Feature Selection & Dynamic Tracking F&P Textbook New: Ch 11, Old: Ch 17

Guido Gerig

CS 6320, Spring 2013

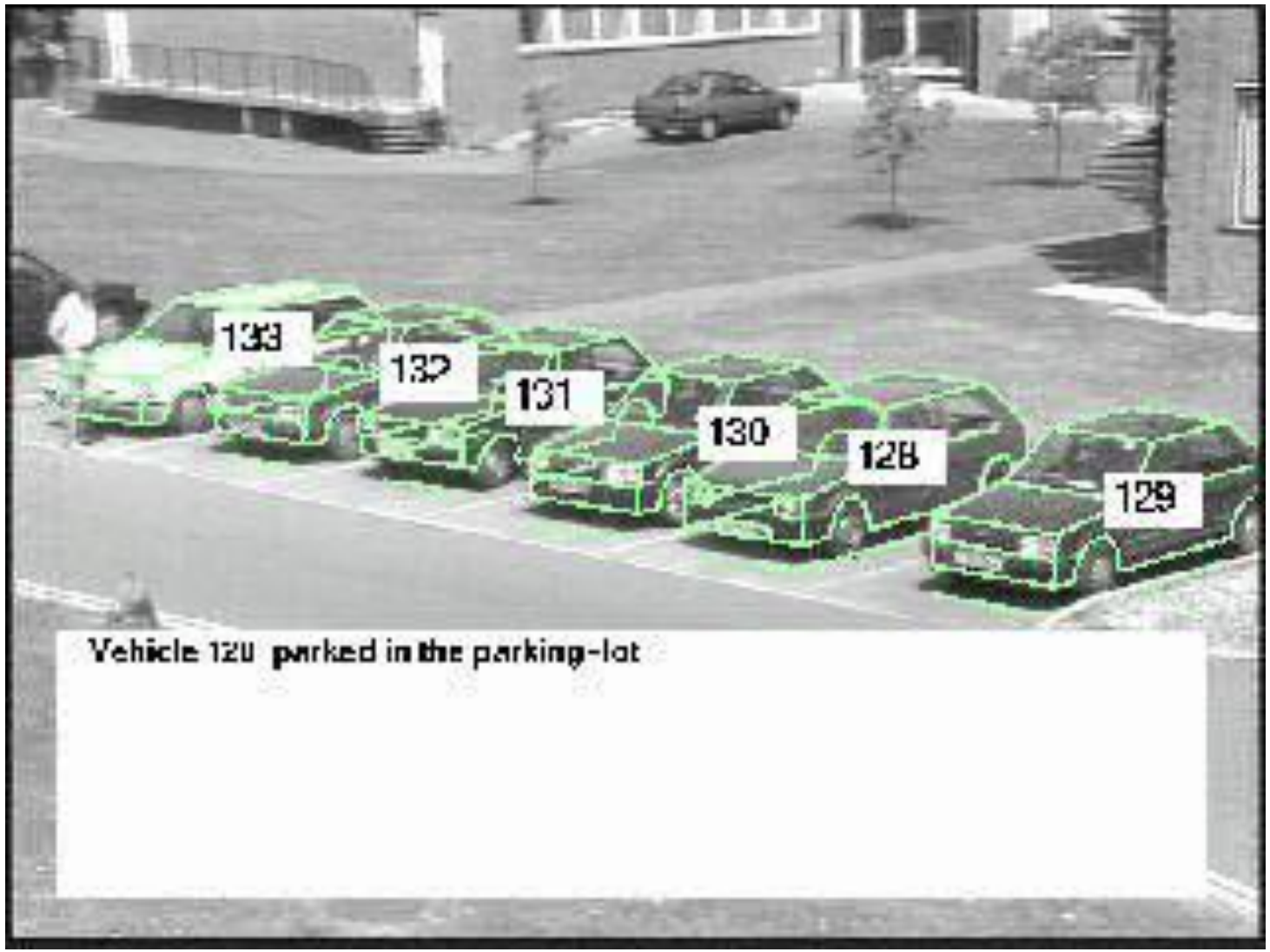
Credits: Material Greg Welch & Gary Bishop, UNC Chapel Hill, some slides modified from J.M. Frahm/ M. Pollefeys, and R. Klette Course Materials

Material

- Feature selection: SIFT Features
- Tracking: Kalman Filter:
  - F&P Chapter 17
  - Greg Welch and Gary Bishop, UNC: http://www.cs.unc.edu/~welch/kalman/
  - Web-site (electronic and printed references, book lists, Java demo, software etc.)
  - Course material SIGGRAPH: http://www.cs.unc.edu/~tracker/ref/s2001/kalman/index.html

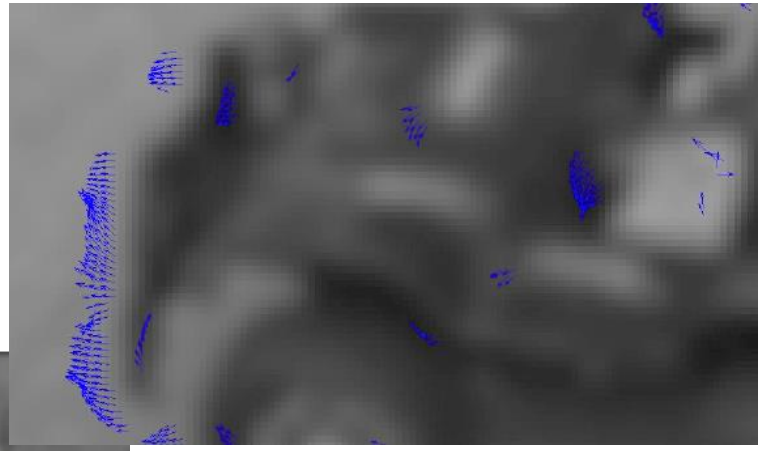
Tracking – Rigid Objects

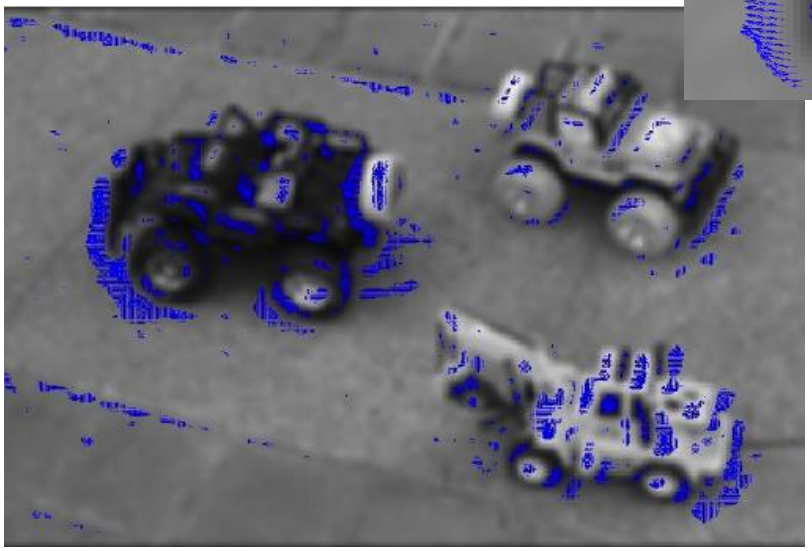
Tracking – Rigid Objects



Tracking – Rigid Objects

Tracking – Rigid Objects





Feature Tracking

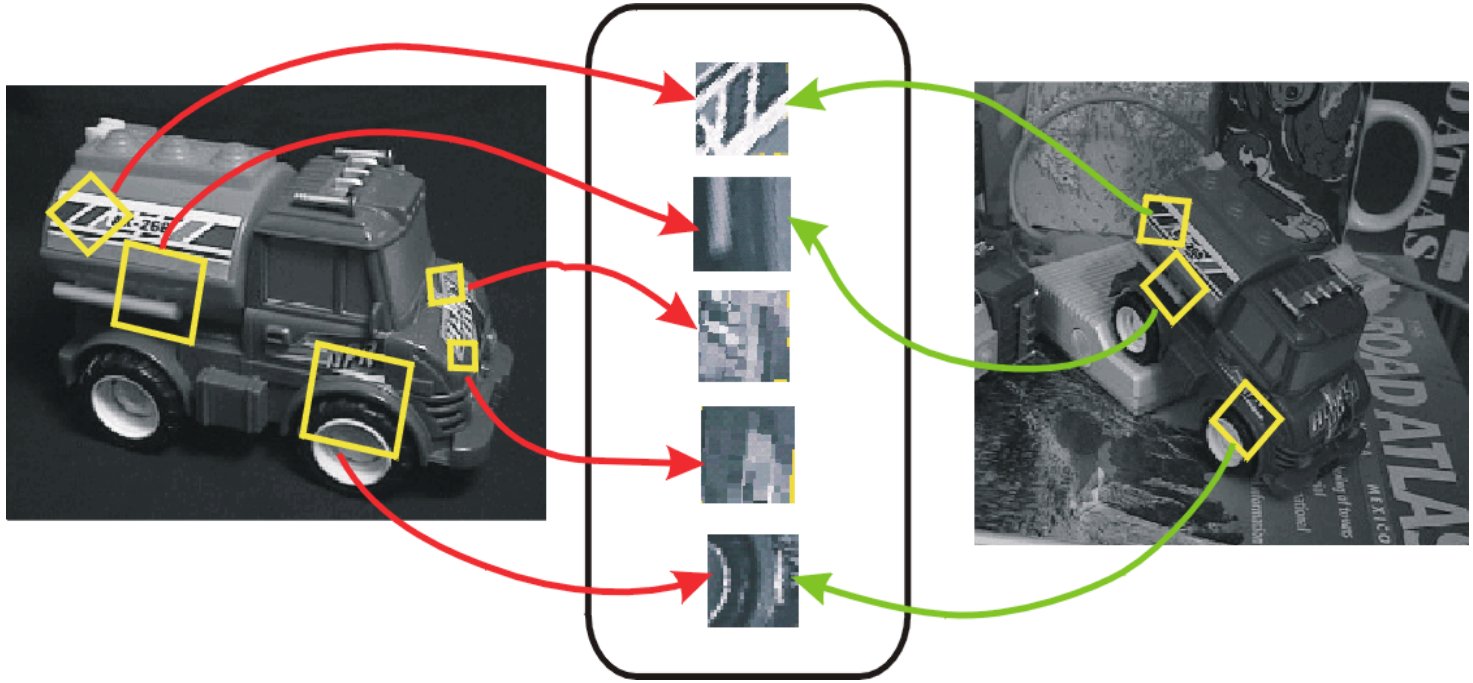
- Tracking of "good" features & efficient search for subsequent positions.
- What are good features?
- Required properties:
  - Well-defined
    - (i.e. neigboring points should all be different)
  - Stable across views
    - (i.e. same 3D point should be extracted as feature for neighboring viewpoints)

Lowe's SIFT features

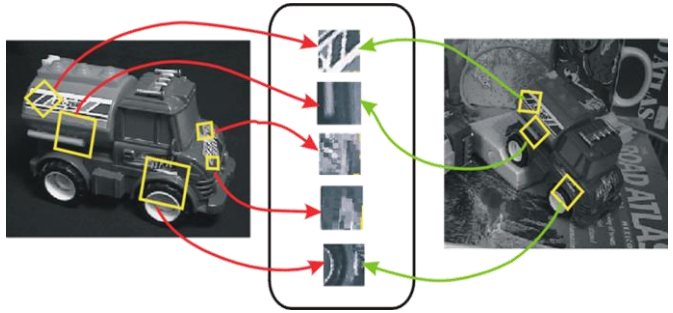
(Lowe, ICCV99)

