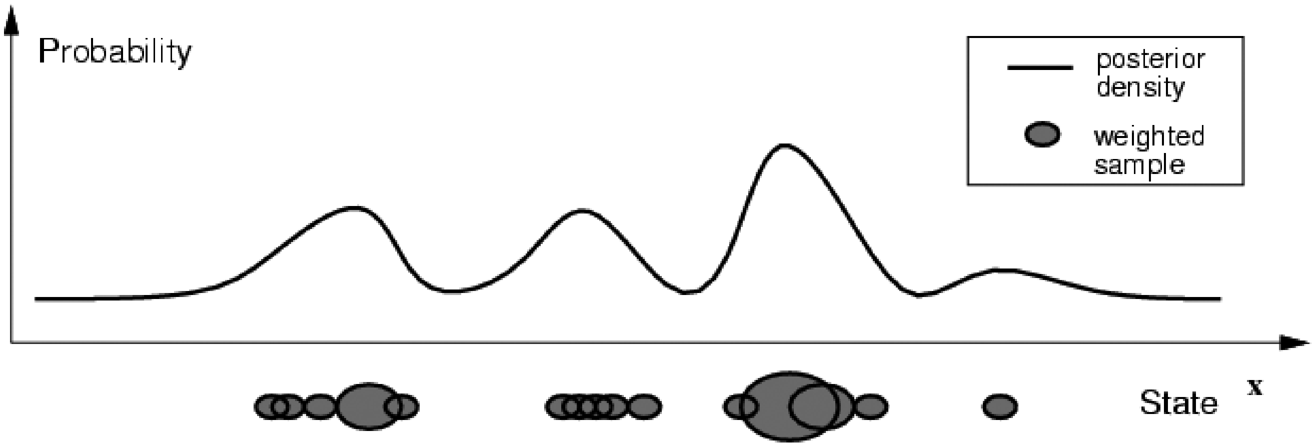


Figure from Isard & Blake

Factored Sampling



- Idea: Represent state distribution non-parametrically
  - Prediction: Sample points from prior density for the state, $P(X)$
  - Correction: Weight the samples according to $P(Y|X)$

$$P(X_t \mid y_0, \ldots, y_t) = \frac{P(y_t \mid X_t)P(X_t \mid y_0, \ldots, y_{t-1})}{\int P(y_t \mid X_t)P(X_t \mid y_0, \ldots, y_{t-1})dX_t}$$

Slide credit: Svetlana Lazebnik

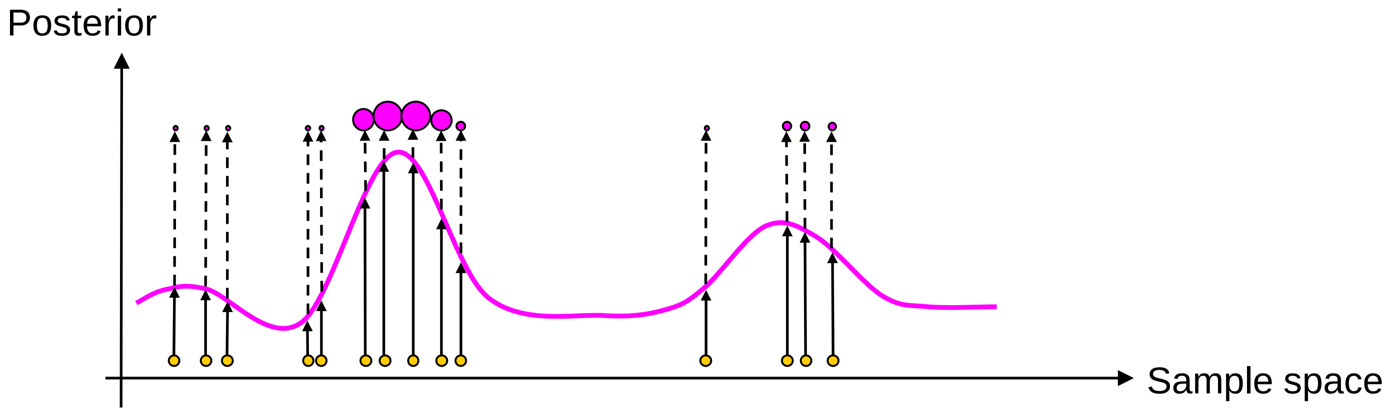
Particle Filtering

- (Also known as Sequential Monte Carlo Methods)

- Idea
  - We want to use sampling to propagate densities over time (i.e., across frames in a video sequence).
  - At each time step, represent posterior $P(X_t|Y_t)$ with weighted sample set.
  - Previous time step's sample set $P(X_{t\text{-}1}|Y_{t\text{-}1})$ is passed to next time step as the effective prior.

