

# Tracking Applications

- Motion capture
- Recognition from motion
- Surveillance
- Targeting

# Things to consider in tracking

What are the

- Real world dynamics
- Approximate / assumed model
- Observation / measurement process

# Density propagation

- Tracking == Inference over time
- Much simplification is possible with linear dynamics and Gaussian probability models

# Outline

- Recursive filters
- State abstraction
- Density propagation
- Linear Dynamic models / Kalman filter
- Data association
- Multiple models

# Tracking and Recursive estimation

- Real-time / interactive imperative.
- Task: At each time point, re-compute estimate of position or pose.
  - At time  $n$ , fit model to data using time  $0\dots n$
  - At time  $n+1$ , fit model to data using time  $0\dots n+1$
- Repeat batch fit every time?

# Recursive estimation

- Decompose estimation problem
  - part that depends on new observation
  - part that can be computed from previous history

- E.g., running average:

$$a_t = \alpha a_{t-1} + (1-\alpha) y_t$$

- Linear Gaussian models: Kalman Filter
- First, general framework...

# 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)

# Three main issues in tracking

- **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

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

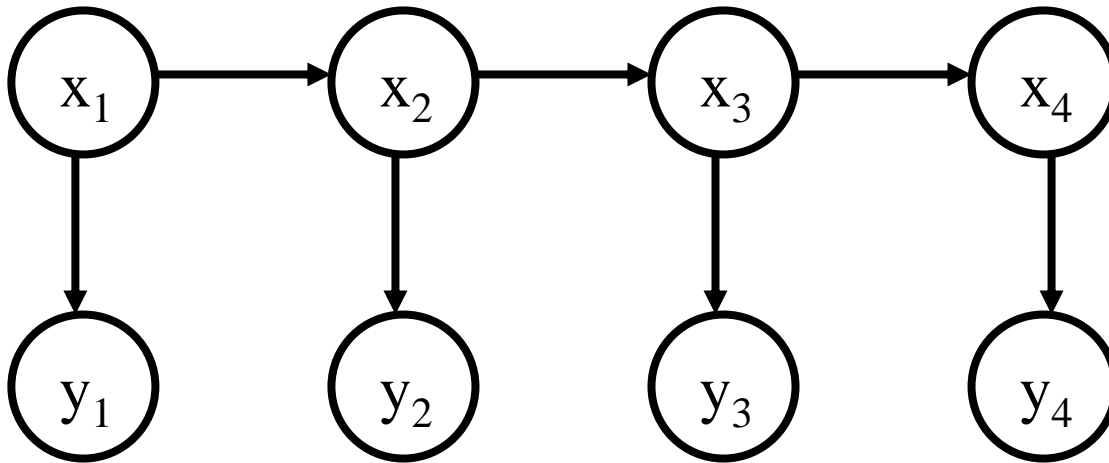
This assumption hugely simplifies the design of algorithms, as we shall see; furthermore, it isn't terribly restrictive if we're clever about interpreting  $\mathbf{X}_i$  as we shall show in the next section.

- **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.

# Kalman filter graphical model



# Tracking as induction

- Assume data association is done
  - we'll talk about this later; a dangerous assumption
- 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  $P(\mathbf{X}_0)$

$$P(\mathbf{X}_0 | \mathbf{Y}_0 = \mathbf{y}_0) = \frac{P(\mathbf{y}_0 | \mathbf{X}_0) P(\mathbf{X}_0)}{P(\mathbf{y}_0)}$$

$$\propto P(\mathbf{y}_0 | \mathbf{X}_0) P(\mathbf{X}_0)$$

# Induction step

## Prediction

Prediction involves representing

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

given

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

Our independence assumptions make it possible to write

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

# Update step

## Correction

Correction involves obtaining a representation of

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

given

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

Our independence assumptions make it possible to write

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

# Linear dynamic models

- 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

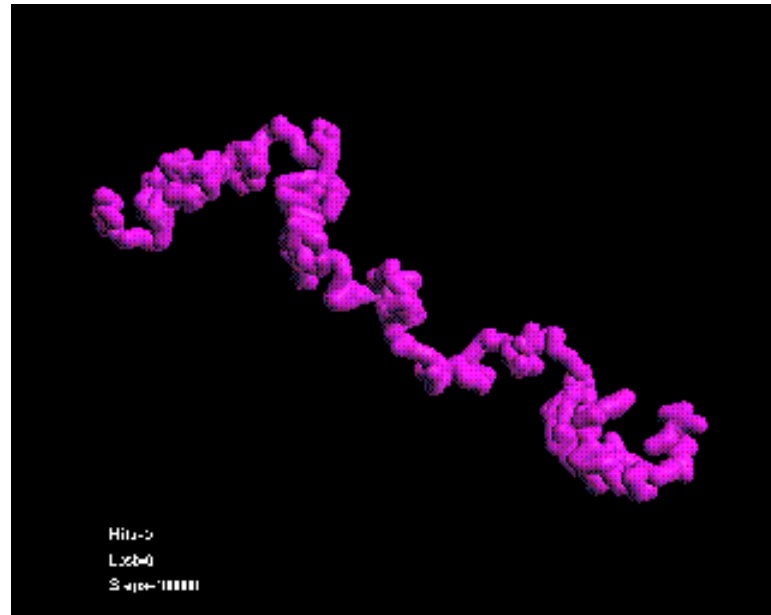
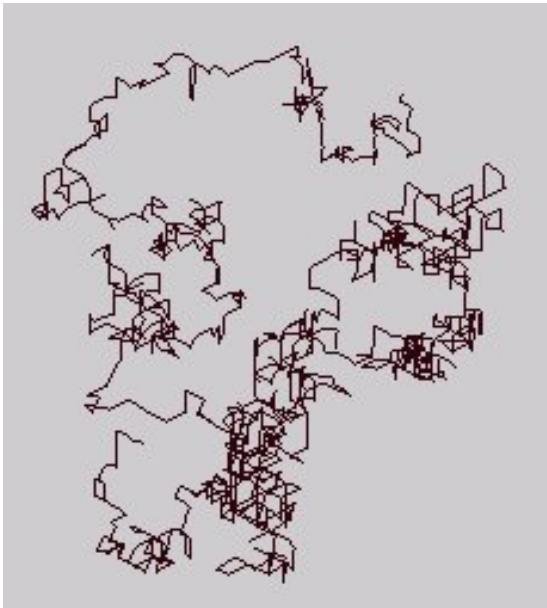
# Examples

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

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

- Drifting points
  - assume that the new position of the point is the old one, plus noise

$$\mathbf{D} = \mathbf{I}_d$$



# Constant velocity

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

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

- We have

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

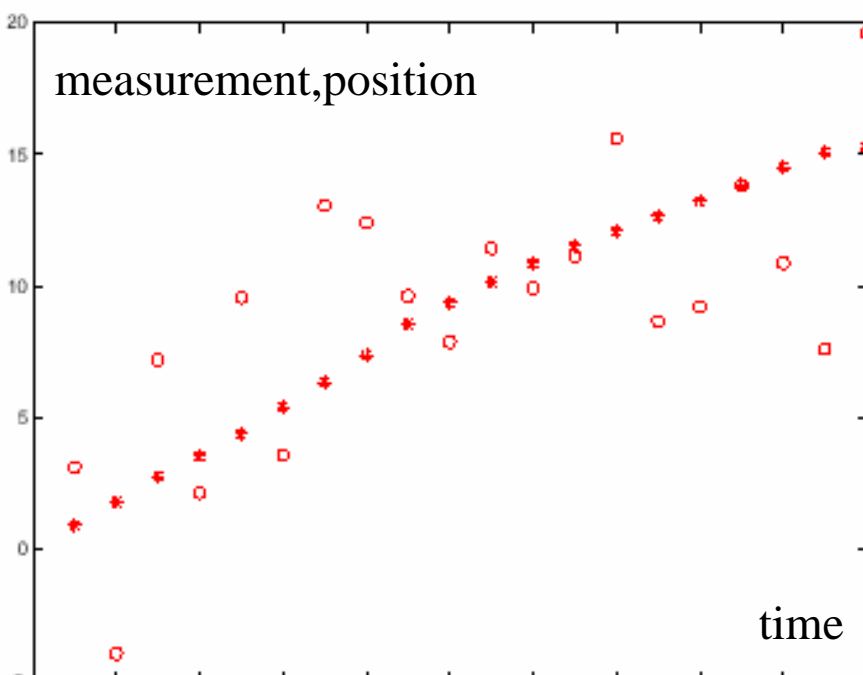
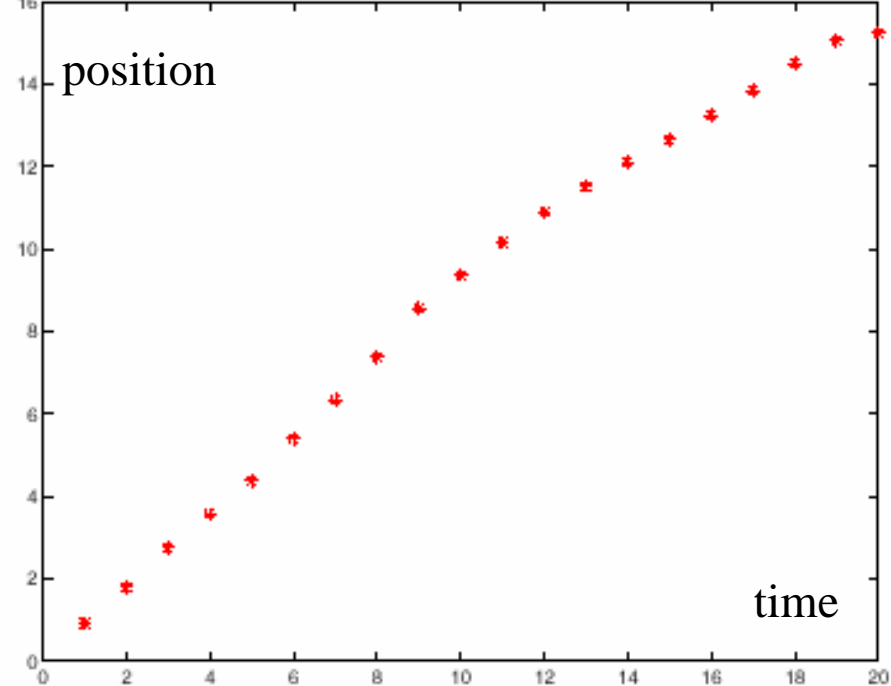
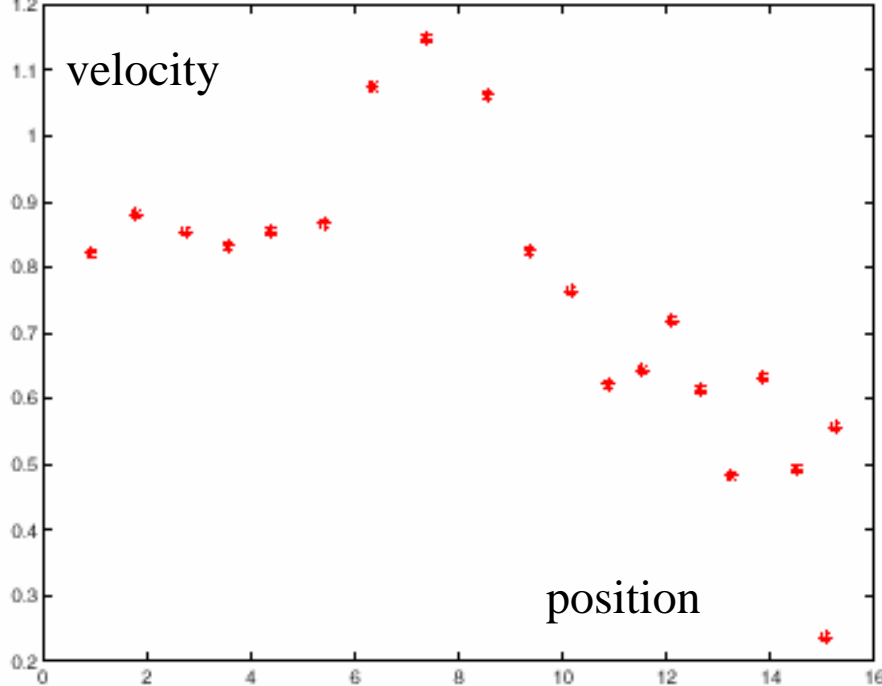
$$v_i = v_{i-1} + \zeta_i$$