SIFT: Scale Invariant Feature Transform

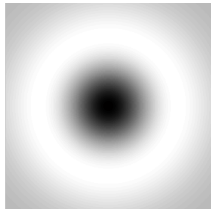
Recover features with change of position, orientation and scale

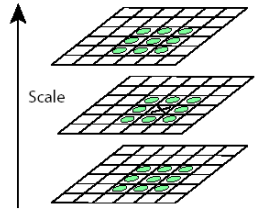


SIFT features

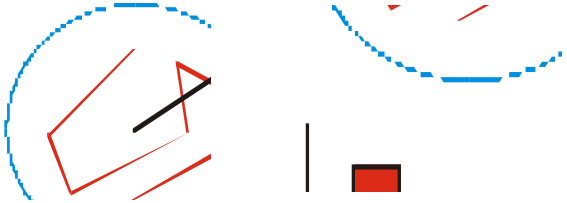


- Scale-space DoG maxima

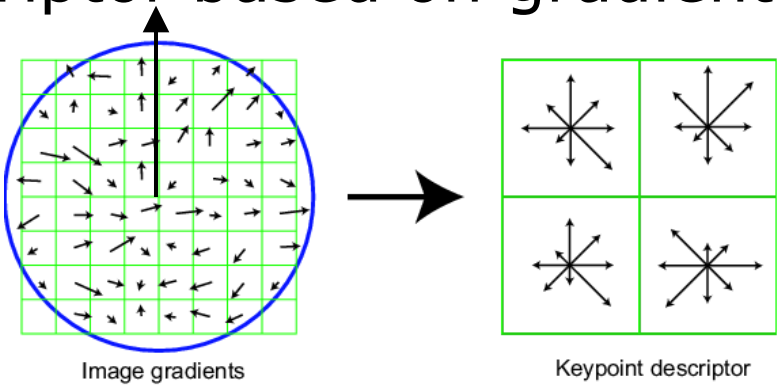




- Verify minimum contrast and “cornerness”
- Orientation from dominant gradient



- Descriptor based on gradient distributions



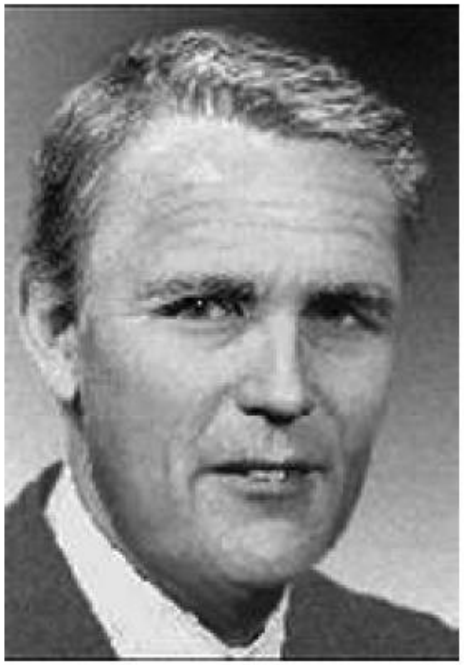
Dynamic Feature Tracking

Tracking is the problem of generating an inference about the motion of an object given a sequence of images.

The **key technical difficulty** is maintaining an accurate representation of the posterior on object position given measurements, and doing so efficiently.

The Kalman Filter

"The *Kalman filter* is a set of mathematical equations that provides an efficient computational (recursive) means to estimate the state of a process, in a way that minimizes the mean of the squared error. The filter is very powerful in several aspects: it supports estimations of past, present, and even future states, and it can do so even when the precise nature of the modeled system is unknown." (G. Welch and G. Bishop, 2004)



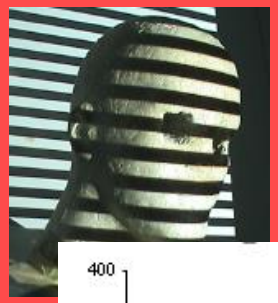
Named after Rudolf Emil Kalman (1930, Budapest/Hungary).

Kalman Filter

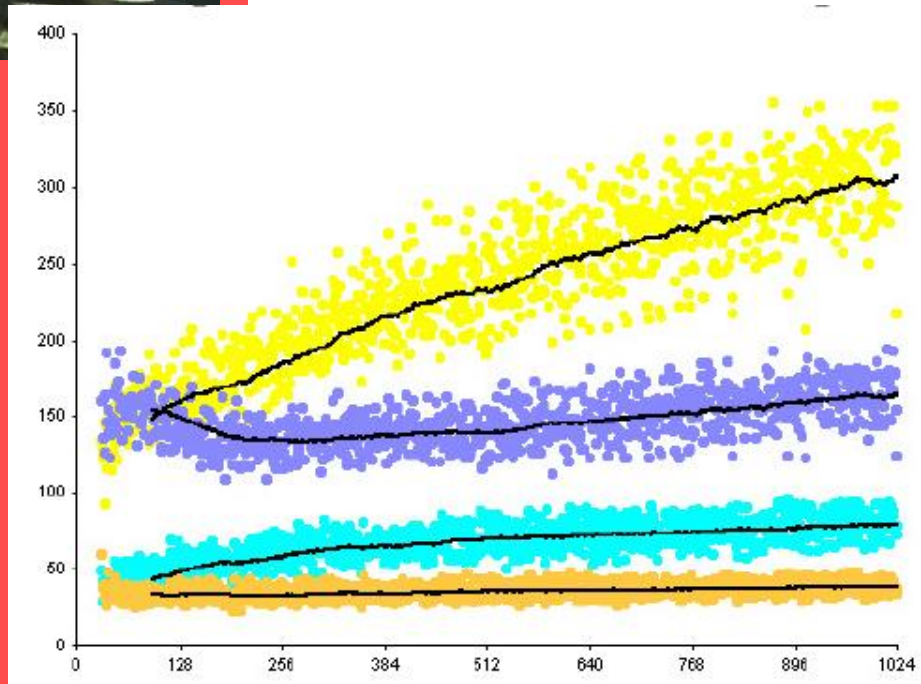
- The Kalman filter is a very powerful tool when it comes to controlling **noisy** systems.
- Apollo 8 (December 1968), the first human spaceflight from the Earth to an orbit around the moon, would certainly not have been possible without the Kalman filter (see www.ion.org/museum/item_view.cfm?cid=6&scid=5&iid=293).
- **Applications:**
  - Tracking
  - Economics
  - Navigation
  - Depth and velocity measurements
  - …..

What is it used for?

- Tracking missiles
- Tracking heads/hands/drumsticks
- Extracting lip motion from video
- Fitting Bezier patches to point data
- Lots of computer vision applications
- Economics
- Navigation



Key Concept



- Noisy process data
- Estimate average trajectories
- Smoothing: Sliding window for averaging (here size 64)

- But: If horizontal axis is time? We know the past but not the future!
- Time-dependent process: Modeling of process itself including noise estimates.

Model for tracking

- Object has internal state X_i
  - Capital indicates random variable X_i
  - Small represents particular value x_i

- Obtained measurements in frame i are Y_i
  - Value of the measurement y_i

Linear Dynamic Models

- State is linearly transformed plus Gaussian noise

$$x_i \sim N\left(D_i x_{i-1}, \Sigma_{d_i}\right)$$

- Relevant measures are linearly obtained from state plus Gaussian noise

$$y_i \sim N\left(M_i x_i, \Sigma_{m_i}\right)$$

- Sufficient to maintain mean and standard deviation

General Steps of Tracking

1. **Prediction:** What is the next state of the object given past measurements

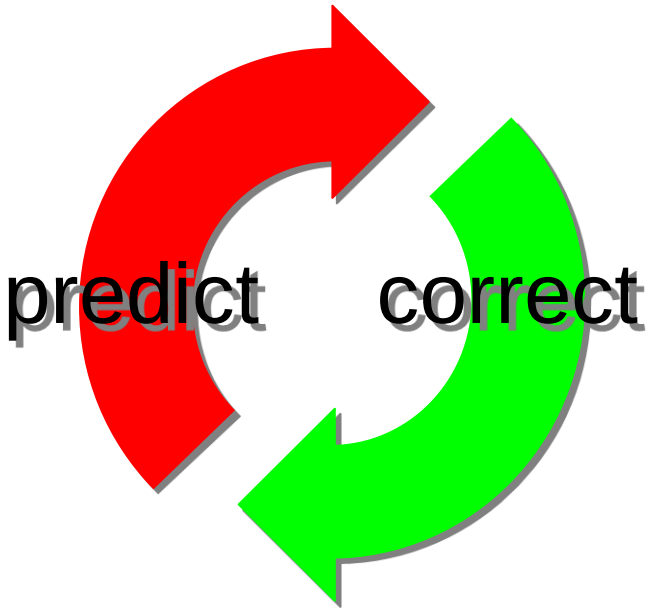
$$P\left(X_i \middle| Y_0 = y_0, \ldots, Y_{i-1} = y_{i-1}\right)$$

2. **Data association:** Which measures are relevant for the state?

3. **Correction:** Compute representation of the state from prediction and measurements.

$$P\left(X_i \middle| Y_0 = y_0, \ldots, Y_{i-1} = y_{i-1}, Y_i = y_i\right)$$

Concept Kalman Filtering



Independence Assumptions

- Only immediate past matters

$$P\left(X_i \middle| X_1,\ldots,X_{i-1}\right) = P\left(X_i \middle| X_{i-1}\right)$$

- Measurements depend only on current state

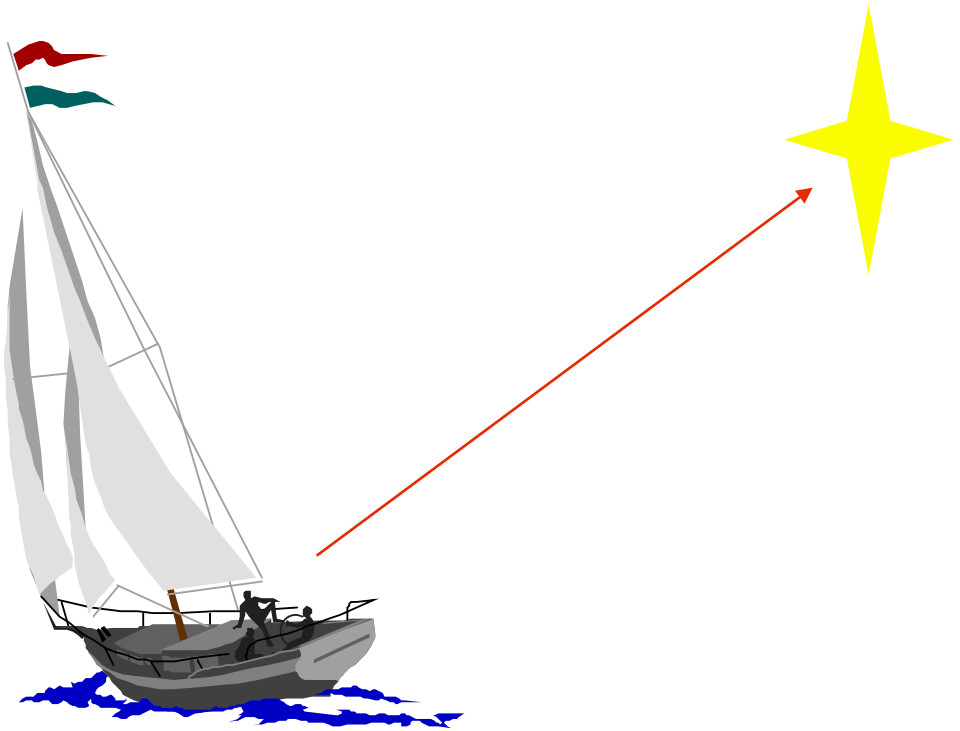
$$P\left(Y_i, Y_j,\ldots,Y_k \middle| X_i\right) = P\left(Y_i \middle| X_i\right) P\left(Y_j,\ldots,Y_k \middle| X_i\right)$$

$\Rightarrow$ Important simplifications

Fortunately it doesn't limit to much!

