Kalman Filtering
(Intelligent Autonomous Robotics)

Dr. Subramanian Ramamoorthy
School of Informatics

Recall: Estimation

Last time, we looked at the following:

- Estimation of "true state" from noisy measurements
- We *assumed* a sensor model ($y = Hx + w$) where the state is linearly transformed and corrupted with noise
- Then, we looked at a variety of estimation procedures
  - Least squares
  - Weighted least squares
  - Recursive formulation of WLS
  - Maximum likelihood estimation
  - Nonlinear sensor model (i.e., $y = h(x) + w$), which resulted in the ideas of Monte Carlo methods and Unscented estimation

Agenda for this lecture

- Recap of the basic multiple observations scenario
- Introduction of *dynamics*
- The Kalman filter algorithm
  - Discrete time, linear dynamics
  - Continuous time, linear dynamics
  - Extensions for nonlinear dynamics
  - Multiple-model estimation

Simple Estimation Problem

- Consider a ship in choppy waters
- Needs to navigate using a distant landmark (star)
- Can make multiple measurements of the same ground truth (anticipating the rest of the lecture, let us say – successively in time)
- We know *(a priori)* one crucial fact – the variance of each measurement
- Then, what is the true value of the distance to the landmark?

Updating Estimated State

How to combine multiple measurements?

Estimate 1:

Given measurements $z_1, \sigma_{z_1}^2$

$\hat{x}_1 = z_1$

$\hat{\sigma}_1^2 = \sigma_{z_1}^2$

Estimate 2:

Next, given measurements $z_2, \sigma_{z_2}^2$

$\hat{x}_1 = ?$

$\hat{\sigma}_1^2 = ?$

Answer: Variance-weighted Sum

Given measurements $z_1, \sigma_{z_1}^2$ and $z_2, \sigma_{z_2}^2$,

$$\hat{x_2} = \left[\frac{\sigma_{z_2}^2}{\sigma_{z_1}^2 + \sigma_{z_2}^2}\right] z_1 + \left[\frac{\sigma_{z_1}^2}{\sigma_{z_1}^2 + \sigma_{z_2}^2}\right] z_2$$

$$\hat{x_2} = z_1 + \left[\frac{\sigma_{z_1}^2}{\sigma_{z_1}^2 + \sigma_{z_2}^2}\right] [z_2 - z_1]$$

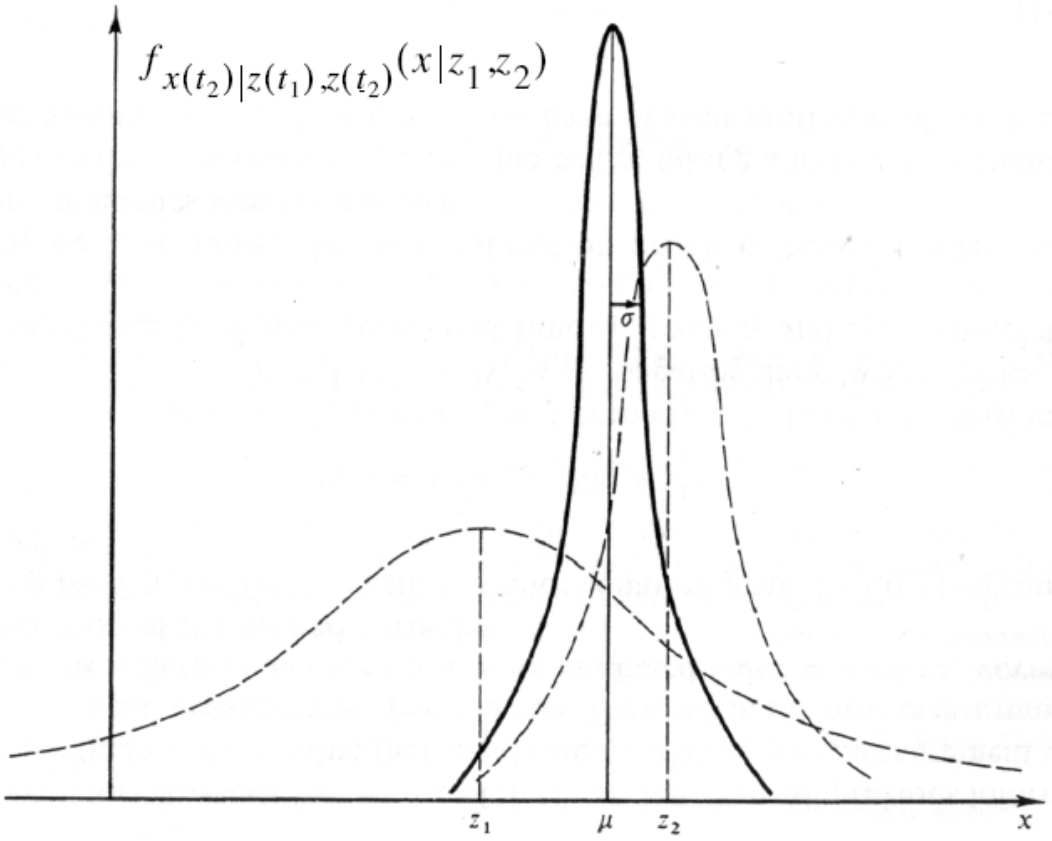
$$\hat{x_2} = \hat{x_1} + K_2 [z_2 - \hat{x_1}]$$