– (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



Constant  
Velocity  
Model

# Constant acceleration

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

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

- We have

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

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

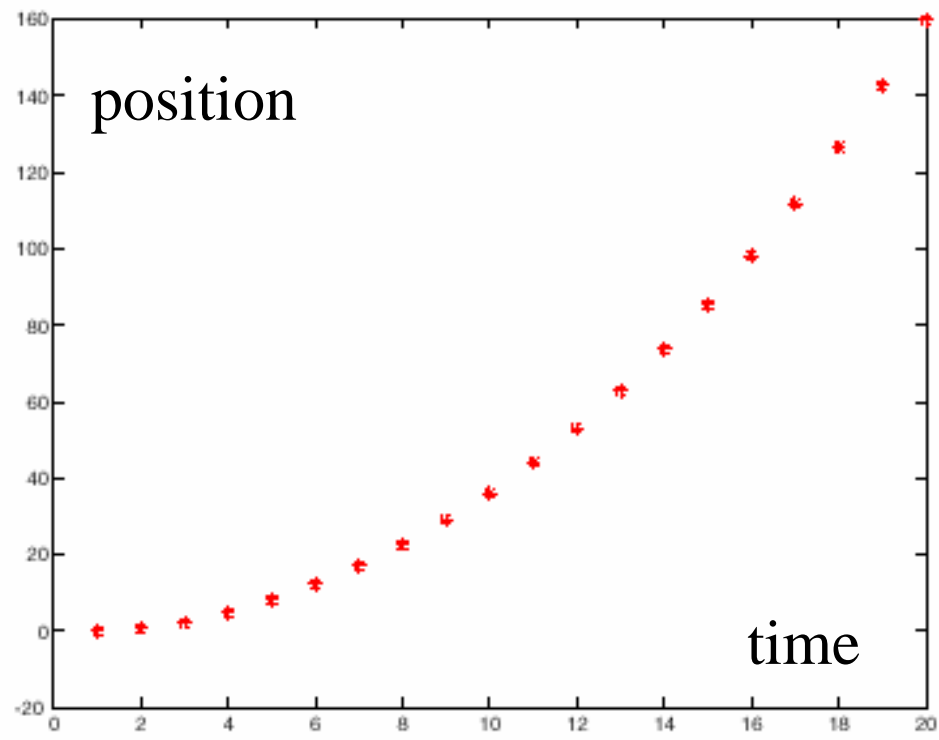
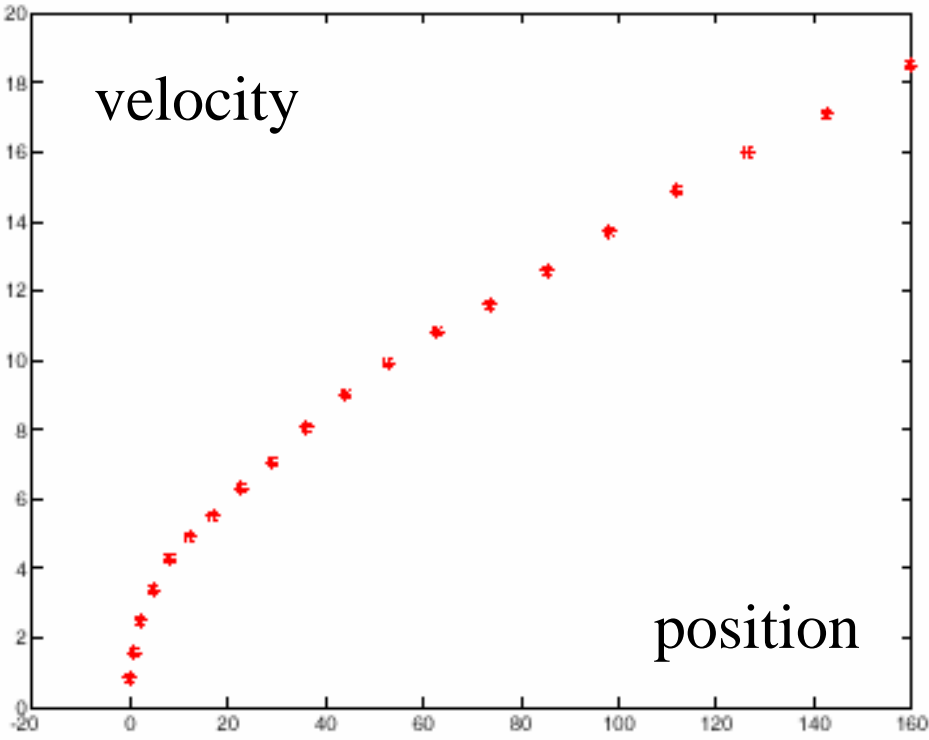
$$a_i = a_{i-1} + \xi_i$$

– (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



# Constant Acceleration Model

# Periodic motion

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

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

Assume we have a point, moving on a line with a periodic movement defined with a differential eq:

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

can be defined as

$$\frac{du}{dt} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} u = Su$$

with state defined as stacked position and velocity  $u=(p, v)$

# Periodic motion

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

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

$$\frac{du}{dt} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} u = Su$$

Take discrete approximation....(e.g., forward Euler integration with  $\Delta t$  stepsize.)

$$\begin{aligned} \mathbf{u}_i &= \mathbf{u}_{i-1} + \Delta t \frac{d\mathbf{u}}{dt} \\ &= \mathbf{u}_{i-1} + \Delta t S \mathbf{u}_{i-1} \\ &= \begin{pmatrix} 1 & \Delta t \\ -\Delta t & 1 \end{pmatrix} \mathbf{u}_{i-1} \end{aligned}$$

# Higher order models

- Independence assumption

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

- Velocity and/or acceleration augmented position
- Constant velocity model equivalent to

$$P(\mathbf{p}_i | \mathbf{p}_1, \dots, \mathbf{p}_{i-1}) = N(\mathbf{p}_{i-1} + (\mathbf{p}_{i-1} - \mathbf{p}_{i-2}), \Sigma_{d_i})$$

- velocity ==  $\mathbf{p}_{i-1} - \mathbf{p}_{i-2}$
- acceleration ==  $(\mathbf{p}_{i-1} - \mathbf{p}_{i-2}) - (\mathbf{p}_{i-2} - \mathbf{p}_{i-3})$
- could also use  $\mathbf{p}_{i-4}$ , etc.

# The Kalman Filter

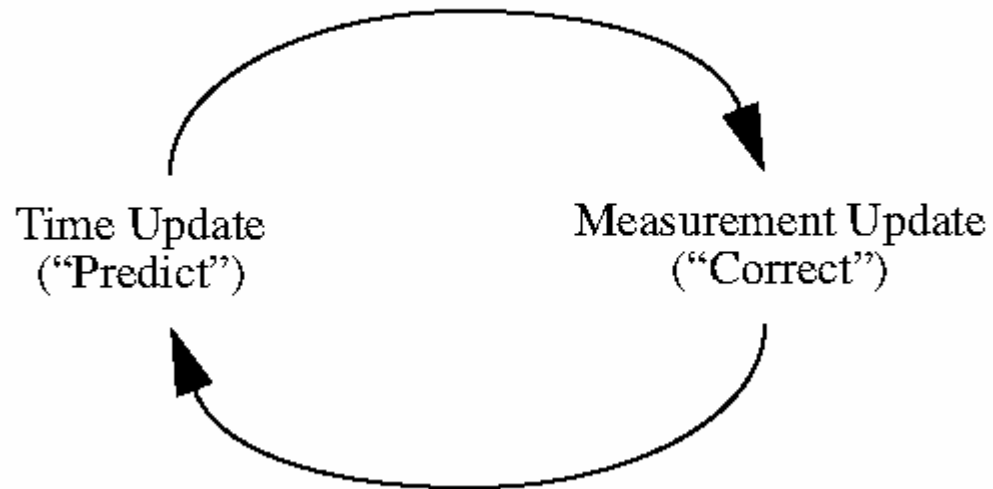
- Key ideas:
  - Linear models interact uniquely well with Gaussian noise - make the prior Gaussian, everything else Gaussian and the calculations are easy
  - Gaussians are really easy to represent --- once you know the mean and covariance, you're done

# Recall the three main issues in tracking

- **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  $P(\mathbf{X}_i | \mathbf{Y}_0 = \mathbf{y}_0, \dots, \mathbf{Y}_i = \mathbf{y}_i)$ .

*(Ignore data association for now)*

# The Kalman Filter



# The Kalman Filter in 1D

- Dynamic Model

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

- Notation

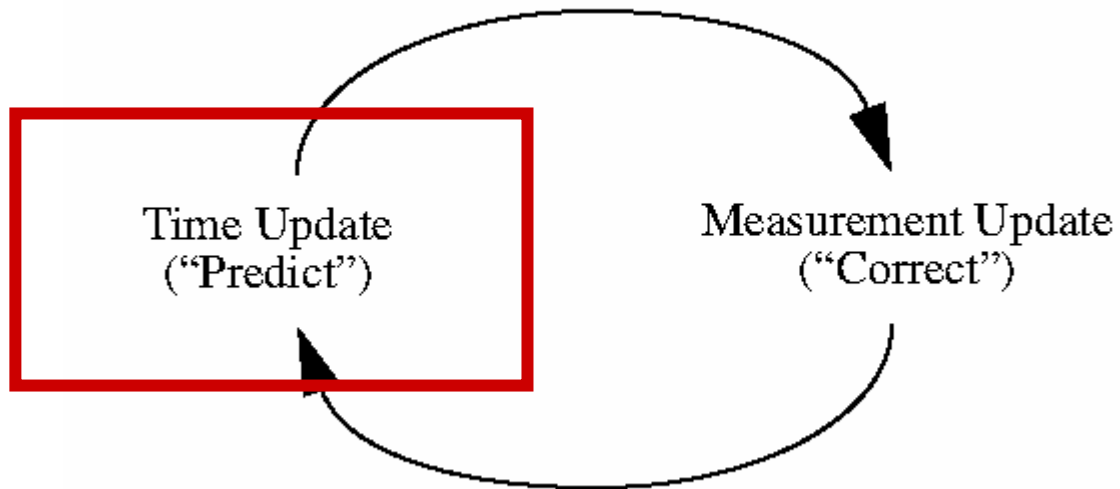
$$y_i \sim N(m_i x_i, \sigma_{m_i}^2)$$

mean of  $P(X_i | y_0, \dots, y_{i-1})$  as  $\bar{X}_i^-$  ← Predicted mean

mean of  $P(X_i | y_0, \dots, y_i)$  as  $\bar{X}_i^+$  ← Corrected mean

the standard deviation of  $P(X_i | y_0, \dots, y_{i-1})$  as  $\sigma_i^-$   
of  $P(X_i | y_0, \dots, y_i)$  as  $\sigma_i^+$ .

# The Kalman Filter



# 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
- Old variance is normal random variable
  - variance is multiplied by square of constant
  - and variance of noise is added.