Spirit of Kalman Filtering: A *really* simple example

We are on a boat at night and lost our position



We know:

- star position

Fixed Position

p is position of boat, v is velocity of boat

$$p_i = p_{i-1}$$

state is $X_i = \begin{bmatrix} p_i \end{bmatrix}$

$$X_i = D_i X_{i-1} \qquad D_i = \begin{bmatrix} I \end{bmatrix}$$

We only measure position so

$$M_i = \begin{bmatrix} I \end{bmatrix}, \quad Y_i = M_i X_i = X_i$$

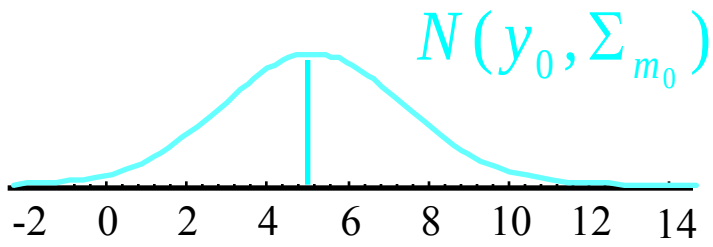
Observer 1 makes a measurement

y_0 , Σ_{m_0}

$x_0 = y_0$

$\Sigma_0 = \Sigma_{m_0}$

Conditional Density Function



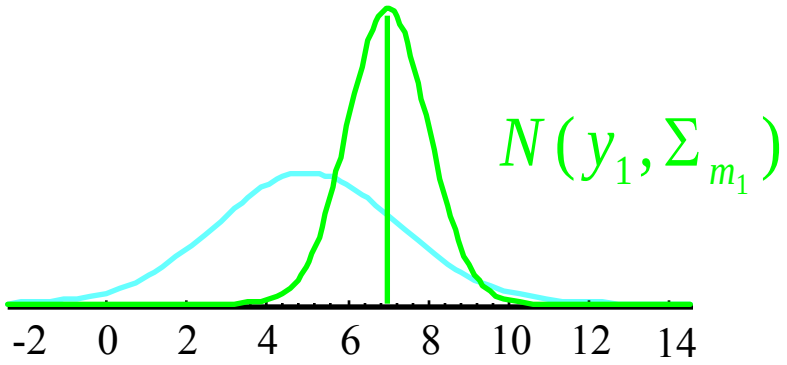
Then: Observer 2 makes a measurement

y_1 , Σ_{m_1}

$x_1 = \ldots ?$

$\Sigma_1 = \ldots ?$

Conditional Density Function



How does second measurement affect estimate of first measurement?

Combine measurements & variances: Kalman

$$x_2 = x_1 + K_2(y_2 - x_1)$$

$$K_2 = \frac{\sigma_1^2}{\sigma_1^2 + \sigma_{y_2}^2}$$

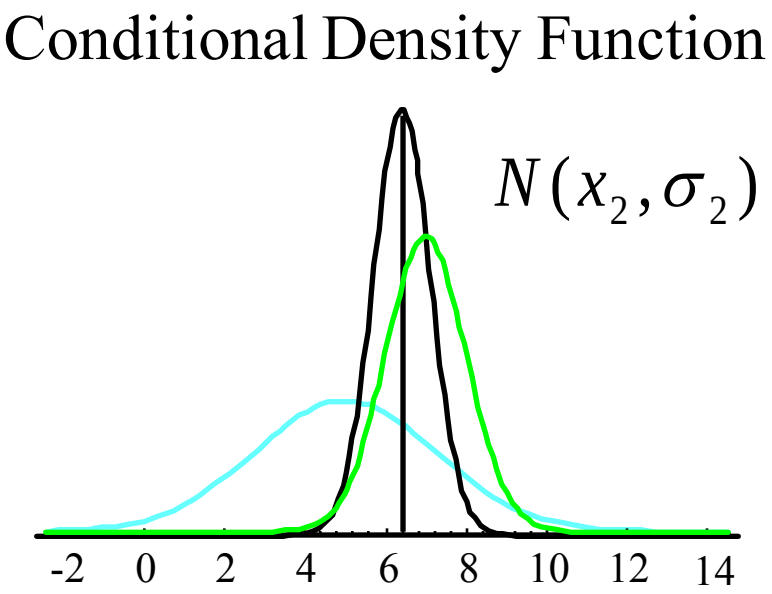
$$\frac{1}{\sigma_2^2} = \frac{1}{\sigma_1^2} + \frac{1}{\sigma_{y_2}^2}$$

Combine Variances (statistics)

Combine measurements & variances: Kalman

$$x = x_2$$

$$\sigma^2 = \sigma_2^2$$

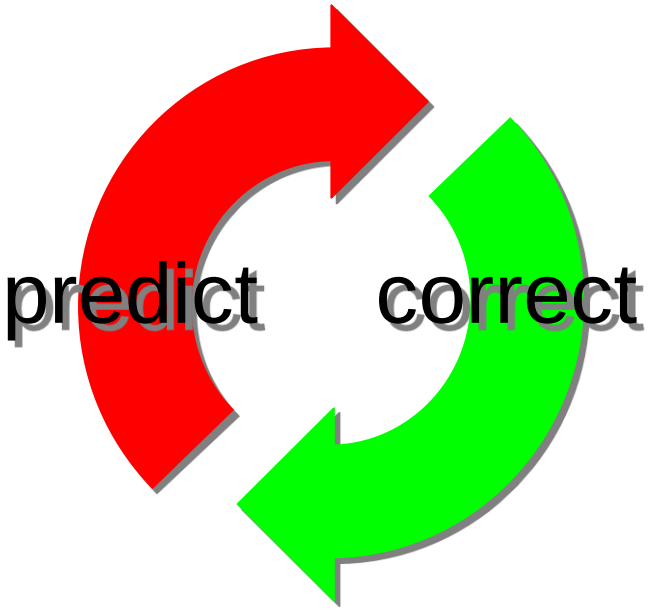


Original estimates updated (corrected) in the presence of a new measurement.

Predict → Correct

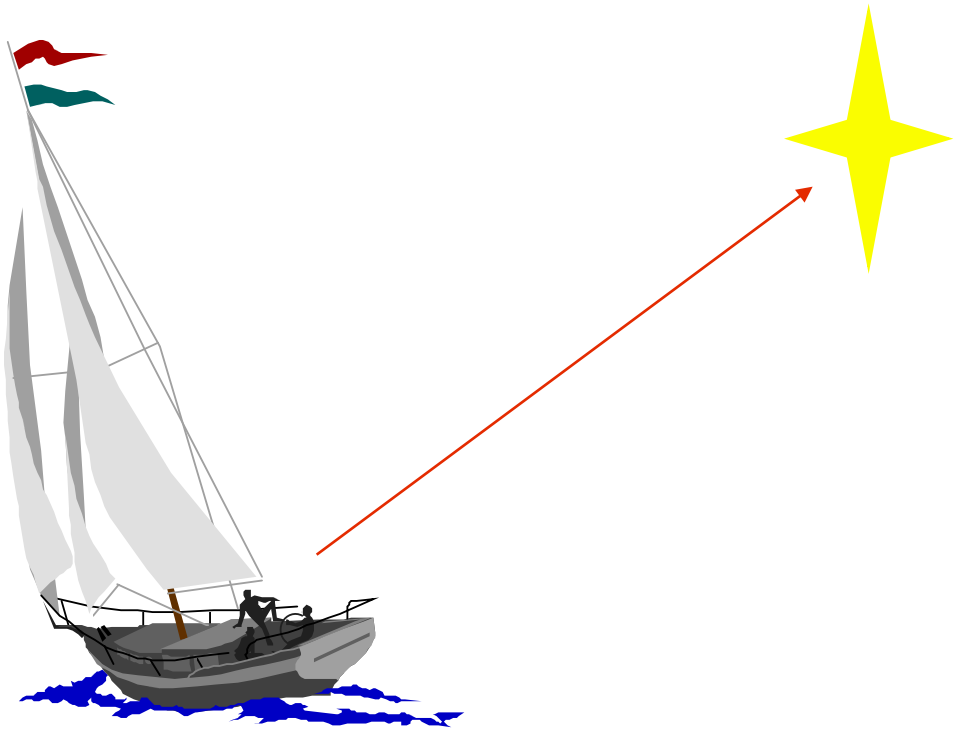
KF operates by

1. Predicting the new state and its uncertainty
2. Correcting with the new measurement



A *really* simple example

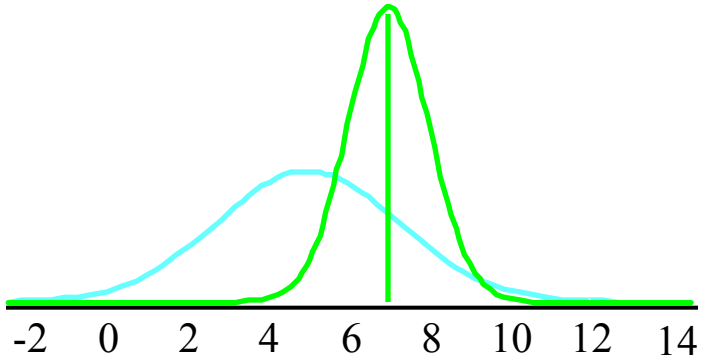
We are on a boat at night and lost our position



We know:

- move with constant velocity
- star position

But suppose we're moving



- Not *all* the difference is error. Some may be motion
- KF can include a **motion model**
- Estimate **velocity** and **position**

Process Model

- Describes how the *state* changes over time
- The *state* for the first example was scalar
- The *process model* was "nothing changes"
- A better model might be constant velocity motion $X = \begin{bmatrix} p & v \end{bmatrix}^t$

$$p_i = p_{i-1} + (\Delta t) v_{i-1}$$

$$v_i = v_{i-1}$$

Measurement Model

"What you see from where you are"

not

"Where you are from what you see"

Constant Velocity

p is position of boat, v is velocity of boat

$$p_i = p_{i-1} + (\Delta t) v_{i-1}$$

state is $X = [p \;\; v]^t$

$$X_i = D_i X_{i-1} \qquad D_i = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}$$

We only measure position so

$$M = [1 \;\; 0]^t, \quad \mathrm{Y_i} = [1 \;\; 0]^t [p_i \;\; v]^t = p_i$$

Multidimensional Statistics

To be seen as a generalization of the scalar-valued mean and variance to higher dimensions.

$$\mathbf{X} = \begin{bmatrix} X_1 \\ \vdots \\ X_n \end{bmatrix}$$

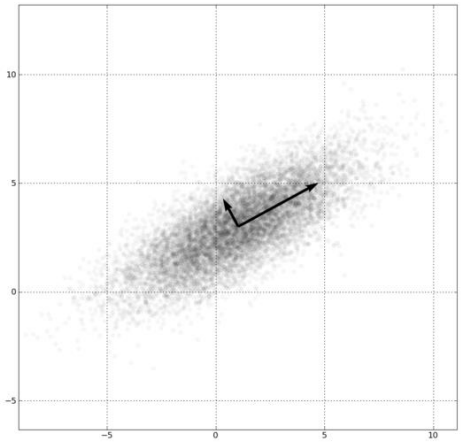
$$\mu_i = \mathrm{E}(X_i)$$

$$\Sigma_{ij} = \mathrm{cov}(X_i, X_j) = \mathrm{E}\left[(X_i - \mu_i)(X_j - \mu_j)\right]$$

$$\Sigma = \mathrm{E}\left[(\mathbf{X} - \mathrm{E}[\mathbf{X}])\,(\mathbf{X} - \mathrm{E}[\mathbf{X}])^{\mathrm{T}}\right]$$

$$\mathrm{var}(\mathbf{X}) = \mathrm{cov}(\mathbf{X}) = \mathrm{E}\left[(\mathbf{X} - \mathrm{E}[\mathbf{X}])(\mathbf{X} - \mathrm{E}[\mathbf{X}])^{\mathrm{T}}\right]$$

$$\mathrm{cov}(\mathbf{X}, \mathbf{Y}) = \mathrm{E}\left[(\mathbf{X} - \mathrm{E}[\mathbf{X}])(\mathbf{Y} - \mathrm{E}[\mathbf{Y}])^{\mathrm{T}}\right]$$



Sample points from a multivariate Gaussian distribution with a standard deviation of 3 in roughly the lower left-upper right direction and of 1 in the orthogonal direction. Because the *x* and *y* components co-vary, the variances of *x* and *y* do not fully describe the distribution. A 2×2 covariance matrix is needed; the directions of the arrows correspond to the eigenvectors of this covariance matrix and their lengths to the square roots of the eigenvalues.