Particle Filtering

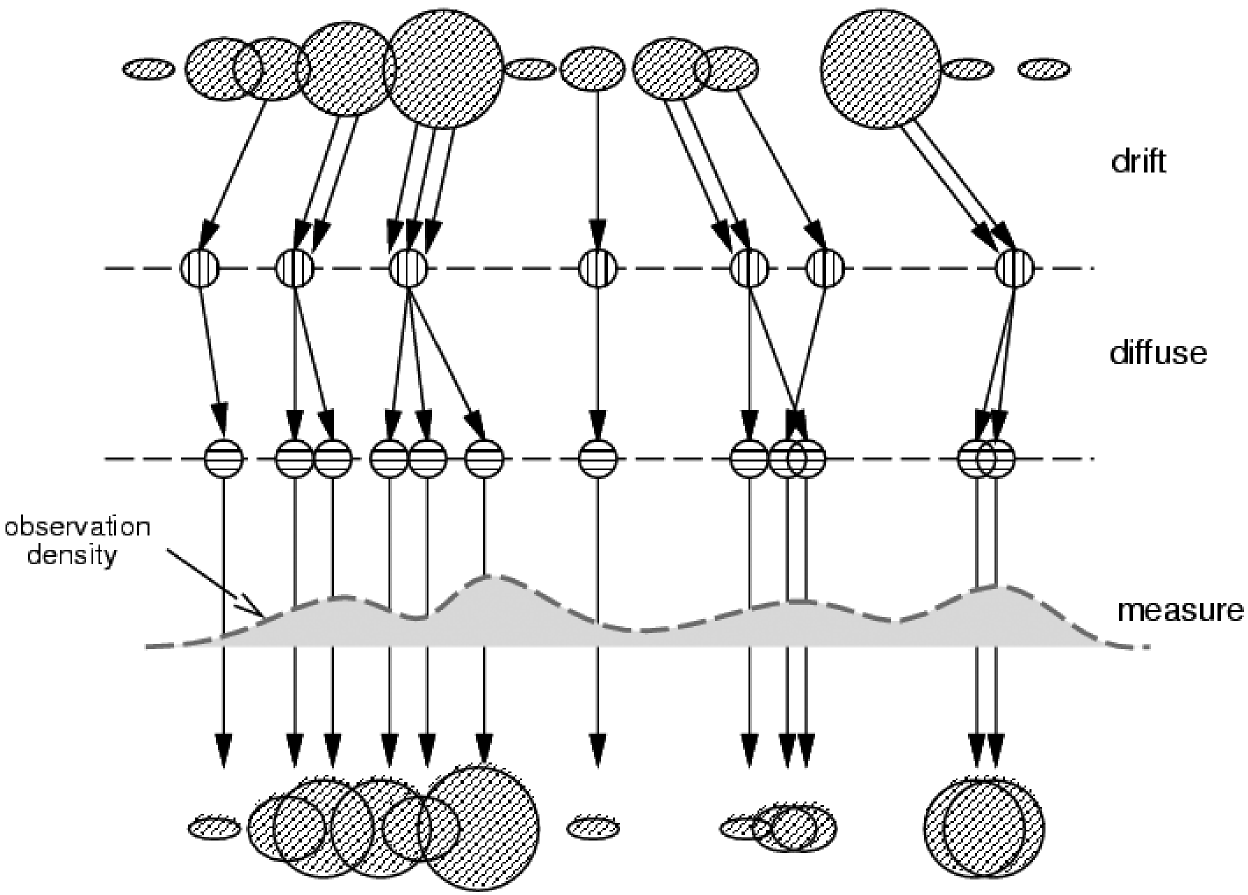
- Many variations, one general concept:
  - *Represent the posterior pdf by a set of randomly chosen weighted samples (particles)*



  - Randomly Chosen = Monte Carlo (MC)
  - As the number of samples become very large – the characterization becomes an equivalent representation of the true pdf.