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

$$\overline{X}_i^- = d_i \overline{X}_{i-1}^+$$

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

Dynamic Model:

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

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

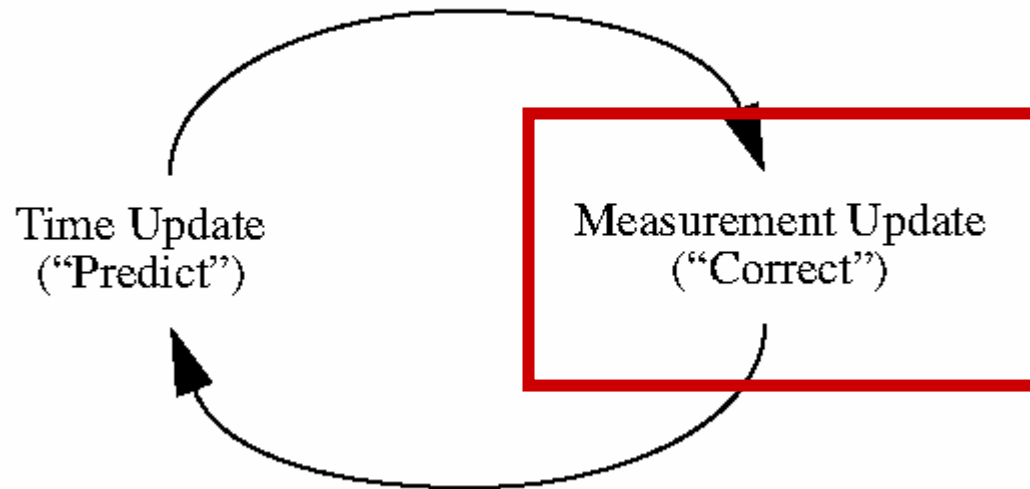
Start Assumptions:  $\bar{x}_0^-$  and  $\sigma_0^-$  are known

Update Equations: Prediction

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

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

# The Kalman Filter



# Correction for 1D Kalman filter

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

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

Notice:

- if measurement noise is small, we rely mainly on the measurement,
- if it's large, mainly on the prediction
- $\sigma$  does not depend on  $y$

Dynamic Model:

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

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

Start Assumptions:  $\bar{x}_0^-$  and  $\sigma_0^-$  are known

Update Equations: Prediction

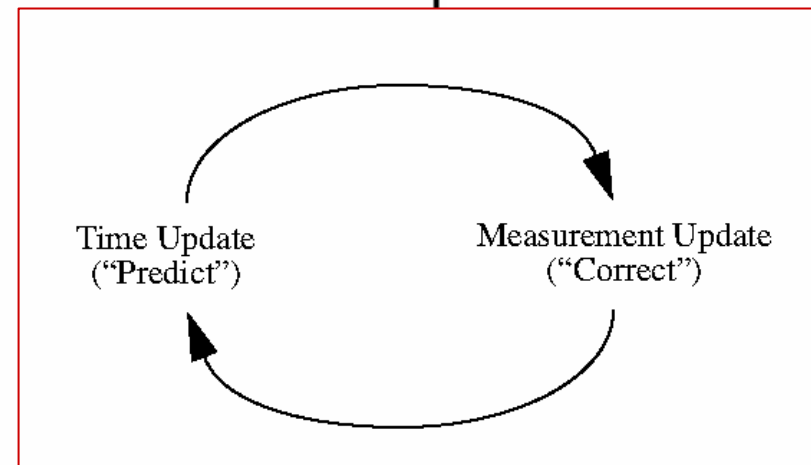
$$\bar{x}_i^- = d_i \bar{x}_{i-1}^+$$

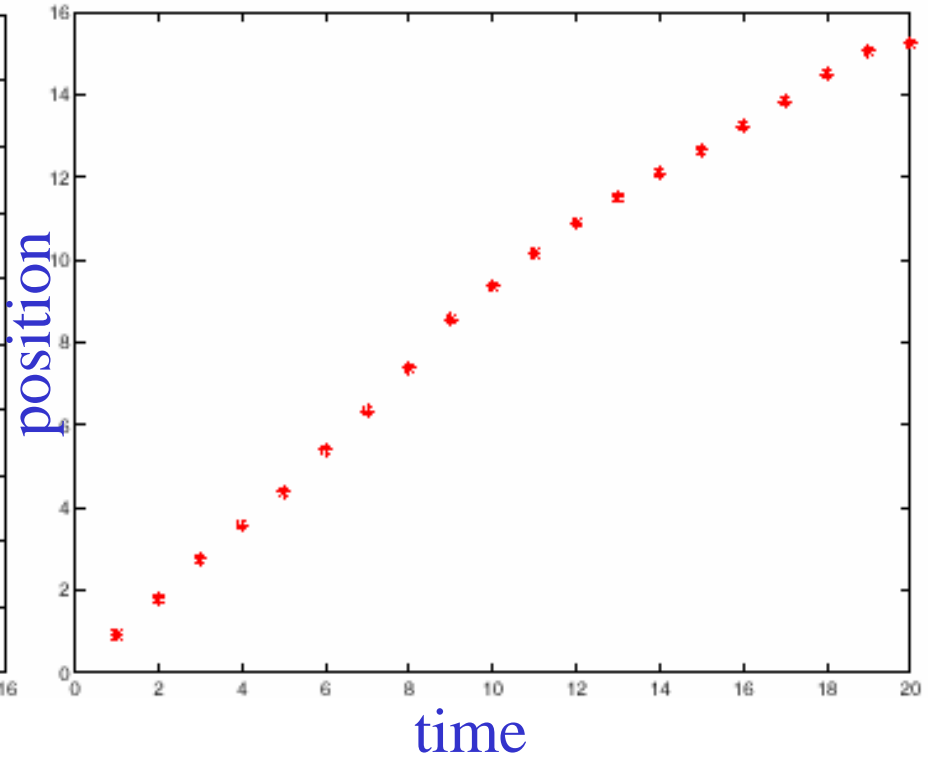
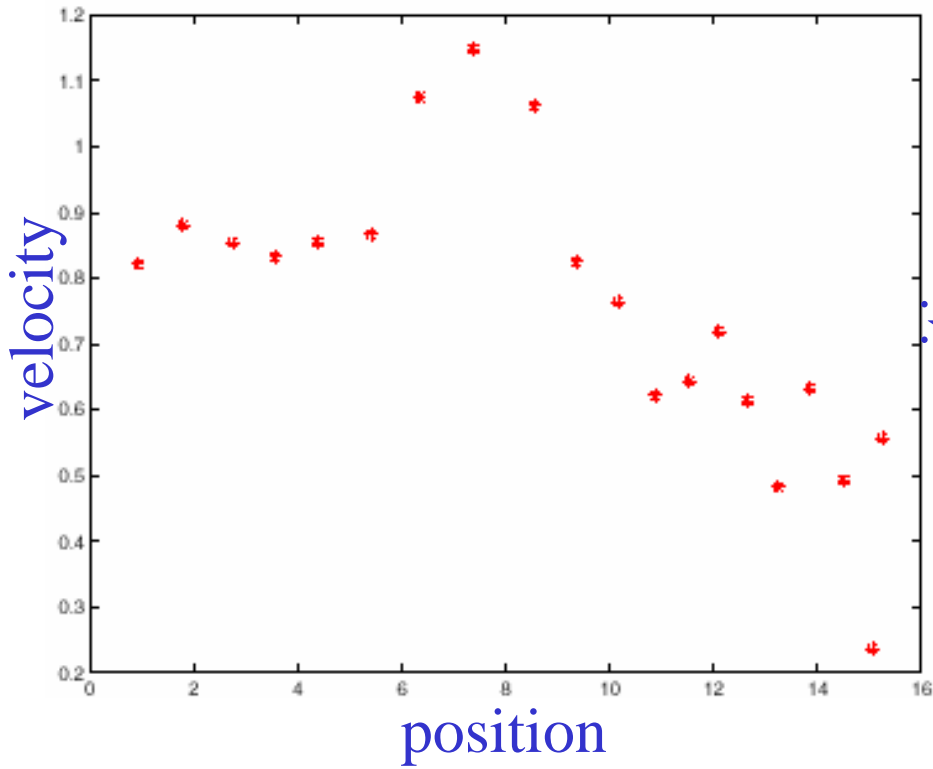
$$\sigma_i^- = \sqrt{\sigma_{d_i}^2 + (d_i \sigma_{i-1}^+)^2}$$

Update Equations: Correction

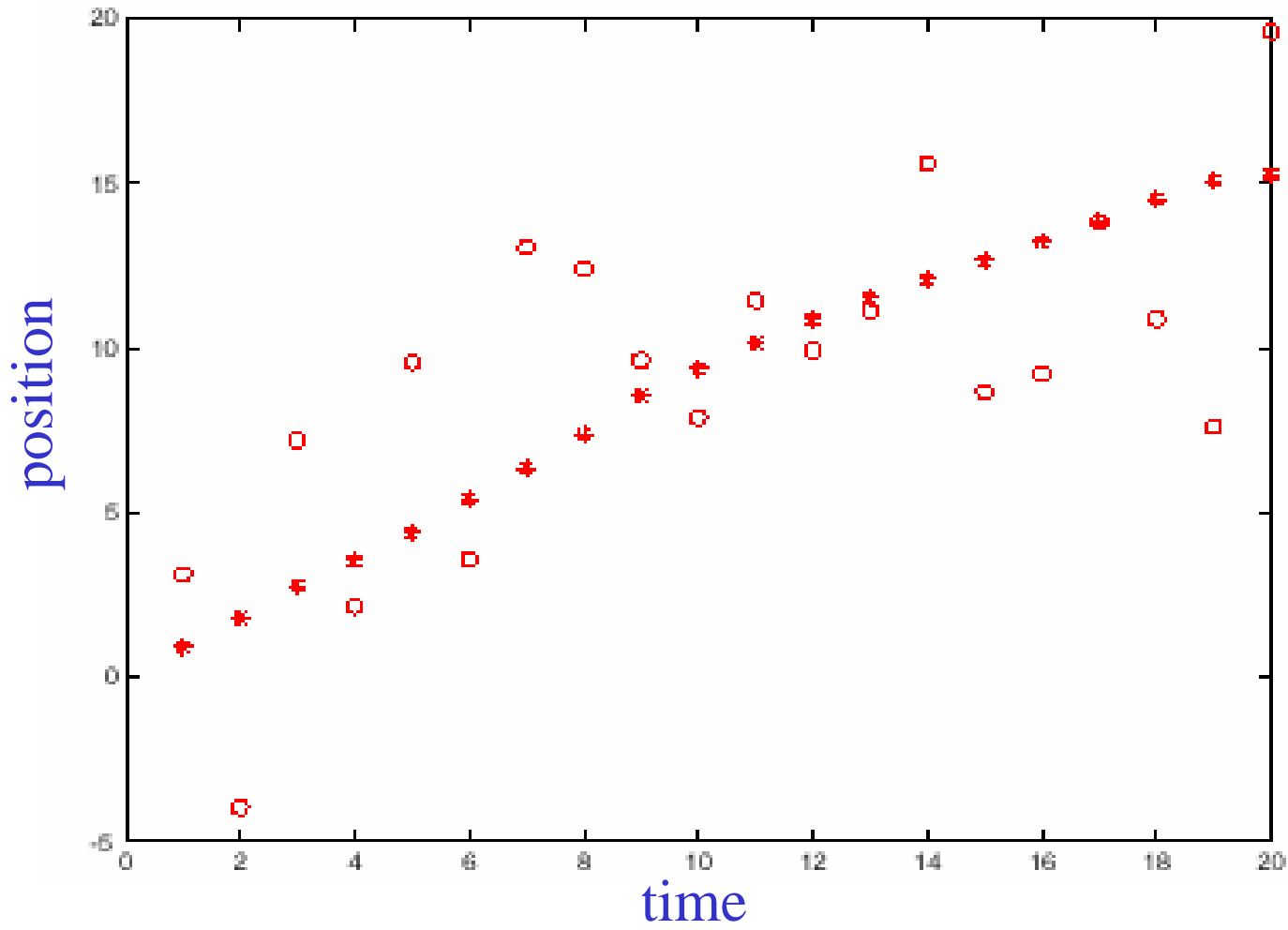
$$x_i^+ = \left( \frac{\bar{x}_i^- \sigma_{m_i}^2 + m_i y_i (\sigma_i^-)^2}{\sigma_{m_i}^2 + m_i^2 (\sigma_i^-)^2} \right)$$