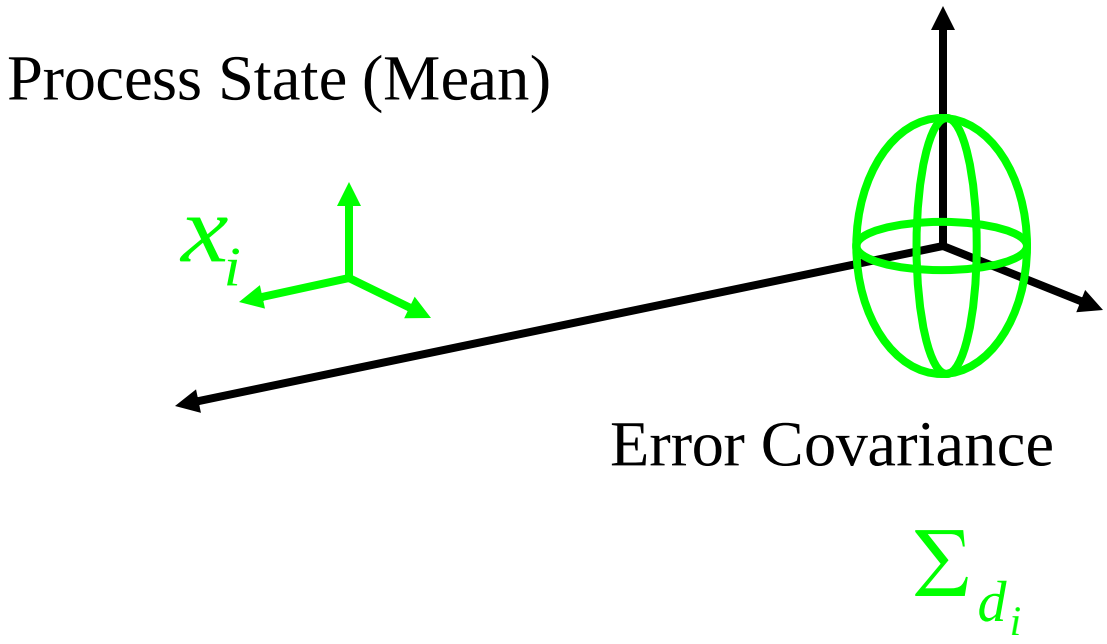
2 variables:

$$\begin{bmatrix} var(1,1) & cov(1,2) \\ cov(2,1) & var(2,2) \end{bmatrix}$$

Source: Wikipedia

State and Error Covariance

- First two moments of Gaussian process



The Process Model

Process dynamics

$$X_i = D_i X_{i-1} + w_i$$

State transition

Uncertainty over interval

$$w_i \sim N(0, \Sigma_{d_i})$$

Difficult to determine

Measurement Model

Measurement relationship to state

$$Y_i = M\ X_i + \varepsilon_i$$

Measurement matrix

Measurement uncertainty

$$\varepsilon_i \sim N\left(0, \Sigma_{m_i}\right)$$

Predict (Time Update)

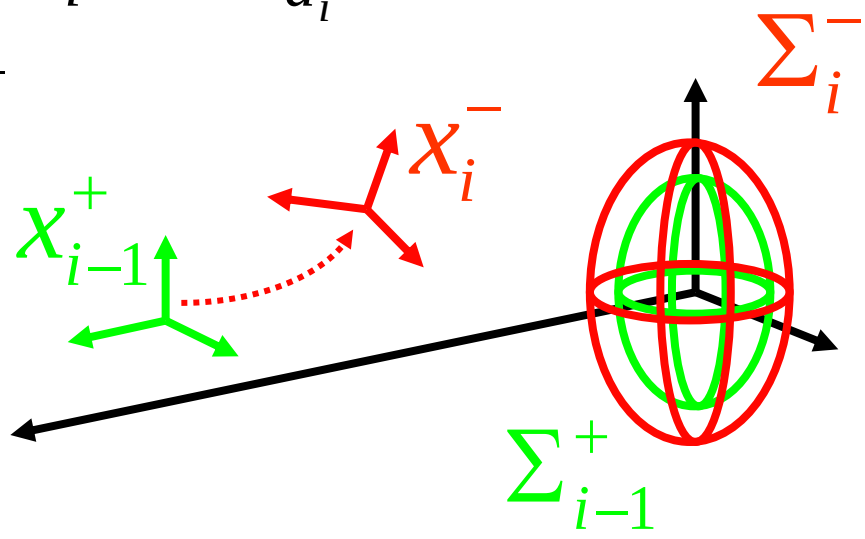
a priori state, error covariance, measurement

Notation: X^-: Prediction, X^+: Correction

$$X_i^- = D_i X_{i-1}^+$$

$$\Sigma_i^- = D_i \Sigma_{i-1}^+ D_i^T + \Sigma_{d_i}$$

$$Y_i^- = M\ X_i^-$$



Measurement Update (Correct)

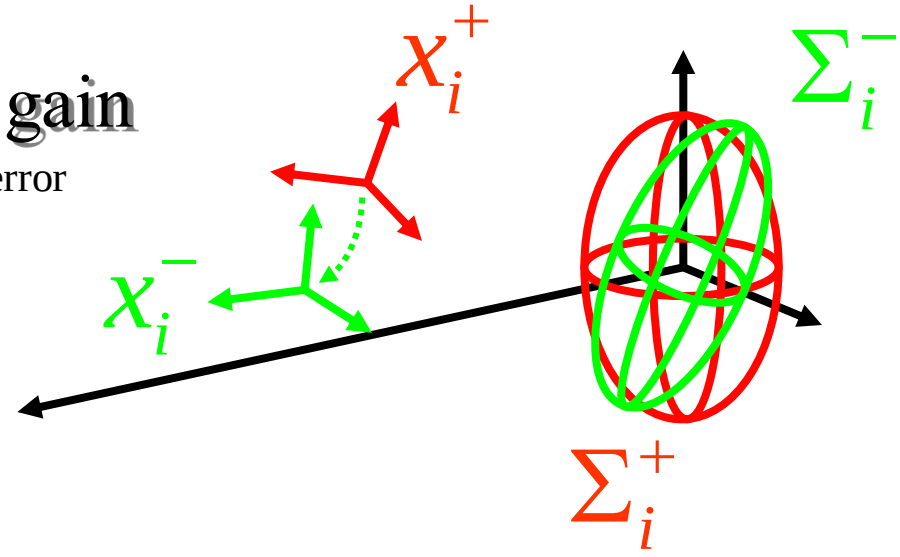
a posteriori state and error covariance

$$X_i^+ = X_i^- + K_i\left(Y_i - M\ X_i^-\right)$$

$$\Sigma_i^+ = \left(I + K_i M\right)\Sigma_i^-$$

Kalman gain

Minimizes posteriori error covariance



The Kalman Gain

Weights between prediction and measurements to posteriori error covariance

$$K_i = \Sigma_i^- M_i^T \left(M_i \Sigma_i^- M_i^T + \Sigma_{m_i} \right)^{-1}$$

For no measurement uncertainty: $\Sigma_{m_i} = 0$

$$K_i = \Sigma_i^- M_i^T M_i^{-T} \left(\Sigma_i^- \right)^{-1} M_i^{-1} = M_i^{-1}$$

$$x_i^+ = x_i^- + M_i^{-1} \left(y_i - M_i x_i^- \right) = M_i^{-1} y_i$$

State is deduced only from measurement

The Kalman Gain

Simple univariate (scalar) example

$$K = \frac{\sigma^-}{\sigma^- + \sigma_m}$$

a posteriori state and error covariance

$$x_i^+ = x_i^- + K\left(y_i - x_i^-\right)$$

$$\sigma_i^+ = \left(1 + K\right)\sigma_i^-$$

Summary

PREDICT

$$x_i^- = D_i x_{i-1}^+$$

$$\Sigma_i^- = D_i \Sigma_{i-1}^+ D_i^T + \Sigma_{d_i}$$

CORRECT

$$x_i^+ = x_i^- + K_i\left(y_i - M_i x_i^-\right)$$

$$\Sigma_i^+ = \left(I + K_i M_i\right)\Sigma_i^-$$

$$K_i = \Sigma_i^- M_i^T \left(M_i \Sigma_i^- M_i^T + \Sigma_{m_i}\right)^{-1}$$

Example: Estimating a Constant

The state transition matrix $D = I$

$$x_i^- = Dx_{i-1}^+ + w_i = x_{i-1}^+ + w_i$$

The measurement matrix $M = I$

$$y_i = Mx_i^- + \varepsilon_i = x_i^- + \varepsilon_i$$

Prediction

$$x_i^- = x_{i-1}^+$$

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

Measurement Update

$$x_i^+ = x_i^- + K_i\left(y_i - x_i^-\right)$$

$$\Sigma_i^+ = \left(1 + K_i\right)\Sigma_i^-$$

$$K_i = \frac{\Sigma_i^-}{\Sigma_i^- + \Sigma_{m_i}}$$

Setup/Initialization

Generate 50 samples centered around -0.37727 with standard deviation of 0.1 (var 0.01).