where $K_2 = \frac{\sigma_{z_1}^2}{\sigma_{z_1}^2 + \sigma_{z_2}^2}$

The resulting variance is simply combined as,

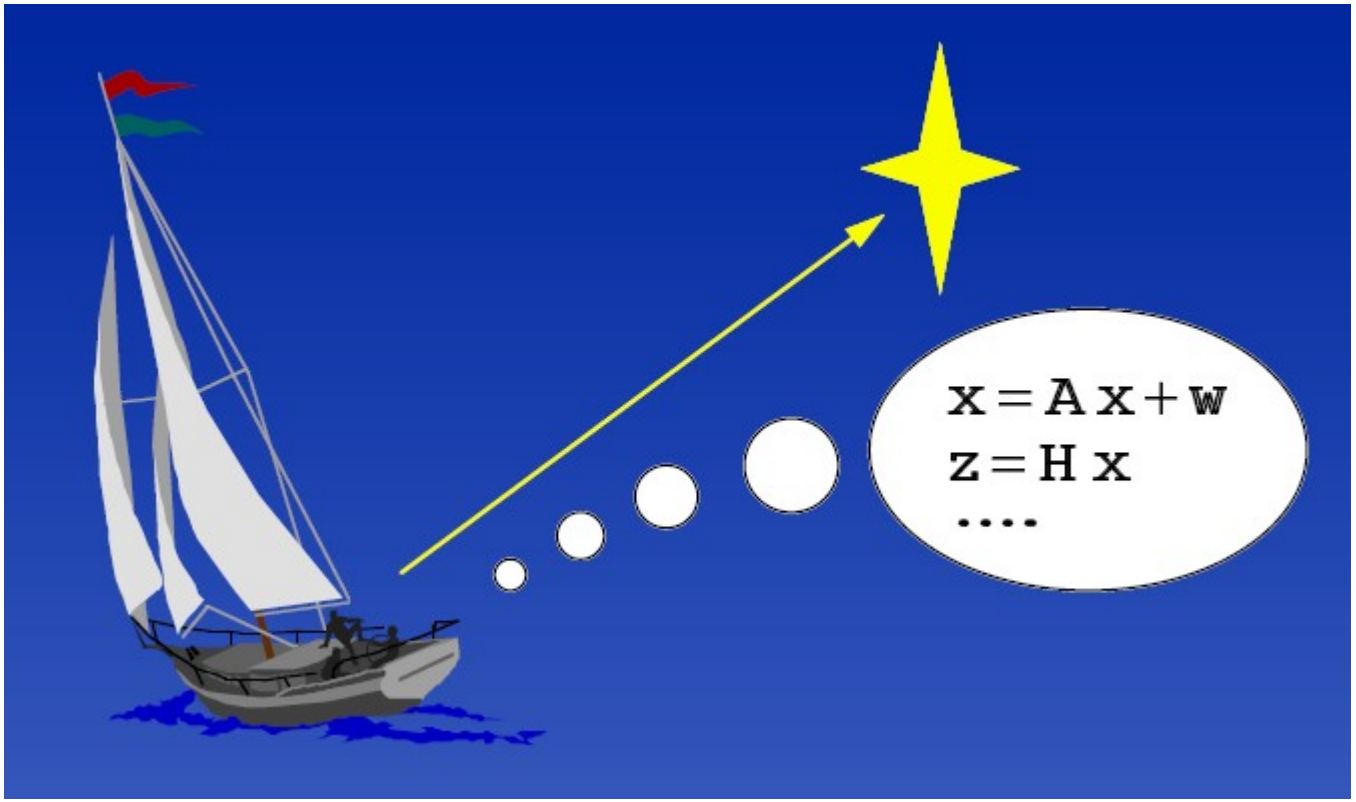
$$\frac{1}{\sigma^2} = \frac{1}{\sigma_{z_1}^2} + \frac{1}{\sigma_{z_2}^2}$$