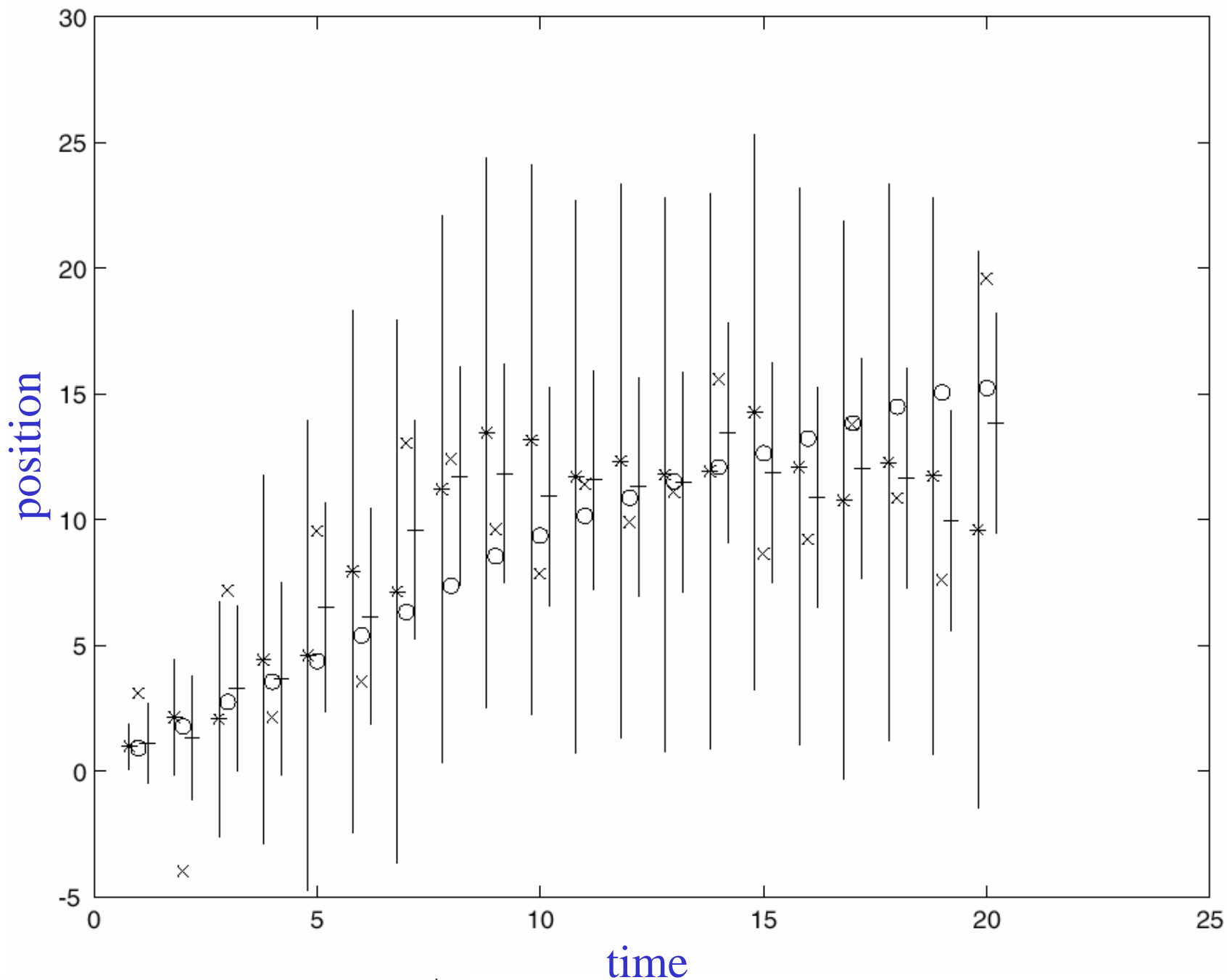
$$\sigma_i^+ = \sqrt{\left( \frac{\sigma_{m_i}^2 (\sigma_i^-)^2}{(\sigma_{m_i}^2 + m_i^2 (\sigma_i^-)^2)} \right)}$$



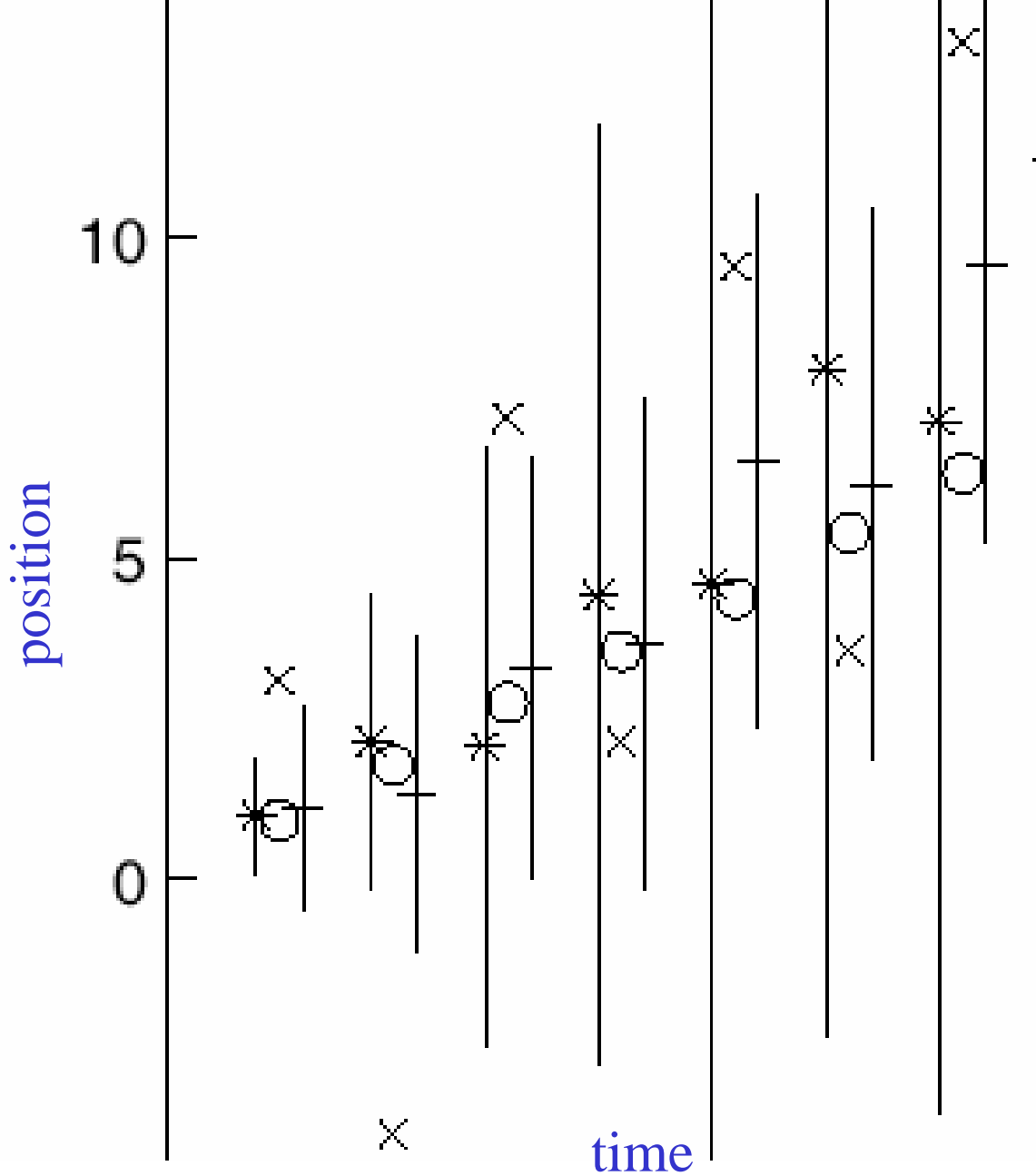


## Constant Velocity Model





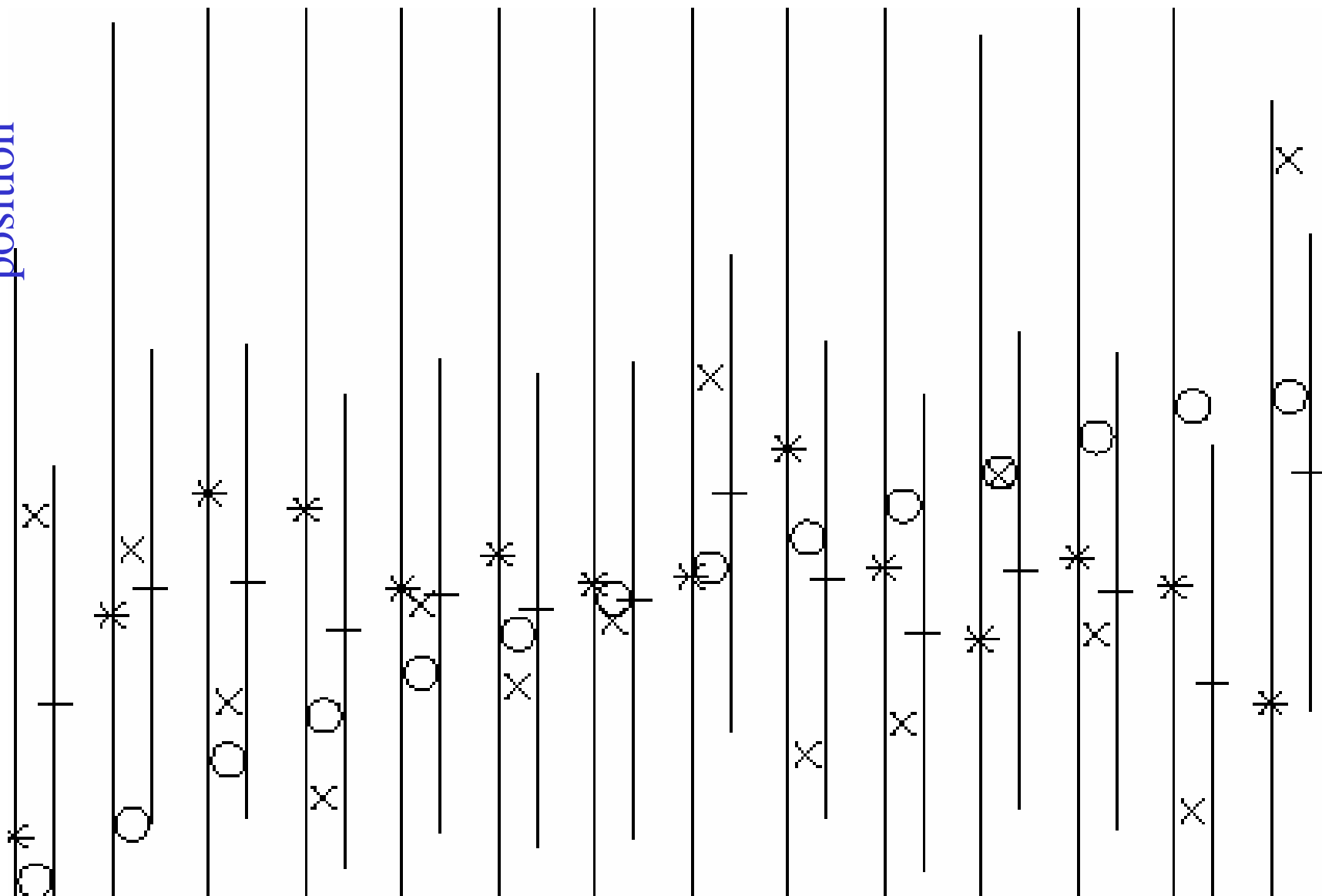
The \*-s give  $\bar{x}_i^-$ , +-s give  $\bar{x}_i^+$ , vertical bars are 3 standard deviation bars



The o-s give state, x-s measurement.