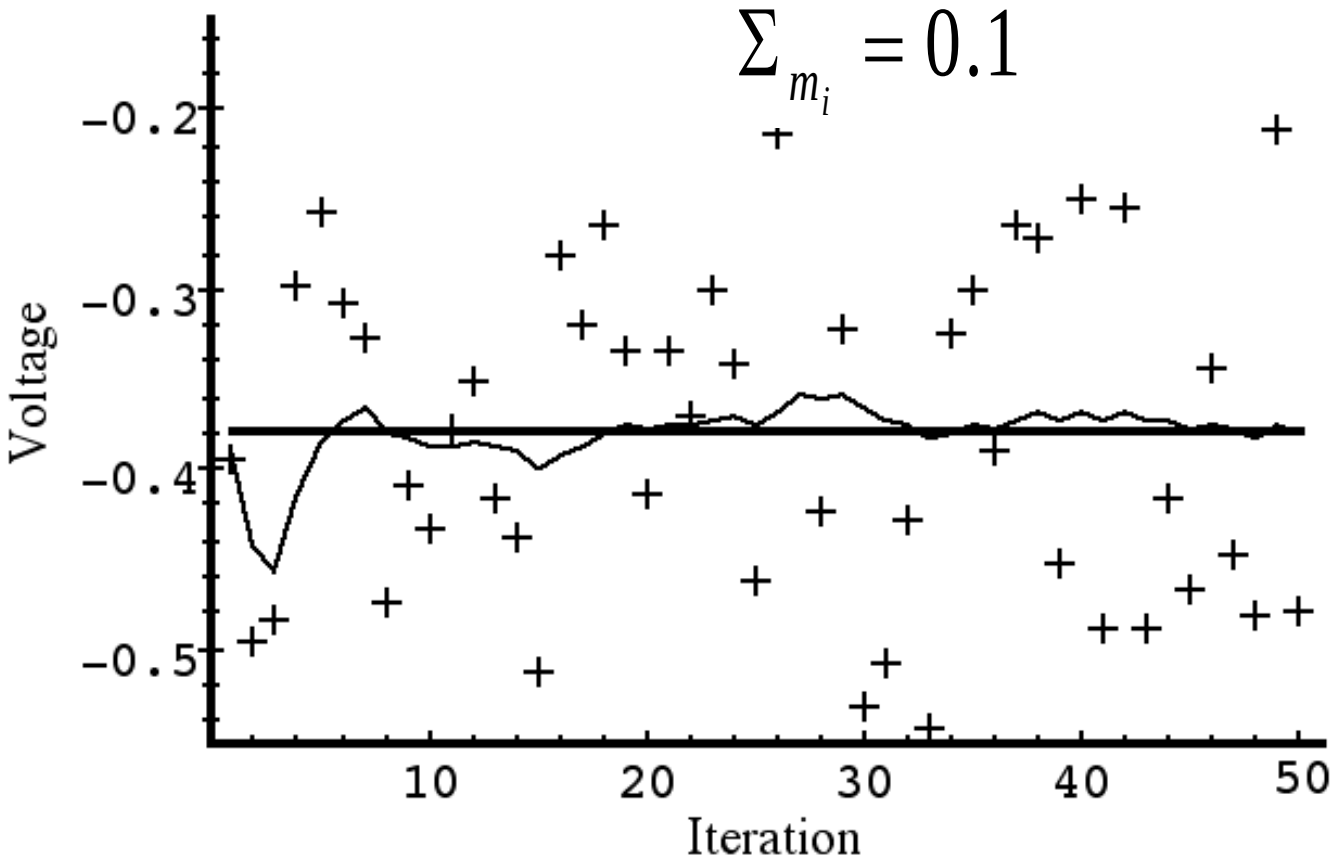
$$\Sigma_{d_i} = 10^{-5}$$

$$\bar{x_0} = 0$$

$$\bar{\Sigma_0} = 1$$

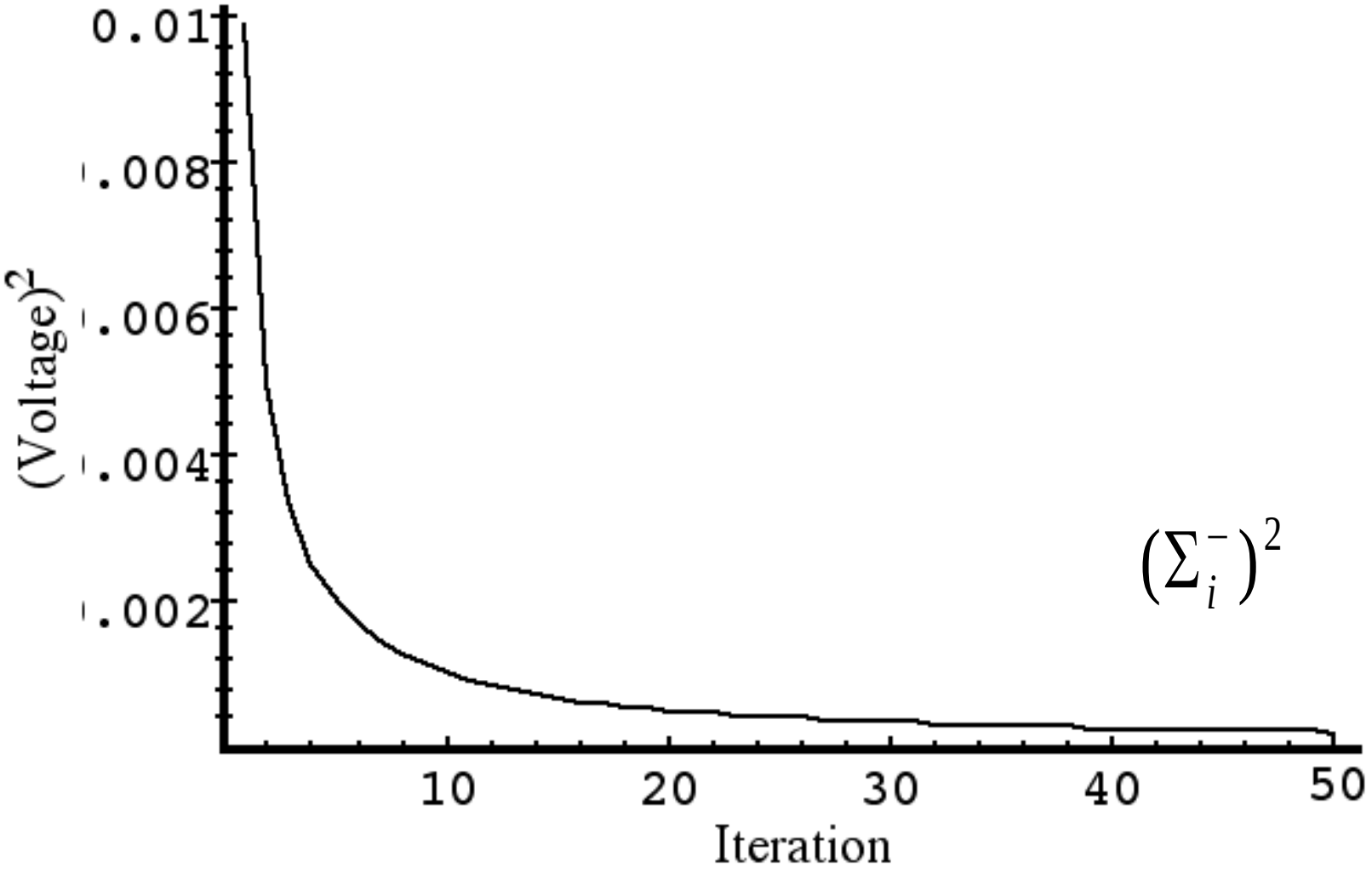
State and Measurements

$\Sigma_{m_i}=0.1^2=0.01$



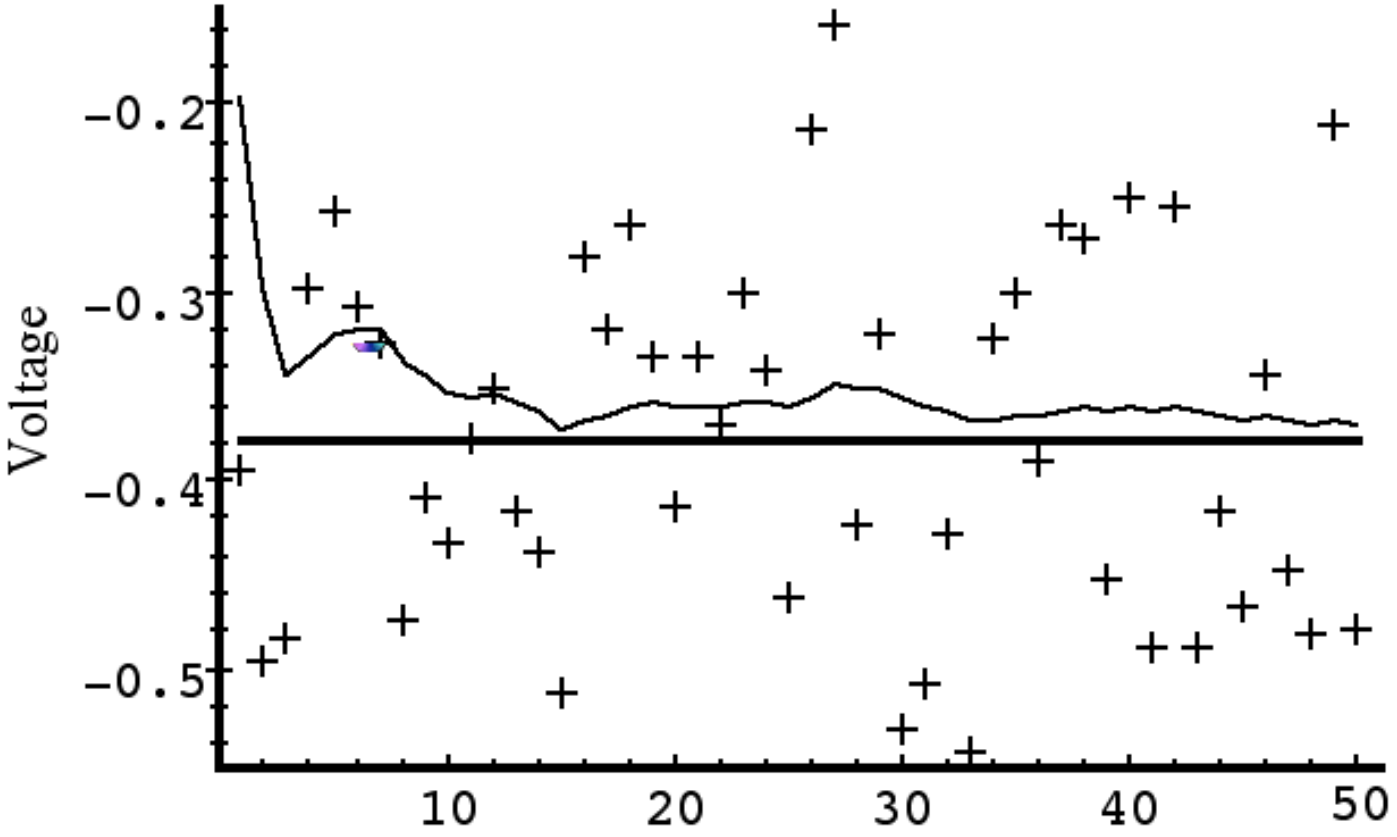
Filter was told the correct measurement variance.

Error Covariance (initially 1)