In pictures…



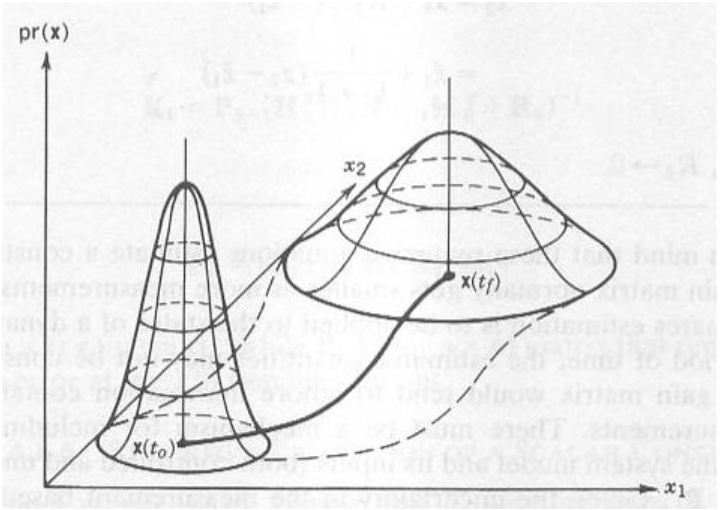
Conditional density of position based on data z_1 and z_2.

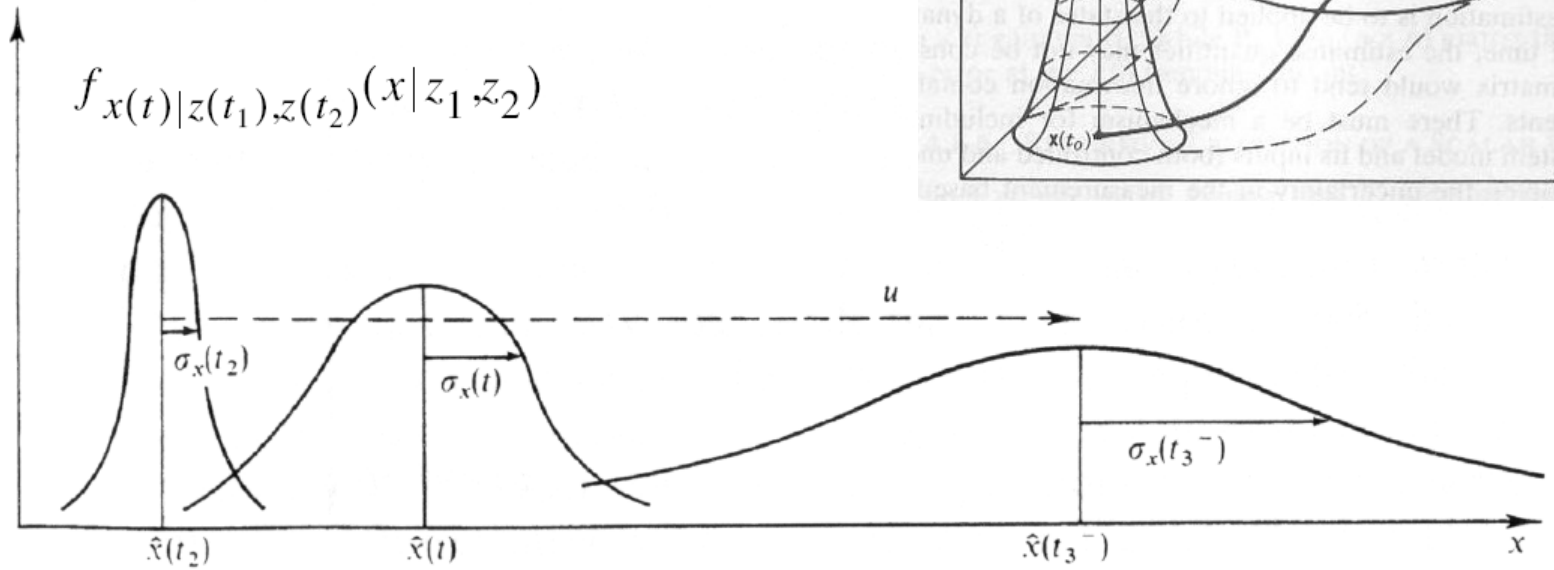
What happens if we Add Dynamics?