The \*-s give  $\bar{x}_i^-$ , +-s give  $\bar{x}_i^+$ , vertical bars are 3 standard deviation bars

position



time

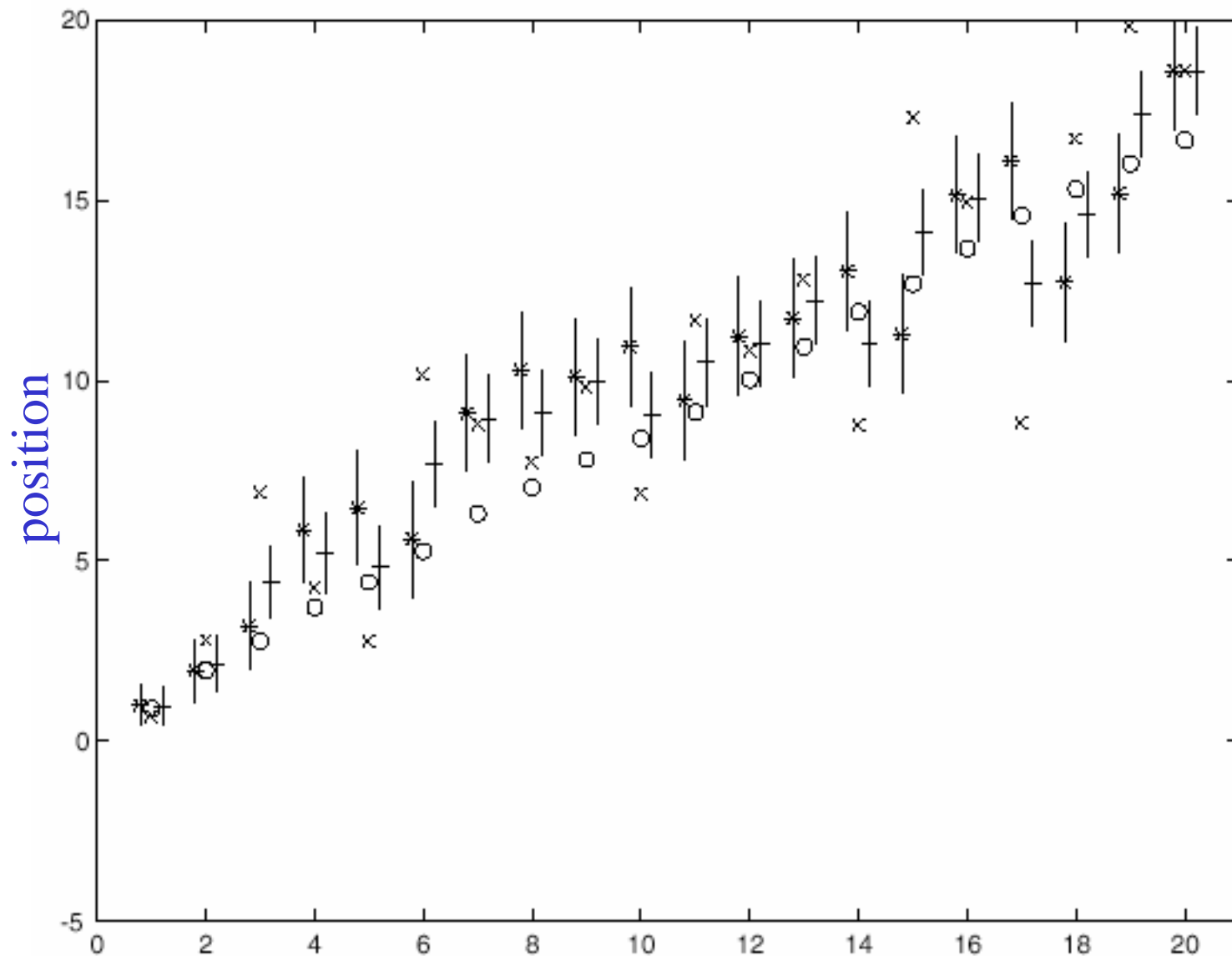
The o-s give state, x-s measurement.

The \*-s give  $\bar{x}_i^-$ , +-s give  $\bar{x}_i^+$ , vertical bars are 3 standard deviation bars

# 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

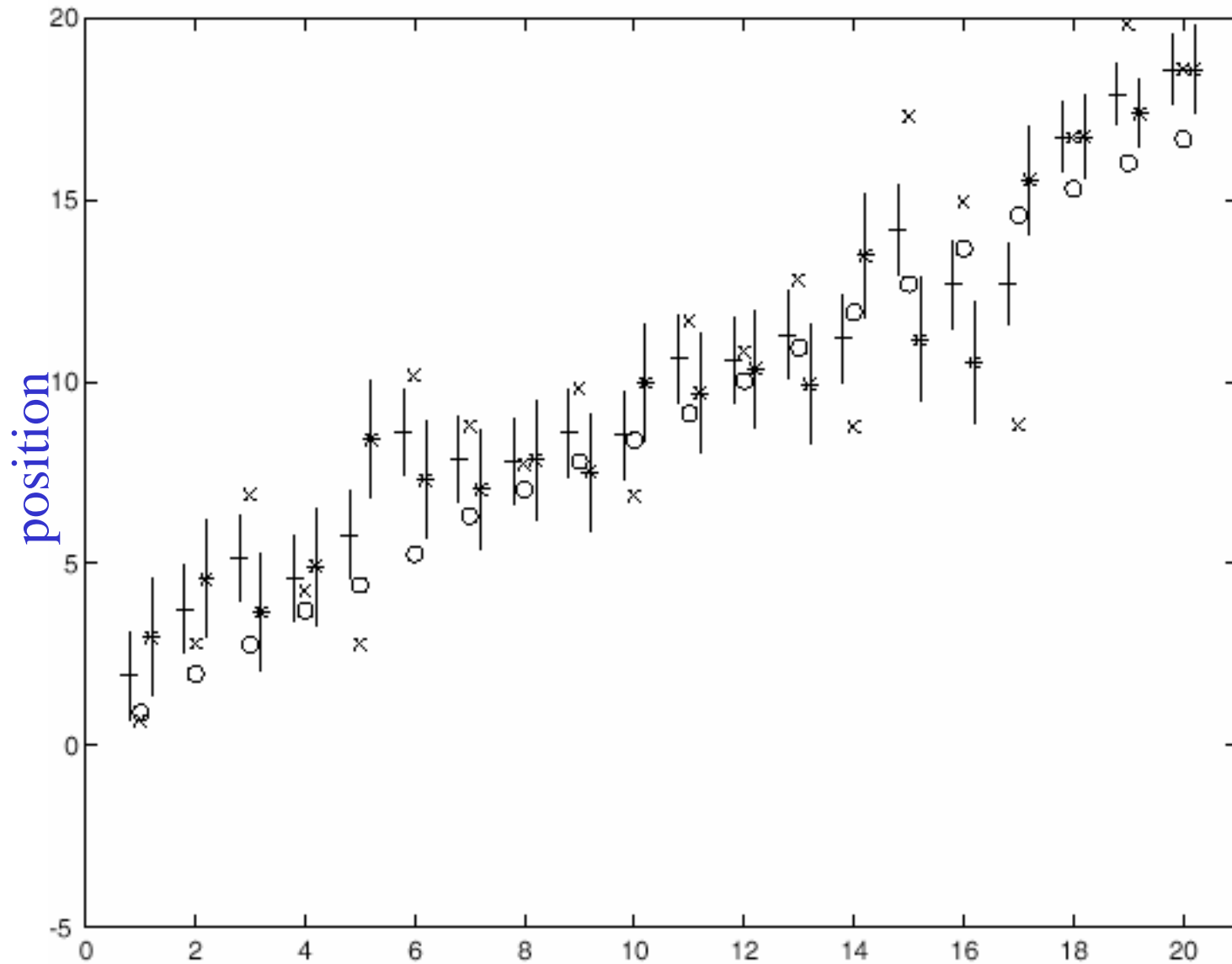
# Forward estimates.



The o-s give state, x-s measurement.

The \*-s give  $\bar{x}_i^-$ , +-s give  $\bar{x}_i^+$ , vertical bars are 3 standard deviation bars

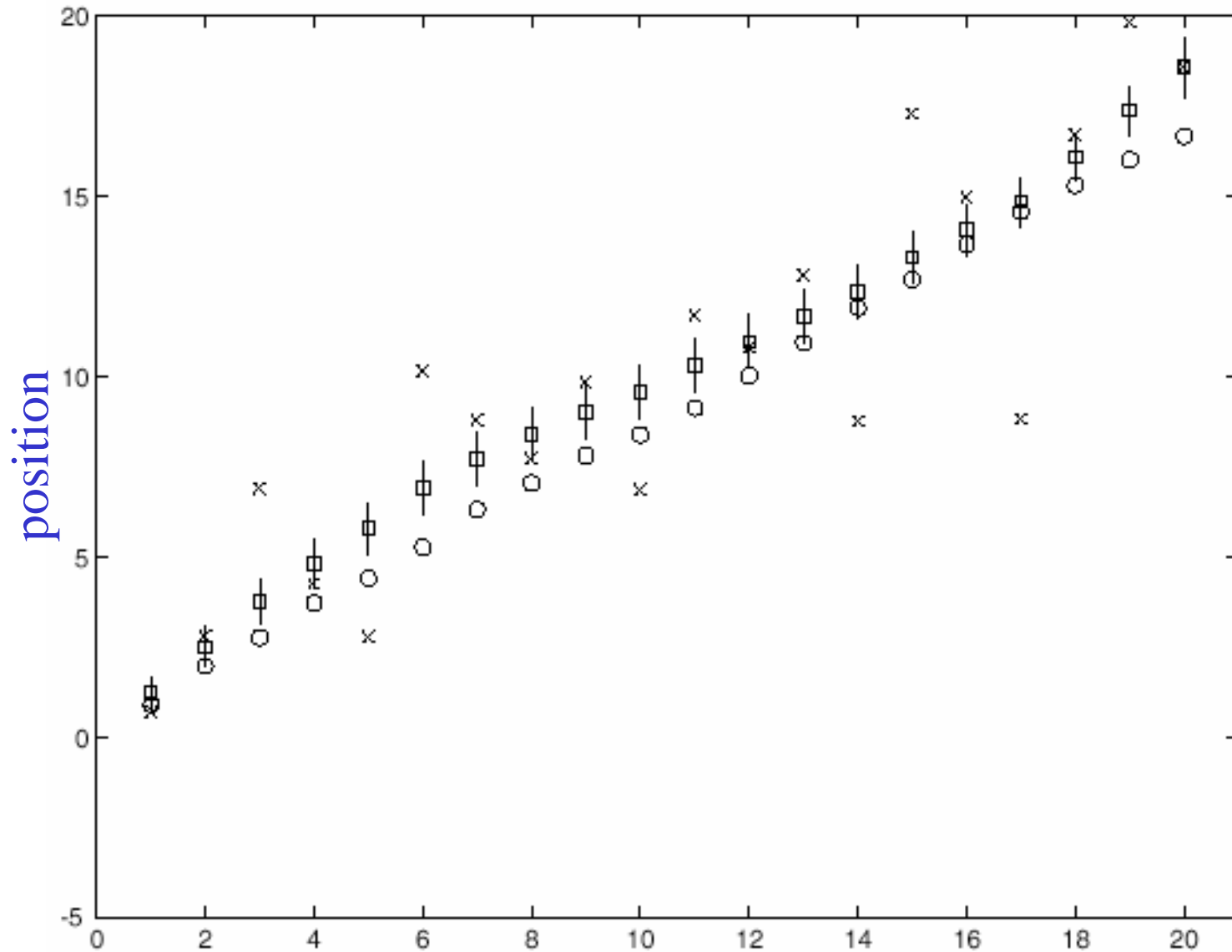
# Backward estimates.



The o-s give state, x-s measurement. **time**

The \*-s give  $\bar{x}_i^-$ , +-s give  $\bar{x}_i^+$ , vertical bars are 3 standard deviation bars

## Combined forward-backward estimates.



The o-s give state, x-s measurement. time

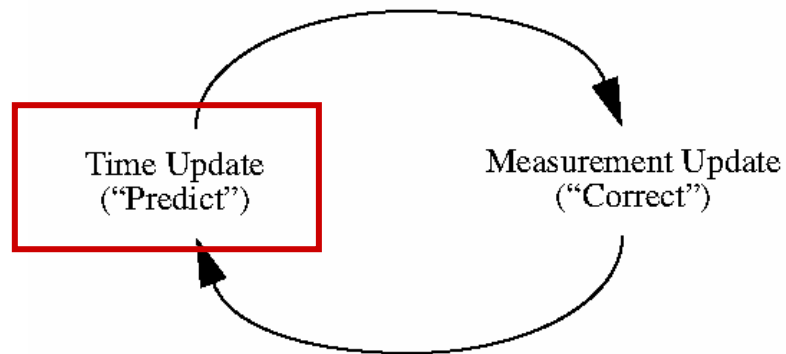
The \*-s give  $\bar{x}_i^-$ , +-s give  $\bar{x}_i^+$ , vertical bars are 3 standard deviation bars

# n-D

Generalization to n-D is straightforward but more complex.