State and Measurements

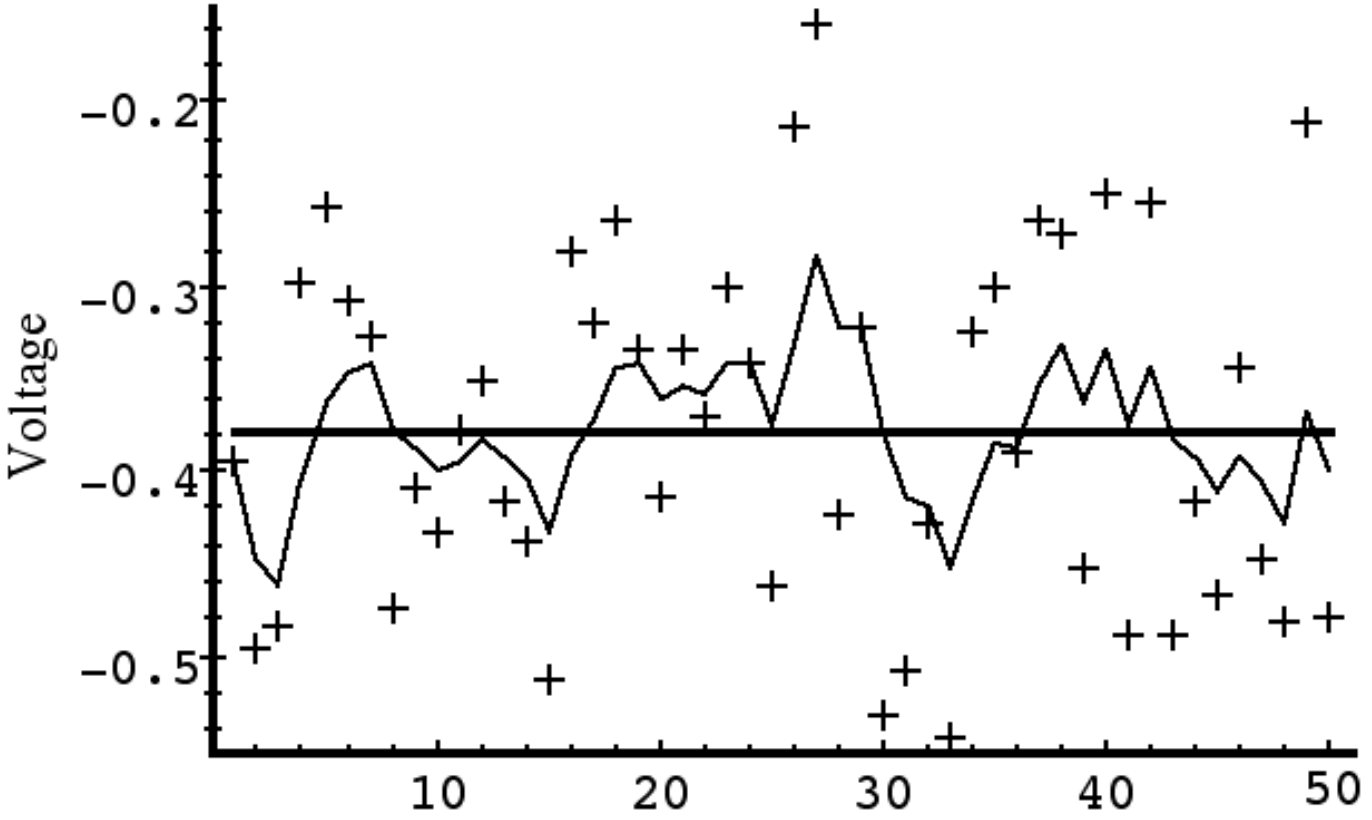
$$\Sigma_{m_i} = 1$$



Filter was told that the measurement variance was 100 times greater (i.e. 1) so it was "slower" to believe the measurements.

State and Measurements

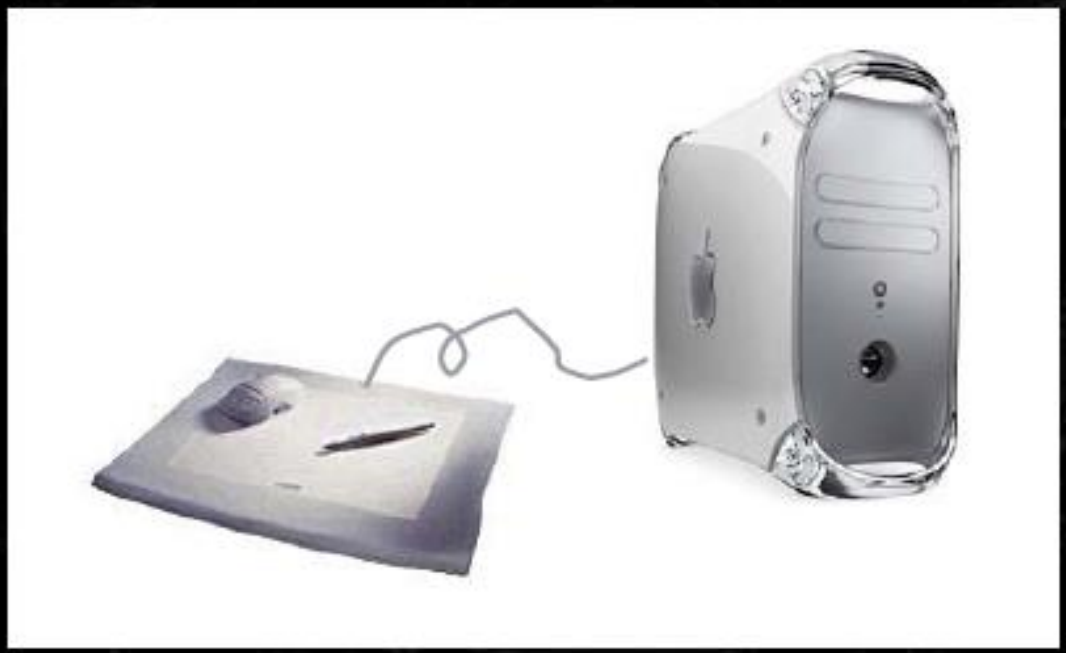
$\sum_{m_i} = 0.01^2 = 0.0001$



Filter was told that the measurement variance was 100 times smaller (i.e. 0.0001) so it was very "quick" to believe the noisy measurements.