Effect of Adding System Dynamics

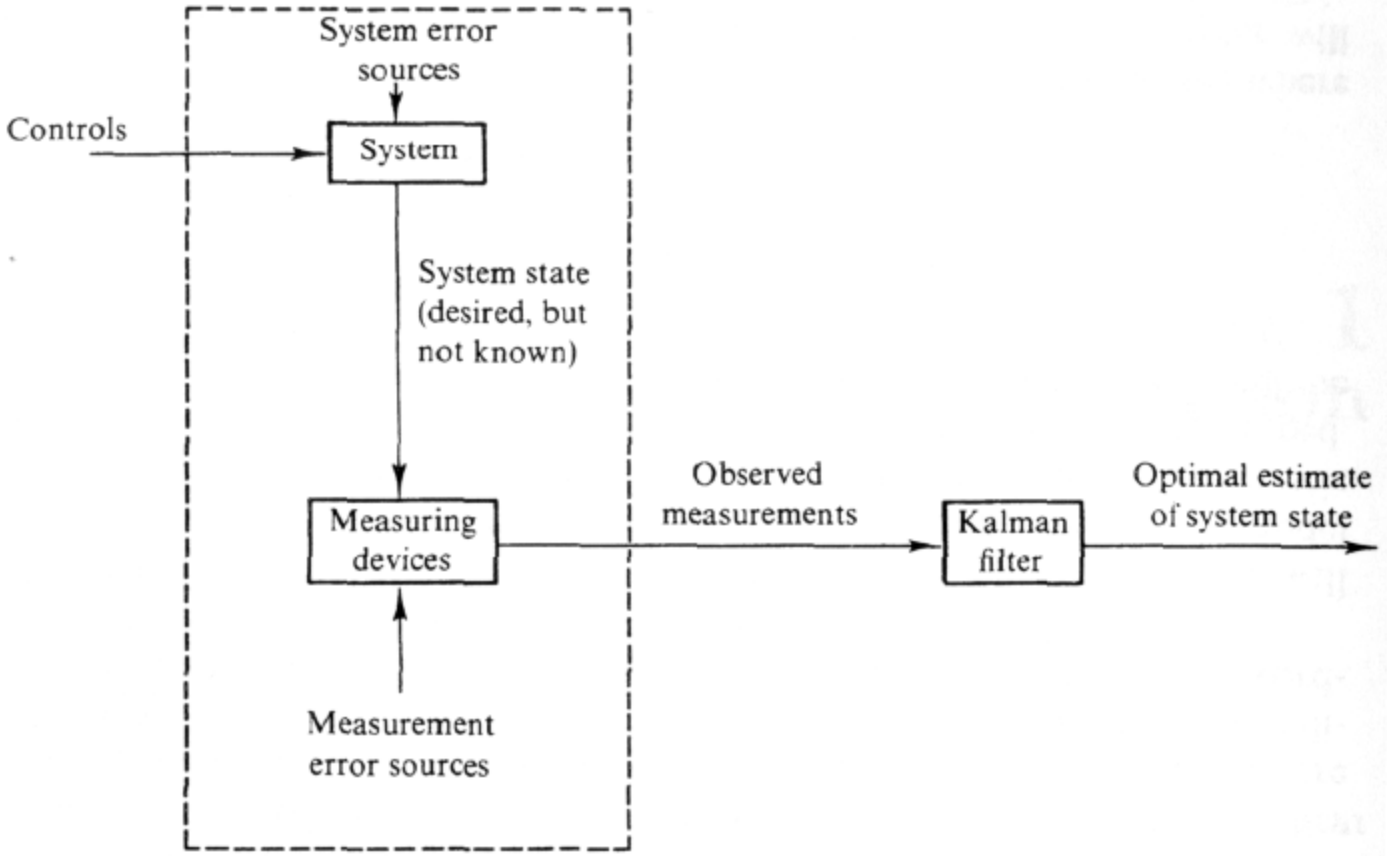
The uncertainty is amplified – density function gets "squashed"