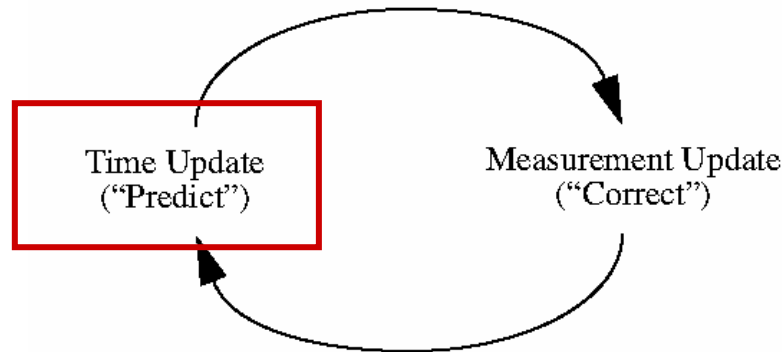
# n-D

Generalization to n-D is straightforward but more complex.



# n-D Prediction

Generalization to n-D is straightforward but more complex.



Prediction:

- Multiply estimate at prior time with forward model:

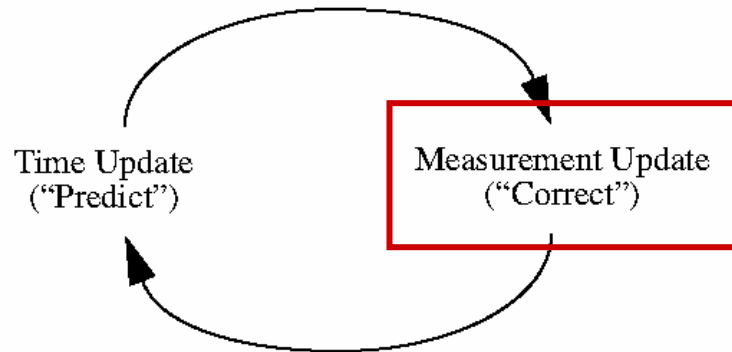
$$\bar{\mathbf{x}}_i^- = \mathcal{D}_i \bar{\mathbf{x}}_{i-1}^+$$

- Propagate covariance through model and add new noise:

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

# n-D Correction

Generalization to n-D is straightforward but more complex.



Correction:

- Update *a priori* estimate with measurement to form *a posteriori*

# n-D correction

Find linear filter on innovations

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

which minimizes *a posteriori* error covariance:

$$E \left[ \left( x - \bar{x}^+ \right)^T \left( x - \bar{x}^+ \right) \right]$$

$\mathbf{K}$  is the *Kalman Gain* matrix. A solution is

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

# Kalman Gain Matrix

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

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

As measurement becomes more reliable,  $\mathcal{K}$  weights residual more heavily,

$$\lim_{\Sigma_m \rightarrow 0} \mathcal{K}_i = \mathcal{M}^{-1}$$

As prior covariance approaches 0, measurements are ignored:

$$\lim_{\Sigma_i^- \rightarrow 0} \mathcal{K}_i = 0$$

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  $\Sigma_0^-$  are known

Update Equations: Prediction

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

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

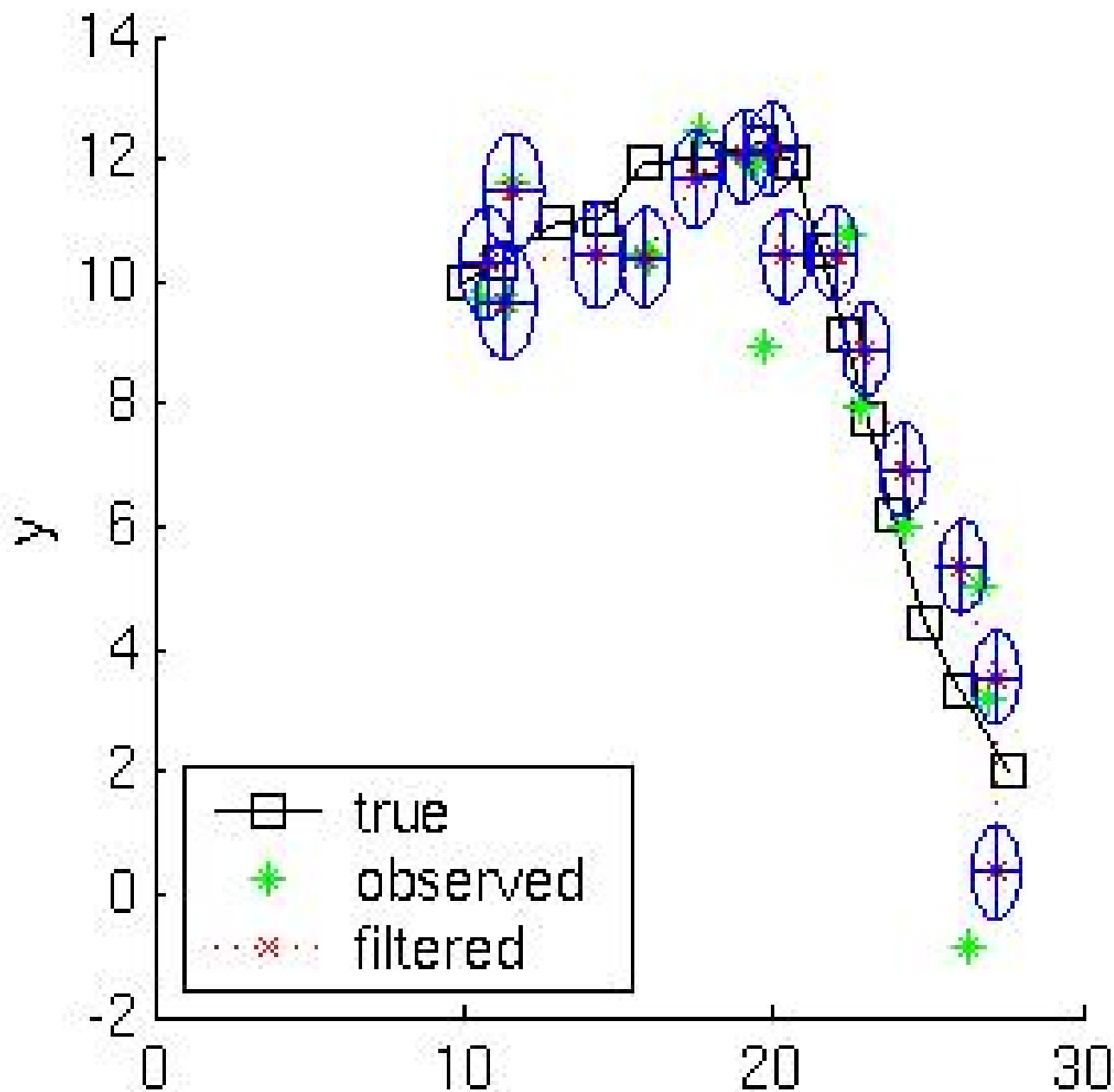
Update Equations: Correction

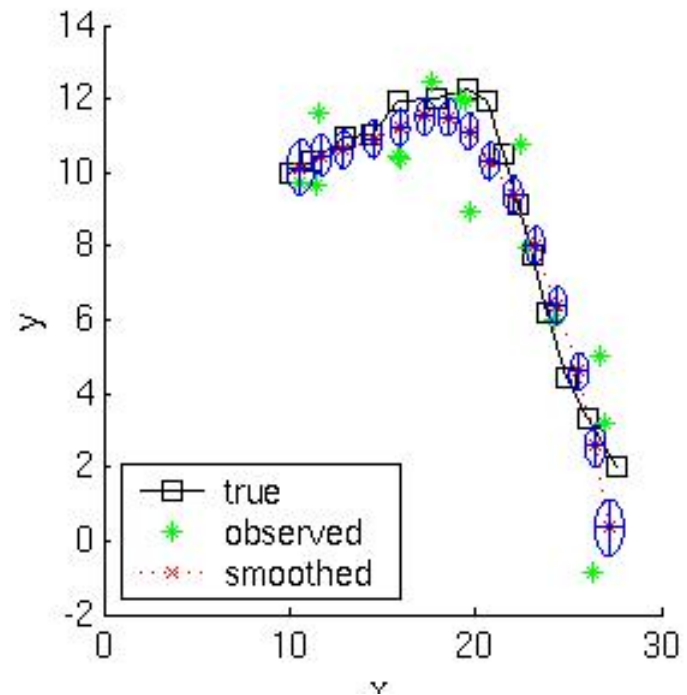
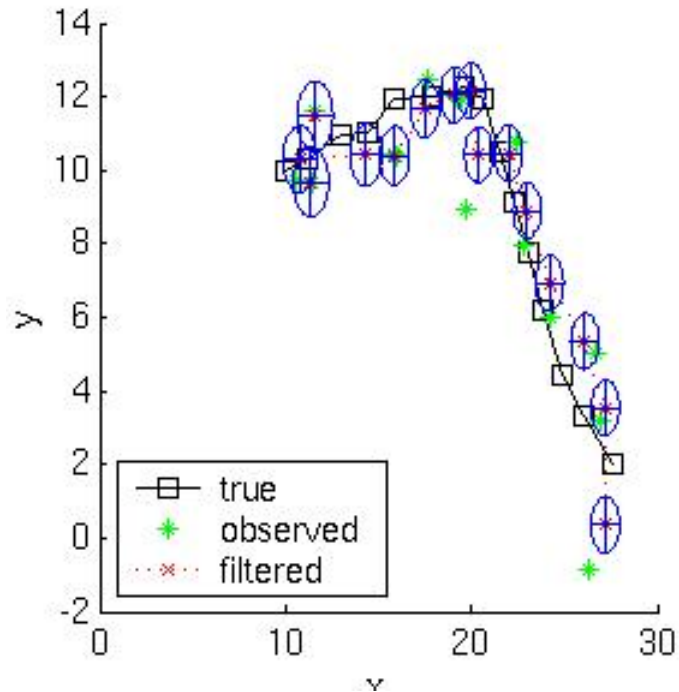
$$\mathcal{K}_i = \Sigma_i^- \mathcal{M}_i^T [\mathcal{M}_i \Sigma_i^- \mathcal{M}_i^T + \Sigma_{m_i}]^{-1}$$

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

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

## 2-D constant velocity example from Kevin Murphy's Matlab toolbox





## 2-D constant velocity example from Kevin Murphy's Matlab toolbox

- MSE of filtered estimate is 4.9; of smoothed estimate. 3.2.
- Not only is the smoothed estimate better, but we know that it is better, as illustrated by the smaller uncertainty ellipses
- Note how the smoothed ellipses are larger at the ends, because these points have seen less data.
- Also, note how rapidly the filtered ellipses reach their steady-state (“Ricatti”) values.

# Data Association

In real world  $y_i$  have clutter as well as data...