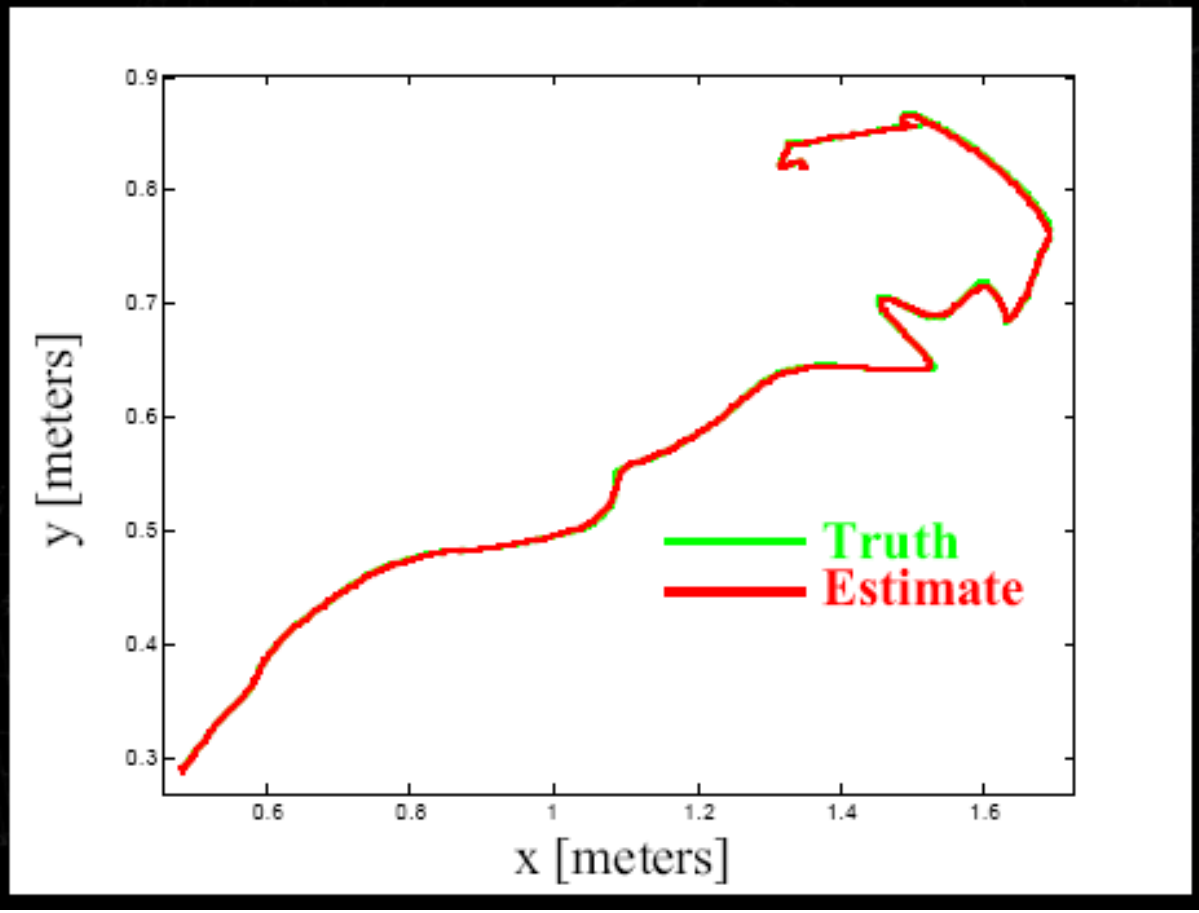
Demonstration own experiments

Apparatus: 2D Tablet

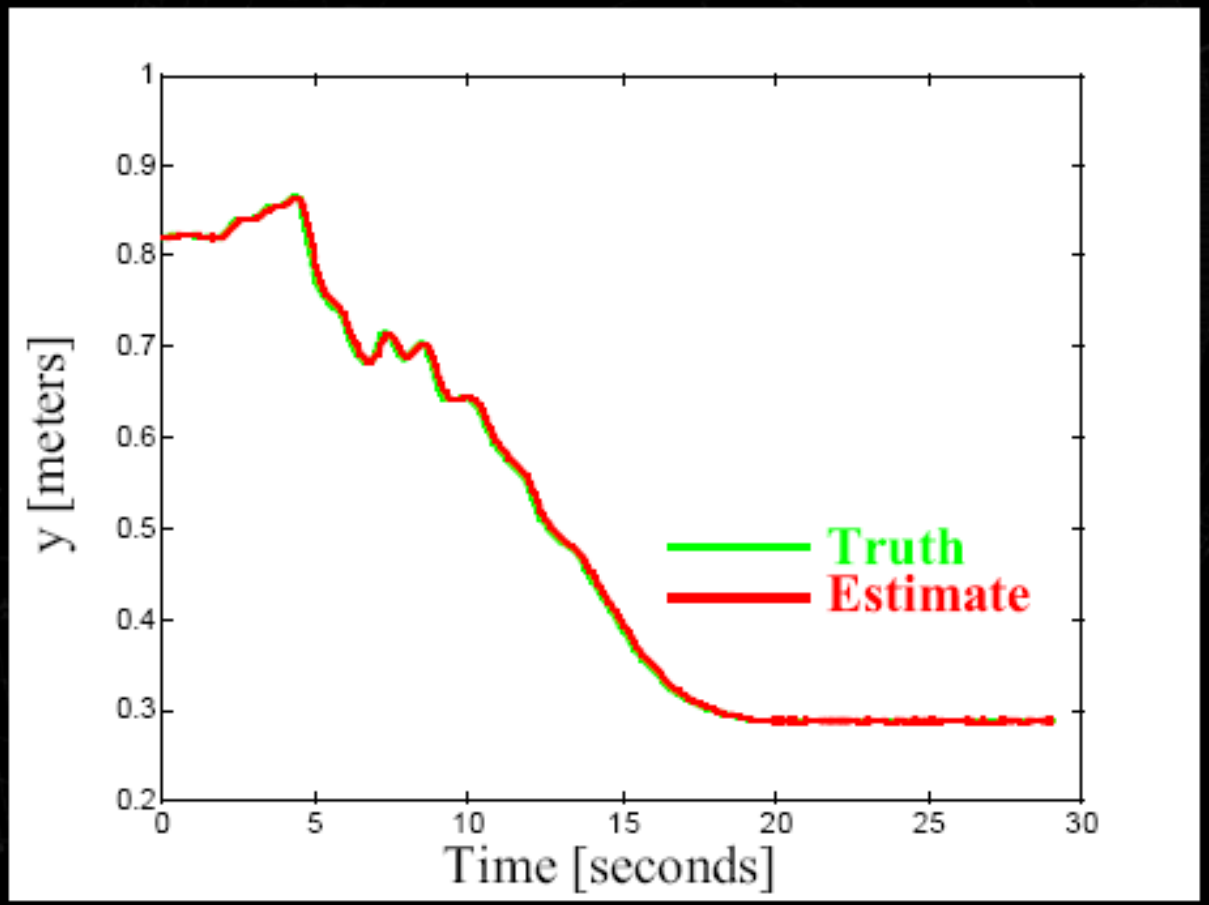


SIGGRAPH 2001 EXPLORE INTERACTION AND DIGITAL IMAGES

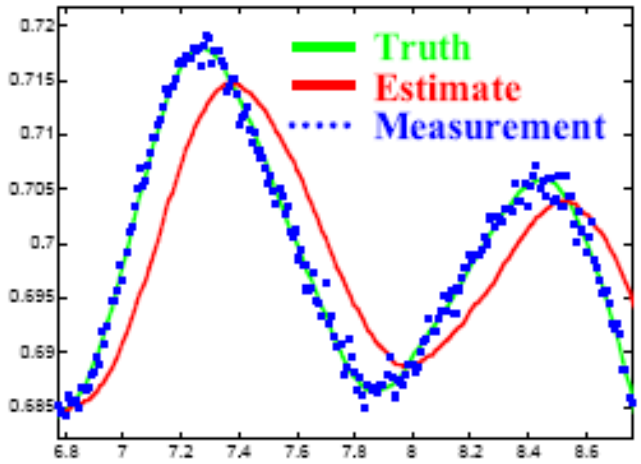
Results: XY Track

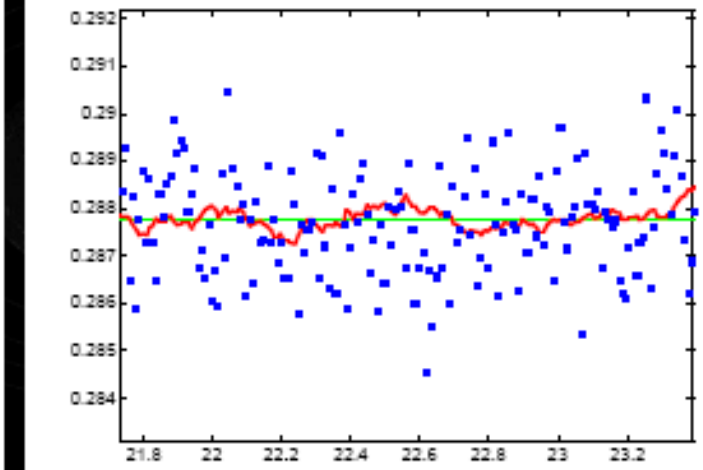


Y Track: Moving then Still

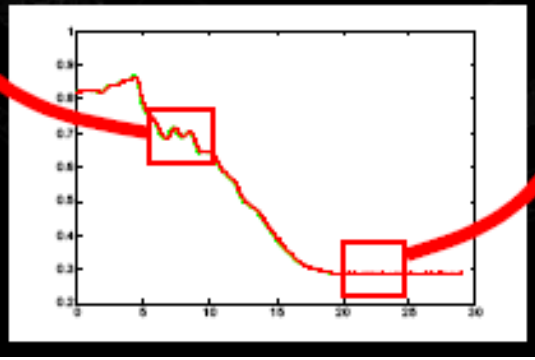


Motion-Dependent Performance





significant *latency* when moving…



…relatively *smooth* when not

2D Position-Velocity (PV)

Process Model (PV)

state transition

$$\begin{bmatrix} 1 & 0 & dt & 0 \\ 0 & 1 & 0 & dt \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

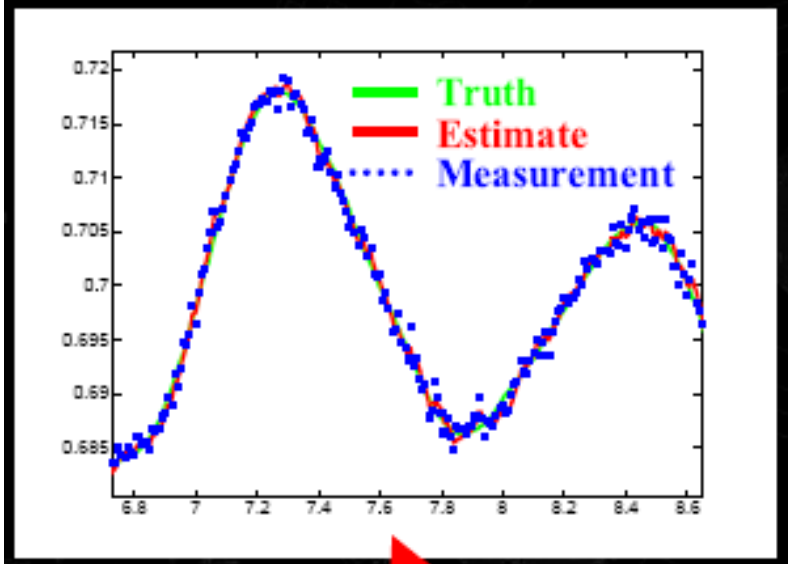
state

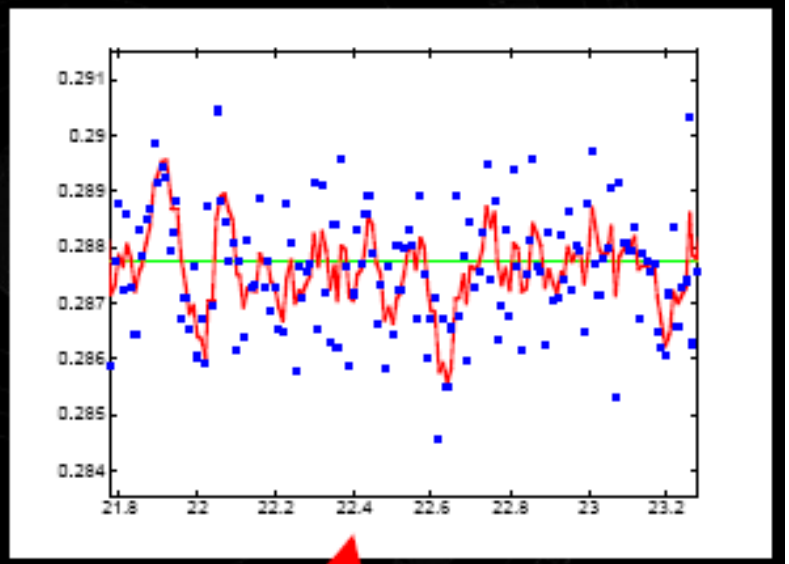
$$\begin{bmatrix} x \\ y \\ dx/dt \\ dy/dt \end{bmatrix}$$

SIGGRAPH 2001

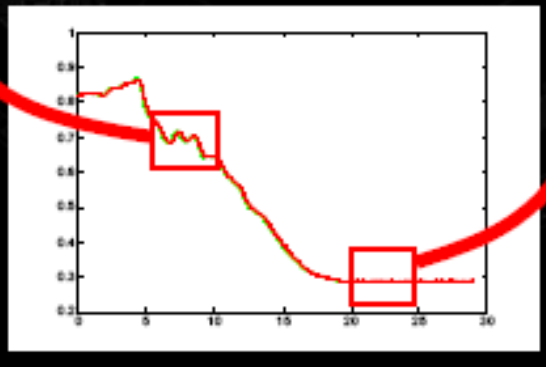
2D Position-Velocity (PV)

Different Performance





improved latency when moving…



…relatively *noisy* when not

Example: Hand Gesture Recognition and Tracking

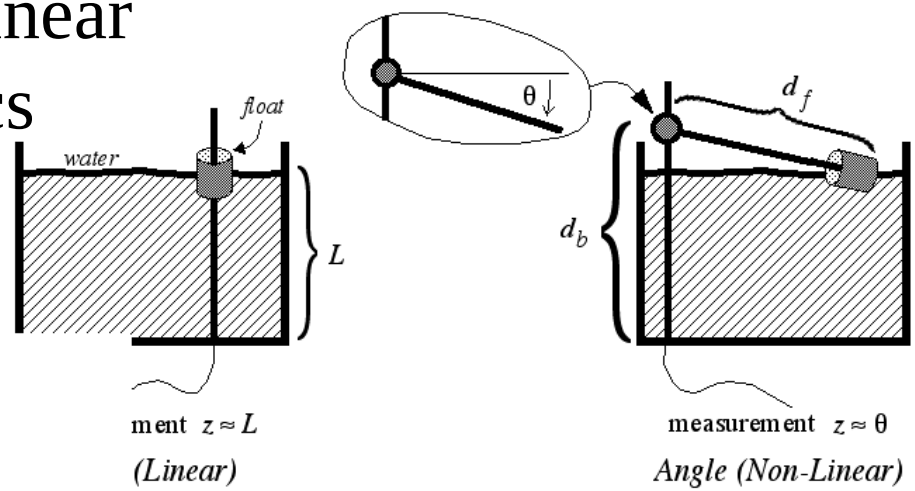
Kalman Filter Web Site

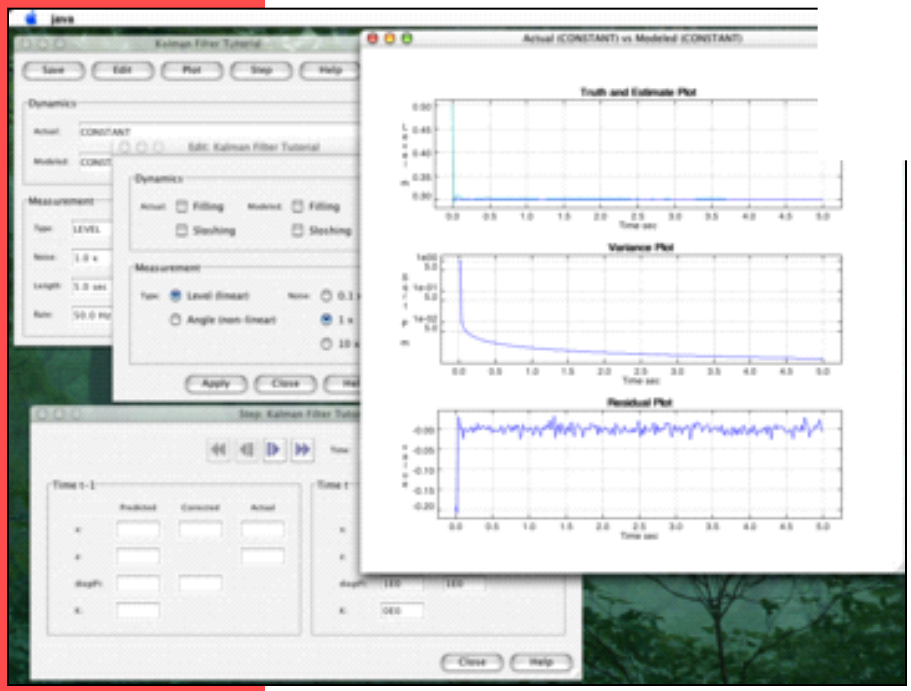
http://www.cs.unc.edu/~welch/kalman/

- Electronic and printed references
  - Book lists and recommendations
  - Research papers
  - Links to other sites
  - Some software
- News

Java-Based KF Learning Tool

- On-line 1D simulation
- Linear and non-linear
- Variable dynamics





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

KF Course Web Page

http://www.cs.unc.edu/~tracker/ref/s2001/kalman/index.html

(http://www.cs.unc.edu/~tracker/)

- Java-Based KF Learning Tool
- KF web page

Relevant References

- Azarbayejani, Ali, and Alex Pentland (1995). "Recursive Estimation of Motion, Structure, and Focal Length," IEEE Trans. Pattern Analysis and Machine Intelligence 17(6): 562-575.
- Dellaert, Frank, Sebastian Thrun, and Charles Thorpe (1998). "Jacobian Images of Super-Resolved Texture Maps for Model-Based Motion Estimation and Tracking," IEEE Workshop on Applications of Computer Vision (WACV'98), October, Princeton, NJ, IEEE Computer Society.
- http://mac-welch.cs.unc.edu/~welch/COMP256/

Extensions: Particle Filtering, Condensation

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

A. Blake, B. Bascle, M. Isard, and J. MacCormick, Statistical models of visual shape and motion, in *Phil. Trans. R. Soc. A.*, vol. 356, pp. 1283–1302, 1998

B. Isard, M., Blake, and A., Condensation — conditional density propagation for visual tracking, in *Int. J. Computer Vision*, vol. 28, no. 1, pp. 5–28, 1998

C. Blake, A., Isard, M.A., Reynard, and D., Learning to track the visual motion of contours, in *J. Artificial Intelligence*, vol. 78, pp. 101–134, 1995

Condensation Algorithm (Blake et al.)