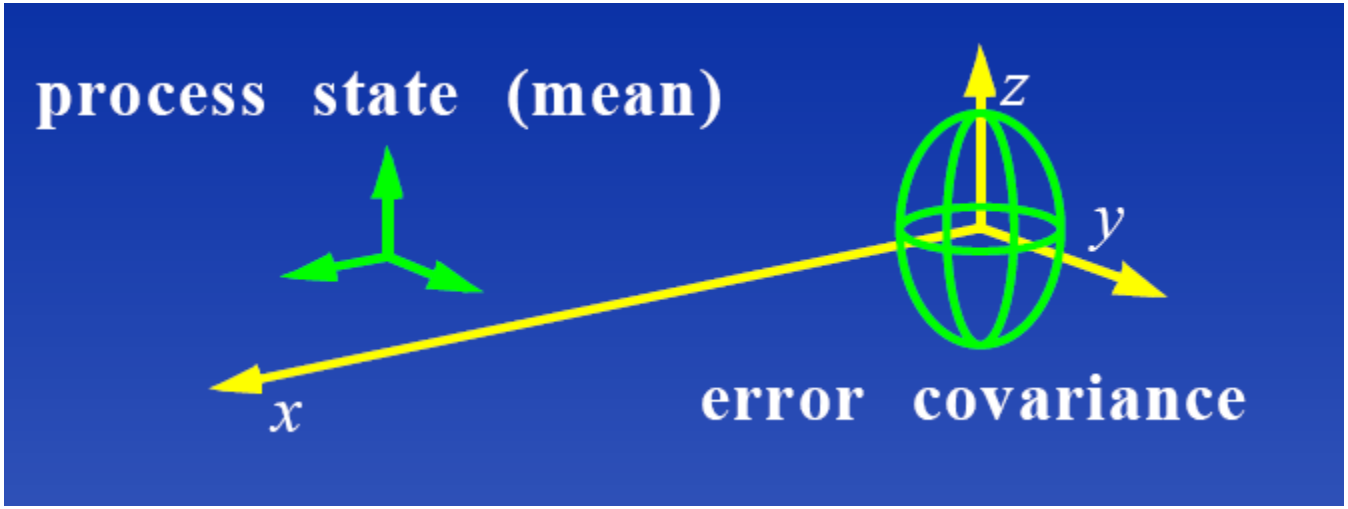


Propagation of conditional probability density.

Idea of the "Kalman Filter"



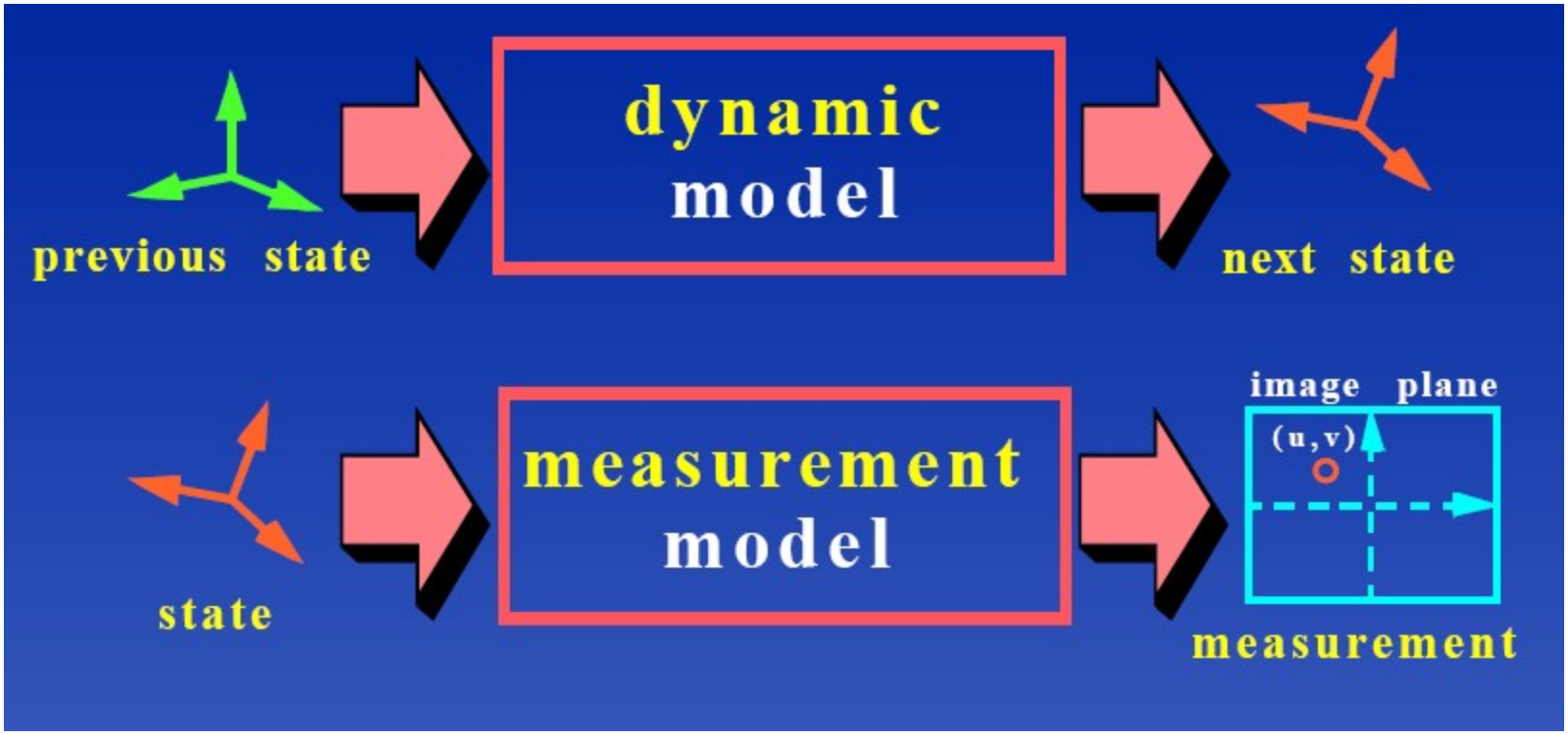
Key Variables: First Two Statistical Moments