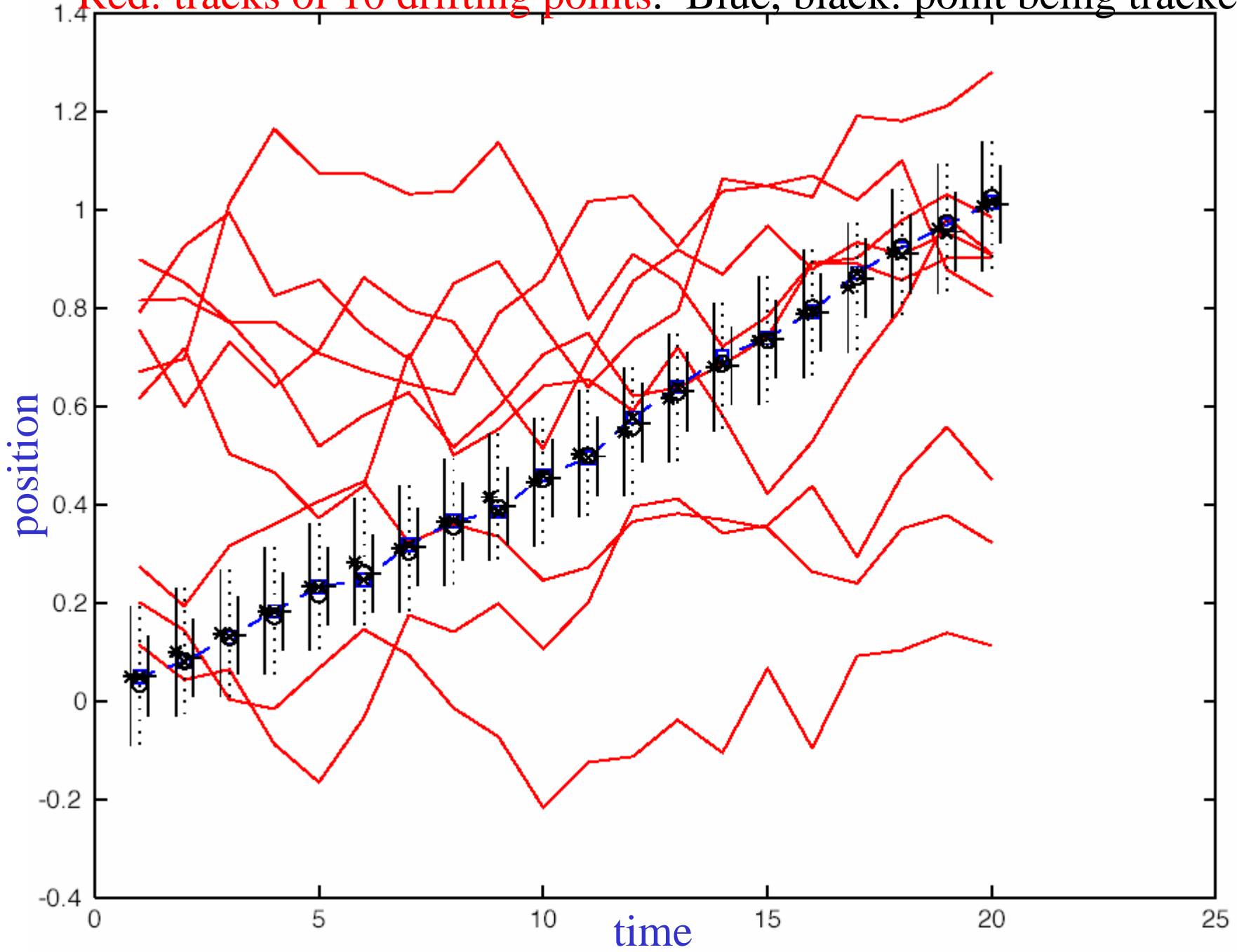
E.g., match radar returns to set of aircraft trajectories.

# Data Association

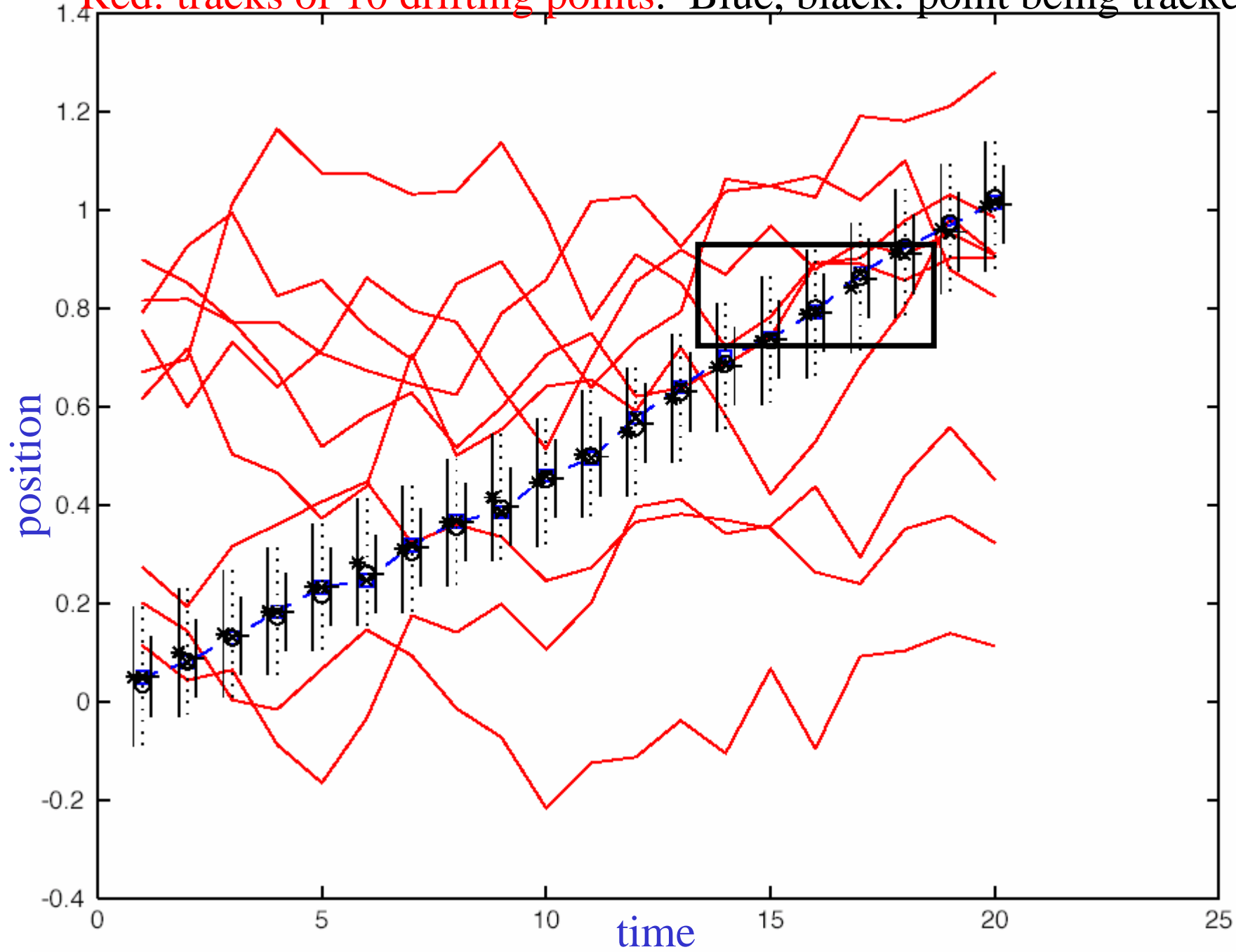
## Approaches:

- Nearest neighbours
  - choose the measurement with highest probability given predicted state
  - popular, but can lead to catastrophe
- Probabilistic Data Association
  - combine measurements, weighting by probability given predicted state
  - gate using predicted state

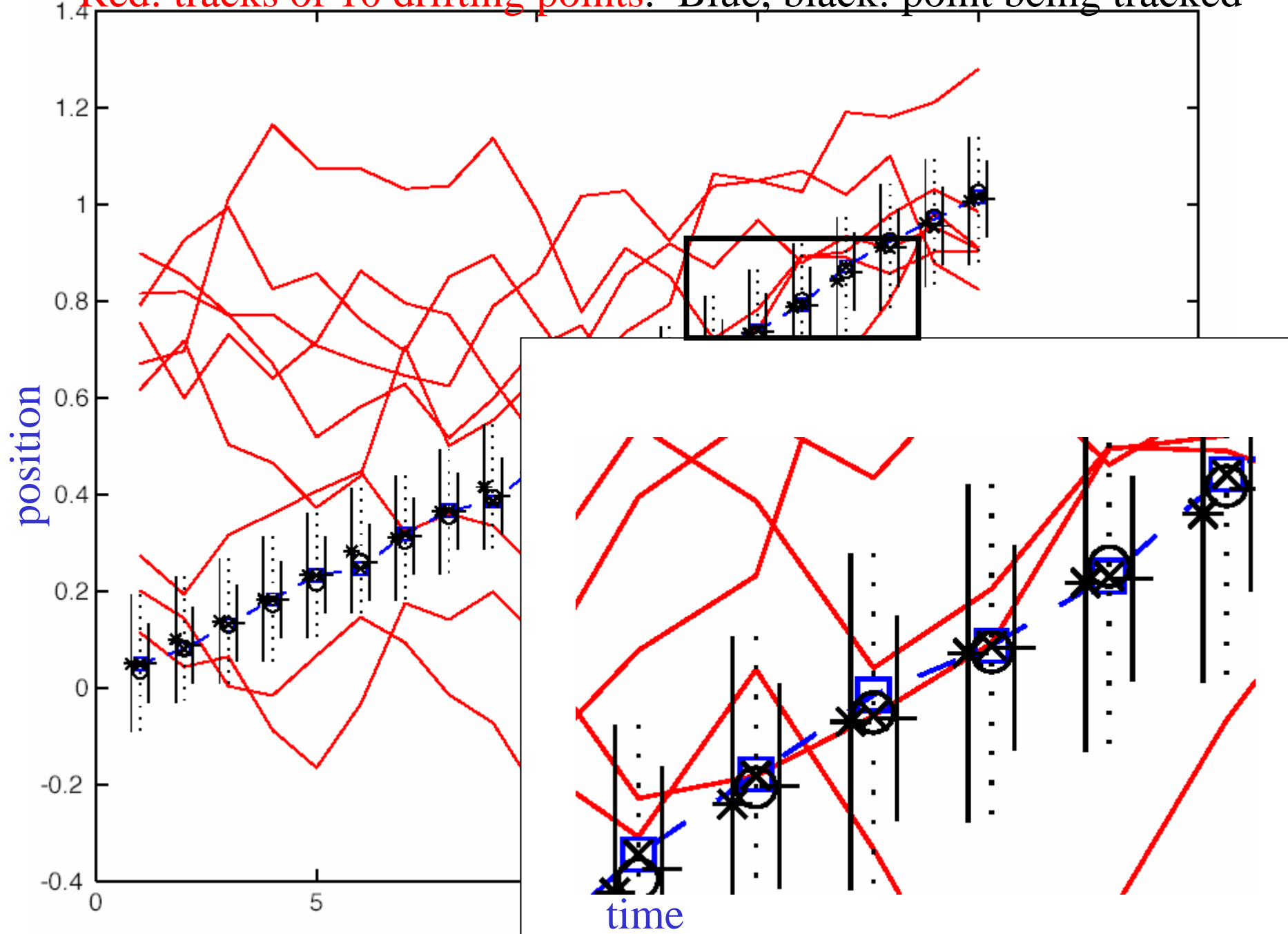
Red: tracks of 10 drifting points. Blue, black: point being tracked