Slide adapted from Michael Rubinstein

Particle Filtering



Start with weighted samples from previous time step

Sample and shift according to dynamics model

Spread due to randomness; this is predicted density $P(X_t|Y_{t-1})$

Weight the samples according to observation density

Arrive at corrected density estimate $P(X_t|Y_t)$

M. Isard and A. Blake, CONDENSATION -- conditional density propagation for visual tracking, IJCV 29(1):5-28, 1998

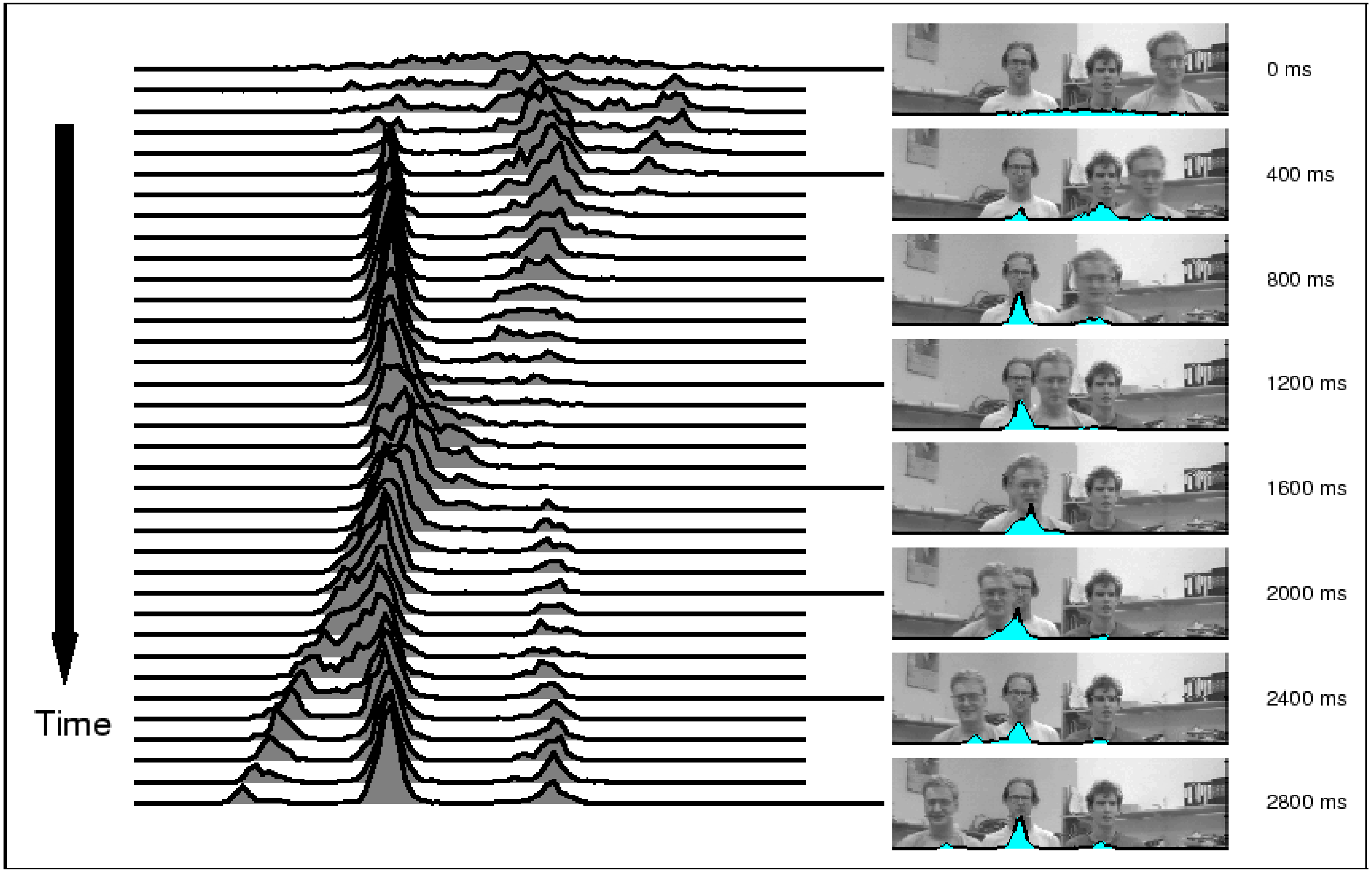
Slide credit: Svetlana Lazebnik

Particle Filtering – Visualization

Code and video available from

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

Particle Filtering Results



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

Particle Filtering Results

- Some more examples

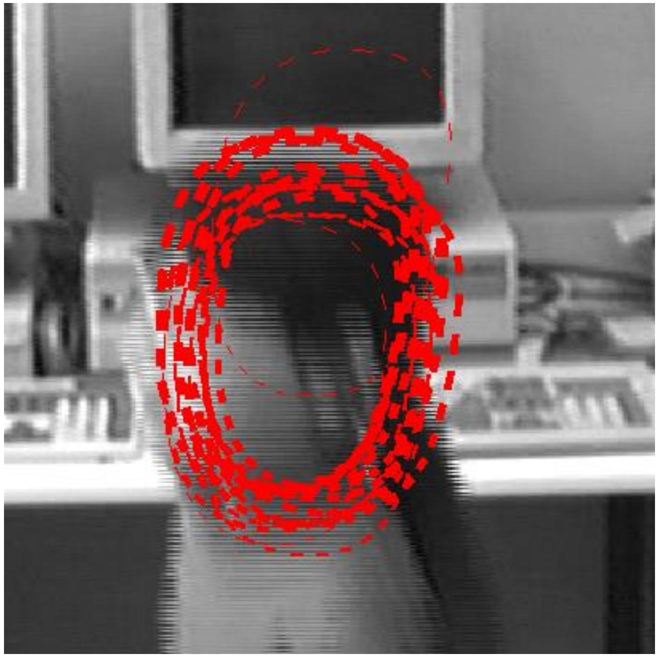
http://www.robots.ox.ac.uk/~misard/condensation.html

Videos from Isard & Blake

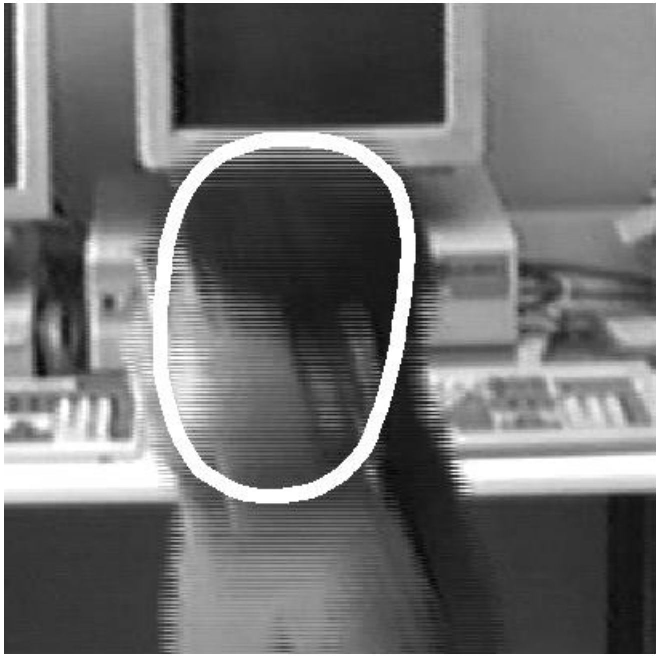
Obtaining a State Estimate

- Note that there’s no explicit state estimate maintained, just a “cloud” of particles
- Can obtain an estimate at a particular time by querying the current particle set
- Some approaches
  - “Mean” particle
    - Weighted sum of particles
    - Confidence: inverse variance
  - Really want a mode finder—mean of tallest peak

Condensation: Estimating Target State



State samples
(thickness proportional to weight)



From Isard & Blake, 1998

Mean of weighted
state samples

Slide credit: Marc Pollefeys

Figures from Isard & Blake

Summary: Particle Filtering

- Pros:
  - Able to represent arbitrary densities
  - Converging to true posterior even for non-Gaussian and nonlinear system
  - Efficient: particles tend to focus on regions with high probability
  - Works with many different state spaces
    - E.g. articulated tracking in complicated joint angle spaces
  - Many extensions available

Summary: Particle Filtering

- <u>Cons / Caveats:</u>
  - #Particles is important performance factor
    - Want as few particles as possible for efficiency.
    - But need to cover state space sufficiently well.
  - Worst-case complexity grows exponentially in the dimensions
  - Multimodal densities possible, but still single object
    - Interactions between multiple objects require special treatment.
    - Not handled well in the particle filtering framework
      (state space explosion).

References and Further Reading

- A good tutorial on Particle Filters
  - M.S. Arulampalam, S. Maskell, N. Gordon, T. Clapp. A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking. In *IEEE Transactions on Signal Processing*, Vol. 50(2), pp. 174-188, 2002.

- The CONDENSATION paper
  - M. Isard and A. Blake, CONDENSATION - conditional density propagation for visual tracking, IJCV 29(1):5-28, 1998