Discrete Kalman Filter: Key Ingredients

- Discrete process model
  - State change over time
  - Linear difference equation
- Discrete measurement model
  - Relationship between state and measurement
  - Linear function (this is what we dealt with in last lecture!)
- Model parameters
  - Process noise characteristics
  - Measurement noise characteristics

Discrete Kalman Filter: Two Models



Discrete Kalman Filter: Model Equations

System Dynamics:

$$x_{k+1} = Ax_k + w_k, x_k \in \Re^n, w_k \sim \mathcal{N}(0, Q)$$

where A is the 'state transition matrix'.

Measurement Process:

$$z_k = Hx_k + v_k, z_k \in \Re^m, v_k \sim \mathcal{N}(0, R)$$

Complete Model Specification

System Dynamics:

$$x_{k+1} = Ax_k + w_k, x_k \in \Re^n, w_k \sim \mathcal{N}(0, Q)$$

where A is the 'state transition matrix'.

Measurement Process:

$$z_k = Hx_k + v_k, z_k \in \Re^m, v_k \sim \mathcal{N}(0, R)$$

$\hat{x}_k^-$: *a priori* state estimate.

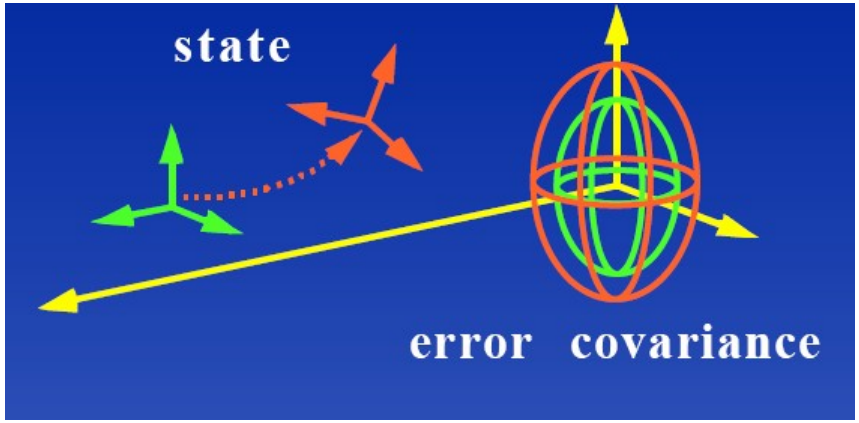
$\hat{x}_k$: *a posteriori* state estimate.

$P_k^- = E[(x_k - \hat{x}_k^-)(x_k - \hat{x}_k^-)']$, the *a priori* estimate error covariance

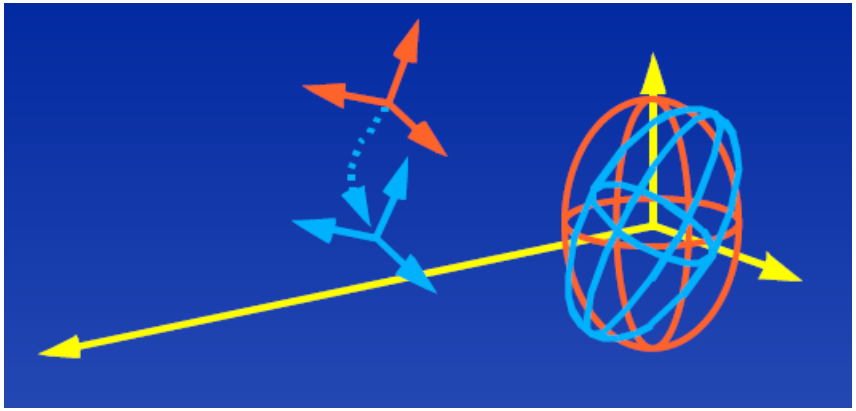
$P_k = E[(x_k - \hat{x}_k)(x_k - \hat{x}_k)']$ the *a posteriori* estimate error covariance

How the Filter Works

- Time Update or Prediction (*a priori* estimates) - Project state and covariance forward in time



- Measurement Update or Correction (*a posteriori* estimates) - Update variables based on a noise measurement