Extensions: Particle Filtering, Condensation

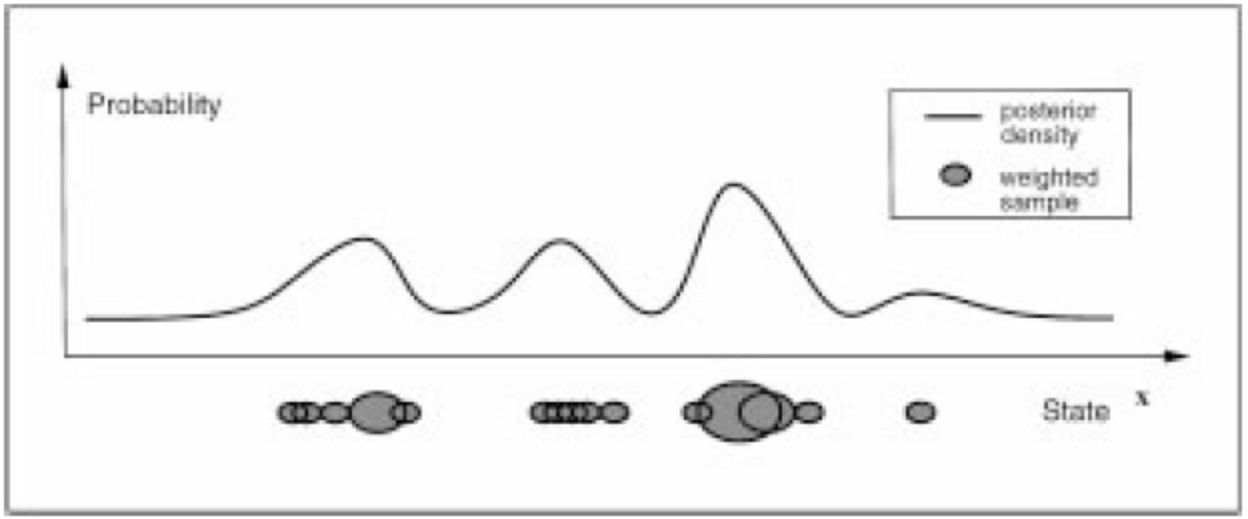


Figure 3. Factored sampling: a set of points $s^{(n)}$, the centres of the blobs in the figure, is sampled randomly from a prior density $p(\mathbf{x})$. Each sample is assigned a weight π_i (depicted by blob area) in proportion to the value of the observation density $p(\mathbf{z} \mid \mathbf{x} = s^{(n)})$. The weighted point-set then serves as a representation of the posterior density $p(\mathbf{x} \mid \mathbf{z})$, suitable for sampling. The one-dimensional case illustrated here extends naturally to the practical case that the density is defined over several position and shape variables.

Extensions: Particle Filtering, Condensation

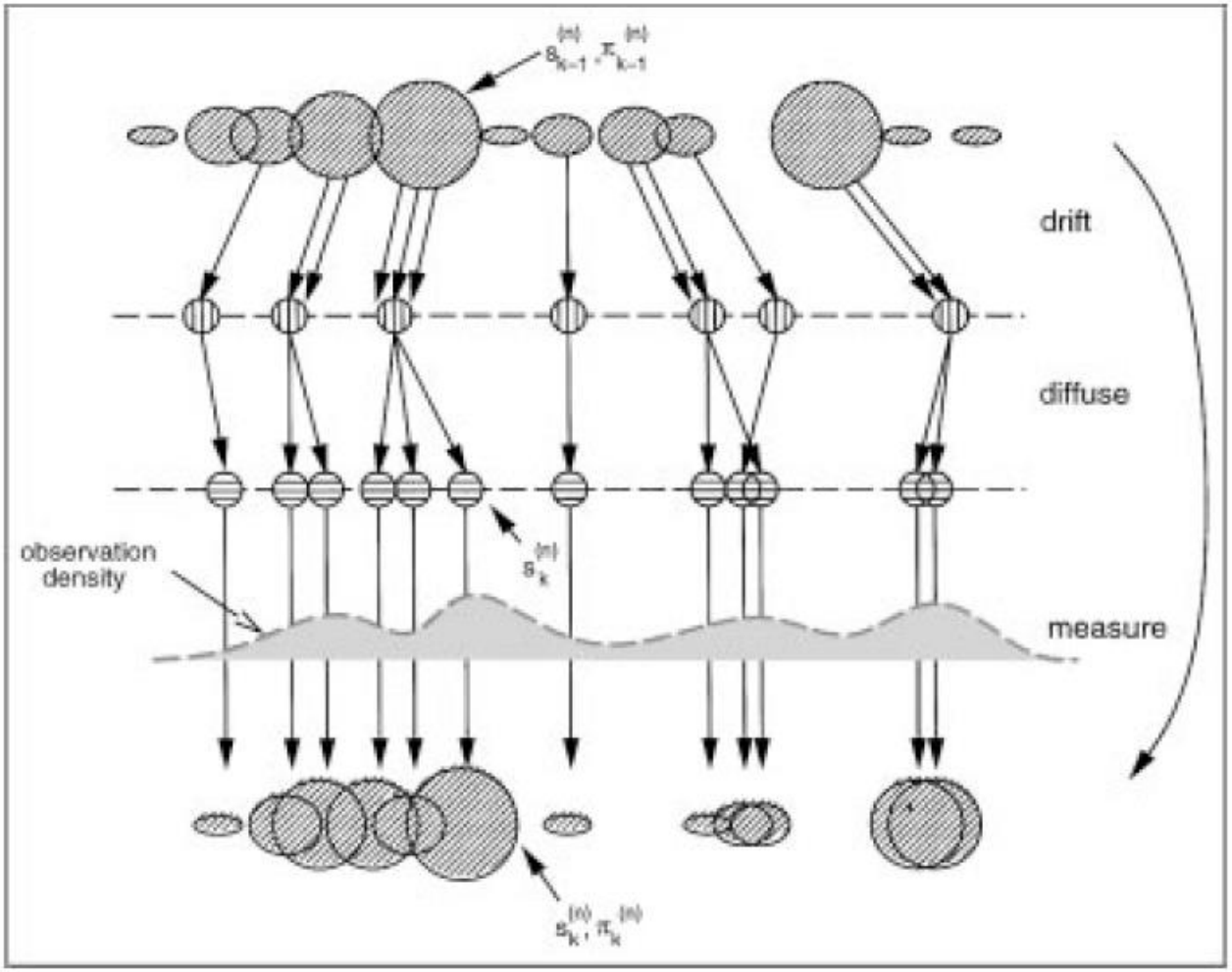


Figure 5. One time-step in the CONDENSATION algorithm: Each of the three steps—drift-diffuse-measure—of the probabilistic propagation process of Fig. 2 is represented by steps in the CONDENSATION algorithm.

Extensions: Particle Filtering, Condensation

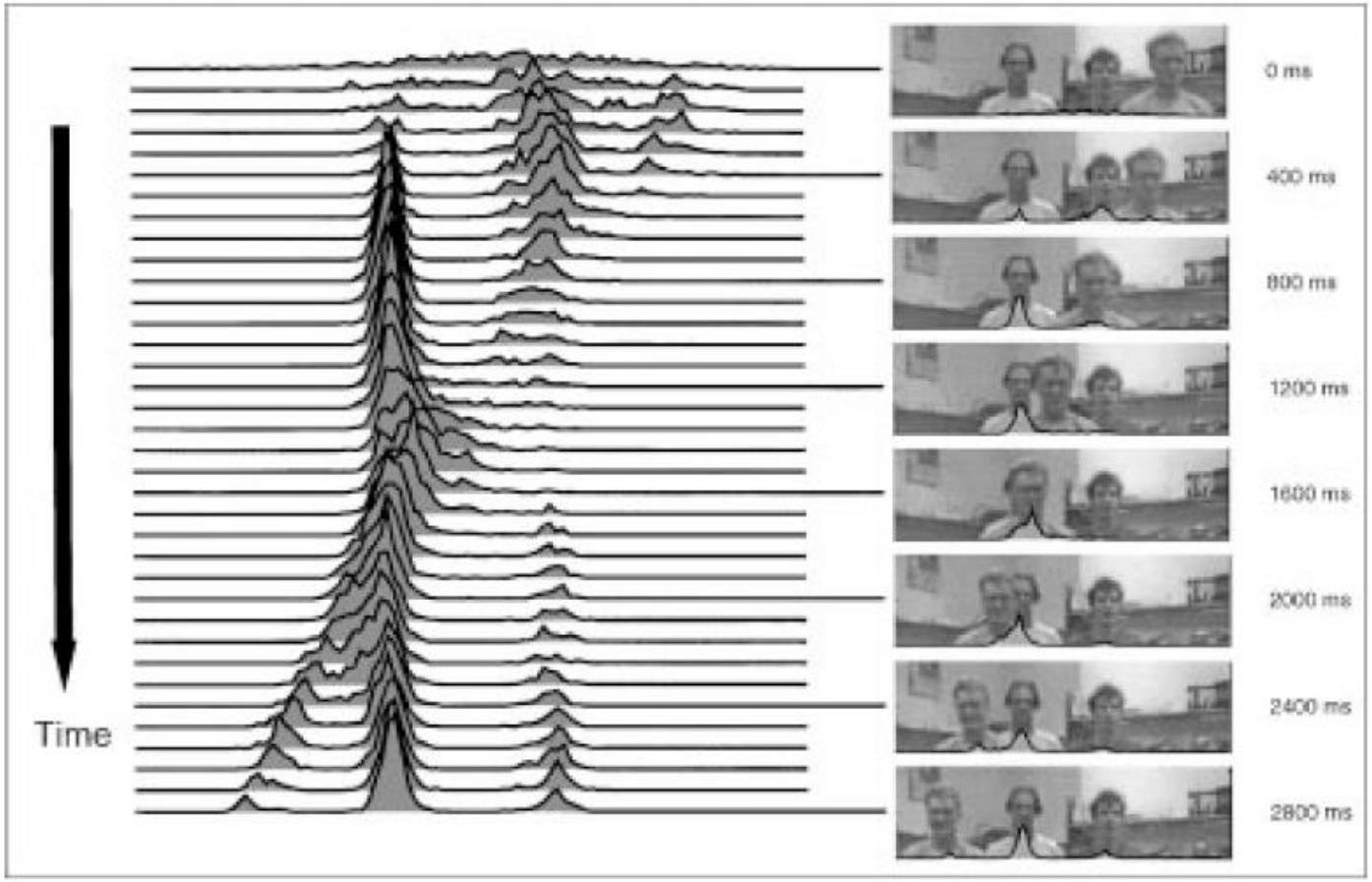


Figure 10. Tracking with multi-modal state-density: an approximate depiction of the state-density is shown, computed by smoothing the distribution of point masses $s_t^{(1)}, s_t^{(2)}, \ldots$ in the CONDENSATION algorithm. The density is, of course, multi-dimensional; its projection onto the horizontal translation axis is shown here. The initial distribution is roughly Gaussian but this rapidly evolves to acquire peaks corresponding to each of the three people in the scene. The right-most peak drifts leftwards, following the moving person, coalescing with and separating from the other two peaks as it moves. Having specified a tracker for one person we effectively have, for free, a multi-person tracker, owing to the innate ability of the CONDENSATION algorithm to maintain multiple hypotheses.