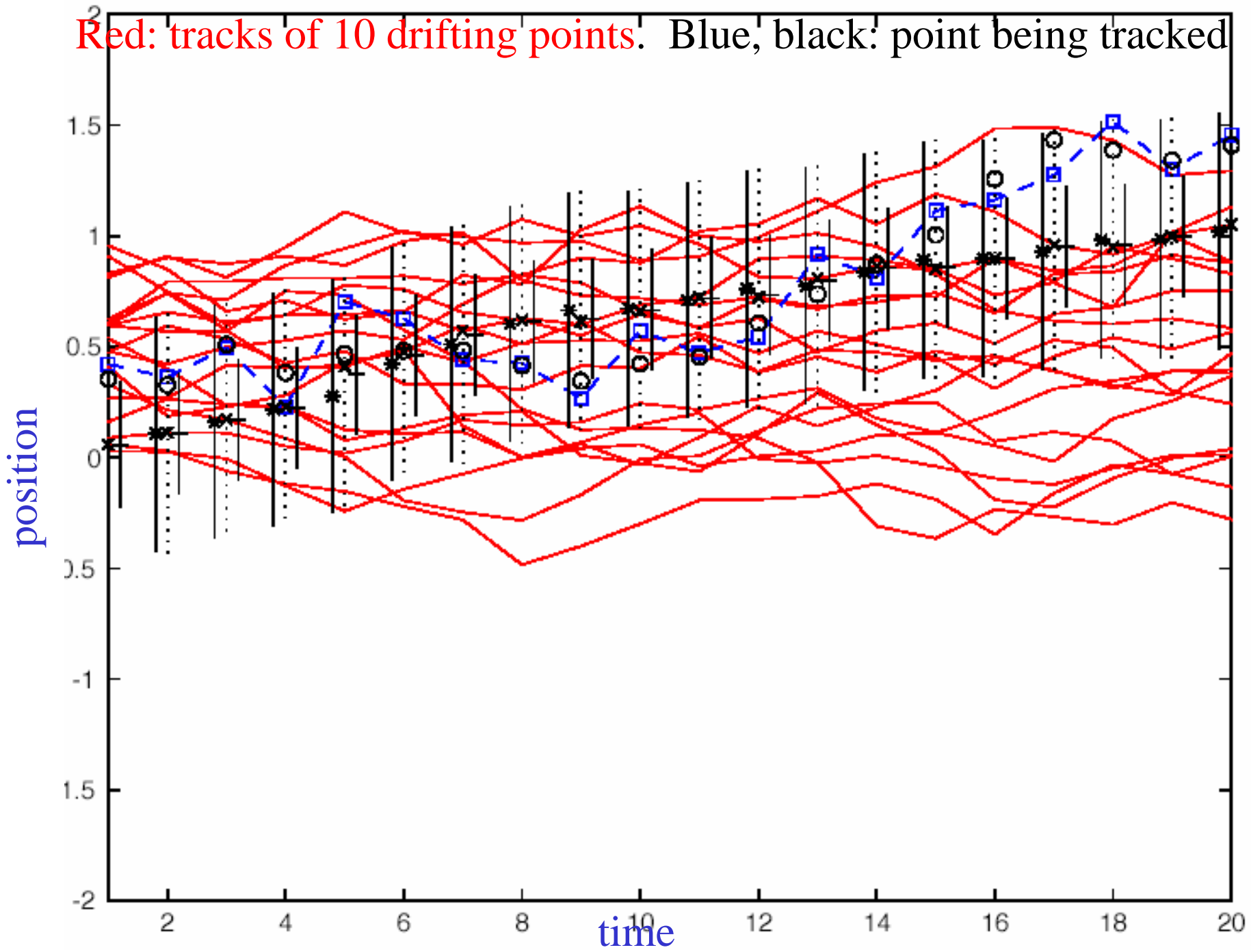
Red: tracks of 10 drifting points. Blue, black: point being tracked



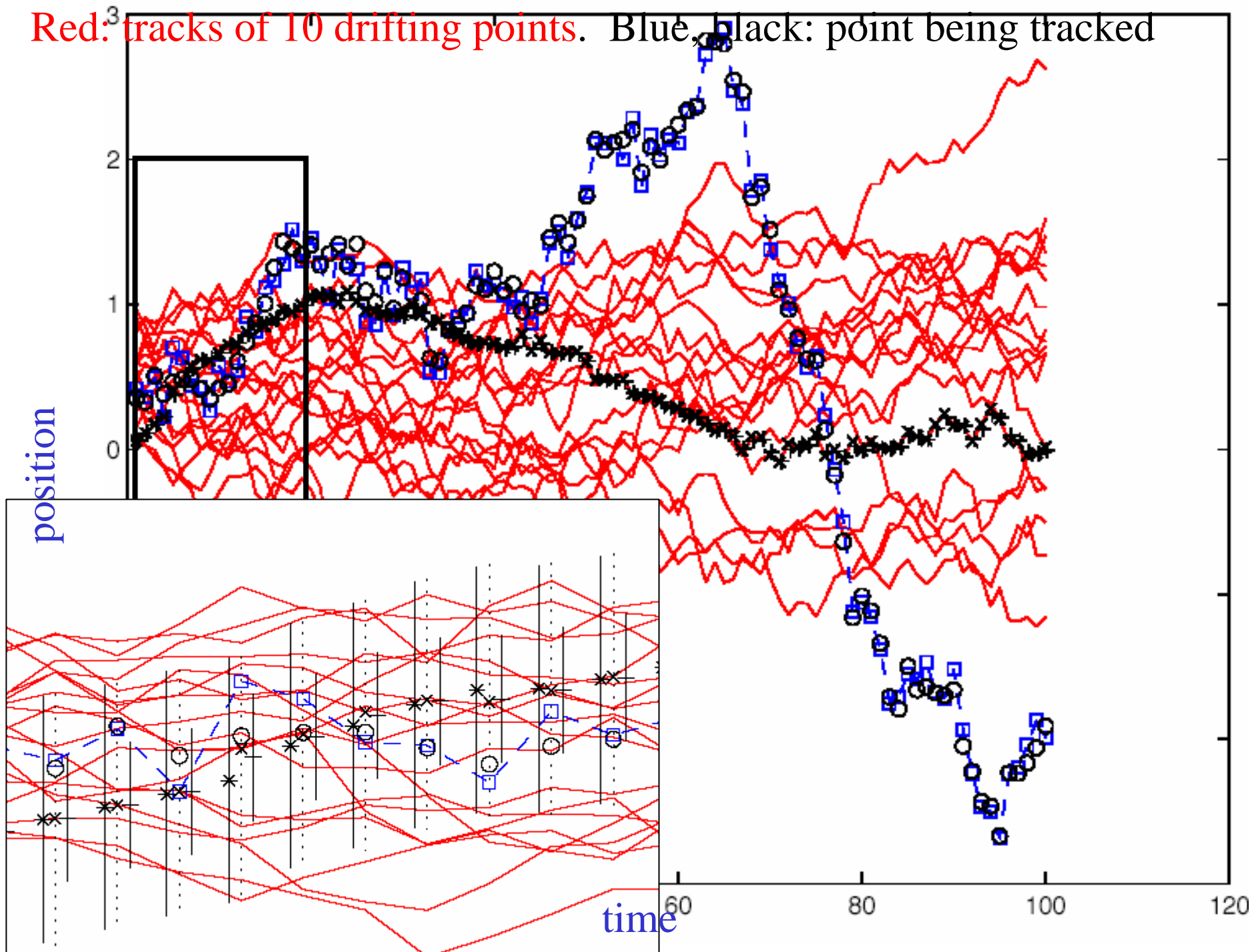
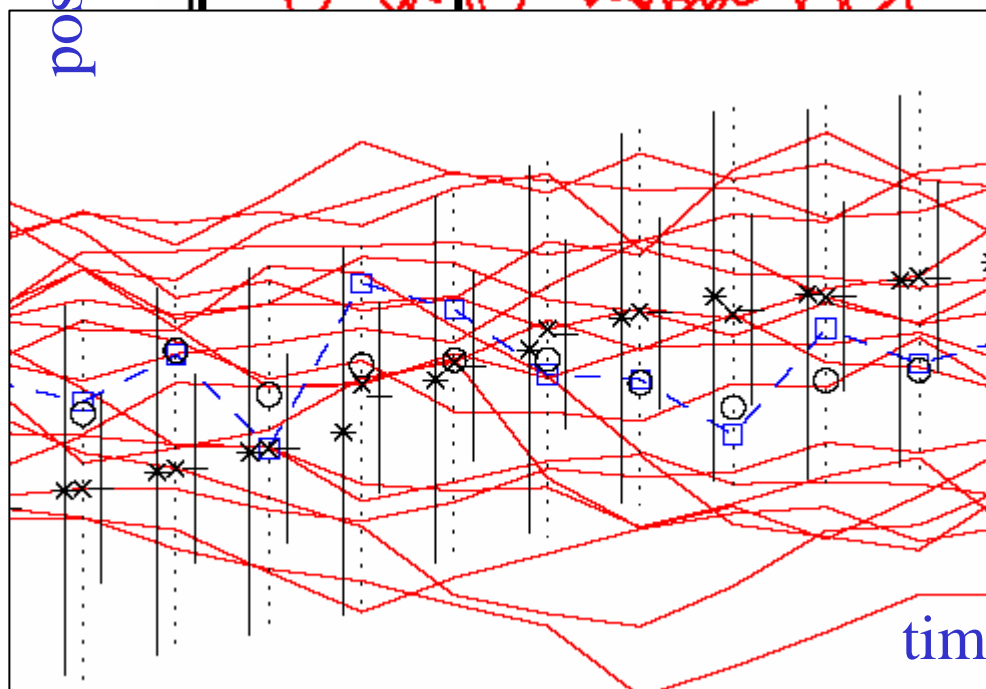
Red: tracks of 10 drifting points. Blue, black: point being tracked



Red: tracks of 10 drifting points. Blue, black: point being tracked



Red: tracks of 10 drifting points. Blue, black: point being tracked



# Abrupt changes

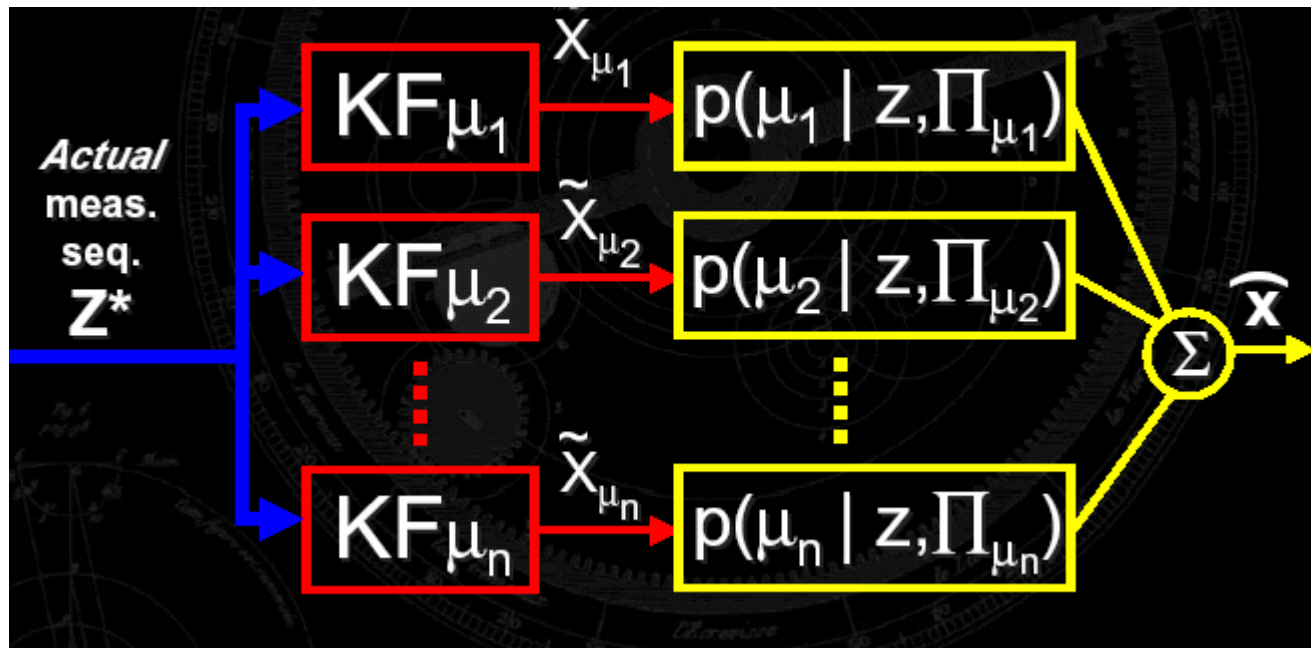
What if environment is sometimes unpredictable?

Do people move with constant velocity?

Test several models of assumed dynamics, use the best.

# Multiple model filters

Test several models of assumed dynamics



[figure from Welsh and Bishop 2001]

# Resources

- Kalman filter homepage

<http://www.cs.unc.edu/~welch/kalman/>

- Kevin Murphy's Matlab toolbox:

<http://www.ai.mit.edu/~murphyk/Software/Kalman/kalman.html>