Discrete Kalman Filter – In Equations

Prediction step:

$$\hat{x}_{k+1}^{-} = Ax_k$$

$$P_{k+1}^{-} = AP_kA + Q$$

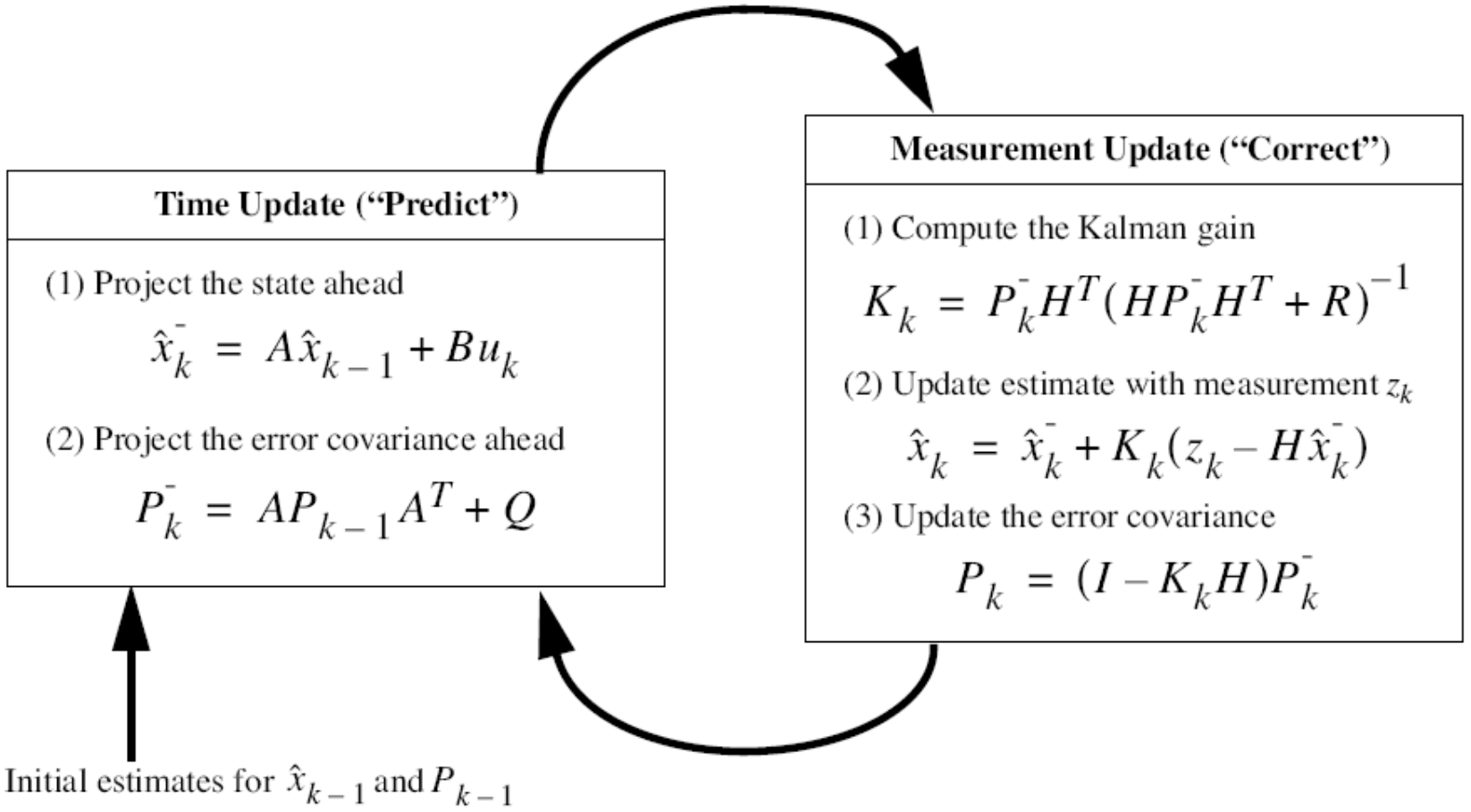
Correction step:

$$\hat{x}_k = \hat{x}_k^{-} + K_k(z_k - H\hat{x}_k^{-})$$

$$P_k = (I - K_kH)P_k^{-}$$

where $K_k = P_k^{-}H'(HP_k^{-}H' + R)^{-1}$ is the *Kalman Gain.*

Discrete Kalman Filter: The Complete Loop



Example: Estimating a (Noisy) Constant

$$x_{k+1} = x_k$$
$$z_k = x_k + v_k$$

Time Update:

$$\hat{x}_{k+1}^- = x_k$$

$$P_{k+1}^- = P_k$$

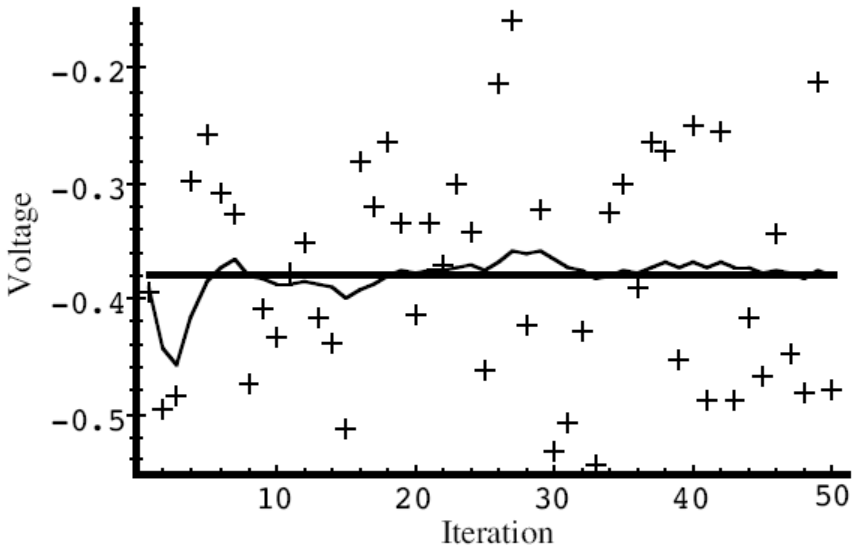
Kalman Gain:

$$K_k = \frac{P_k^-}{P_k^- + R}$$

Measurement Update:

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - \hat{x}_k^-)$$

$$P_k = (1 - K_k)P_k^-$$



Recap: What is the Kalman Filter?

- An optimal, recursive data processing algorithm
- Optimal in what sense?
  - If the system dynamics are linear
  - And system/sensor noise is Gaussian and white
  - Then, there is no alternate algorithm with lower MSE
  - Therefore, Optimal ⇔ Best Linear Unbiased Estimator

- Bayesian View of the Kalman Filter:

  It propagates the conditional probability density of the desired quantities, conditioned on the knowledge of the actual sensor data

Kalman Filter: Continuous Time-variant case

System and Sensor Dynamics:

$$\dot{x}(t) = F(t)x(t) + G(t)u(t) + L(t)w(t), x(0) = x_0; w(t) \sim \mathcal{N}(0, Q_c(t))$$
$$z(t) = H(t)x(t) + n(t); n(t) \sim \mathcal{N}(0, R_c(t))$$

Initial Conditions:

$$E[x(0)] = \hat{x}_0, E[x(0) - \hat{x}_0][x(0) - \hat{x}_0] = P_0$$

The discrete time optimal gain matrix and covariance update equation are:

$$K_{k-1} = P^+_{k+1} H'_{k-1} R_c^{-1}(t_{k-1})\delta t$$
$$\hat{P}_{k-1} = (I - K_{k-1}H_{k-1})P^-_{k-1}$$

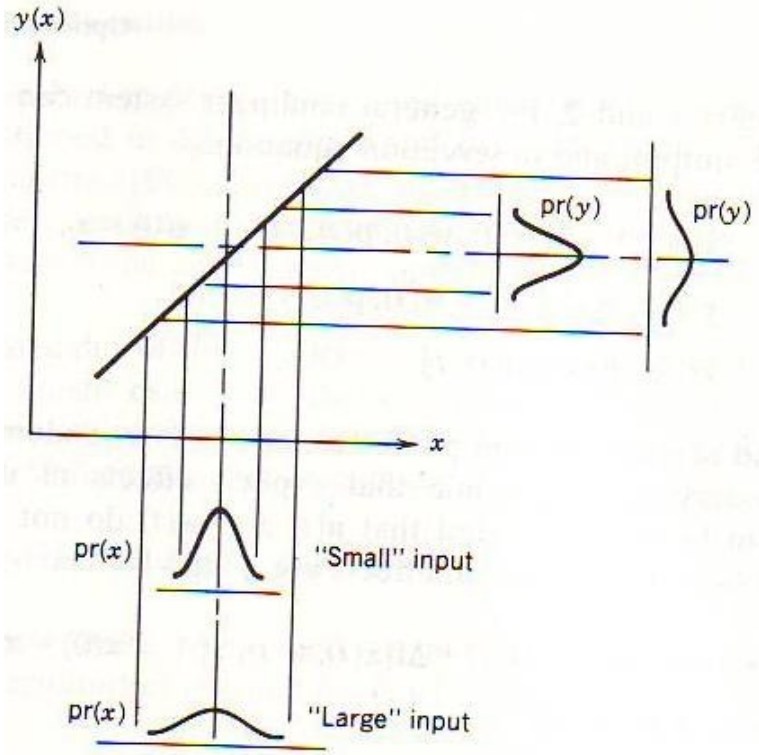
As $\delta t \rightarrow 0$, the evolution of covariance is described by a *matrix Riccati equation*,

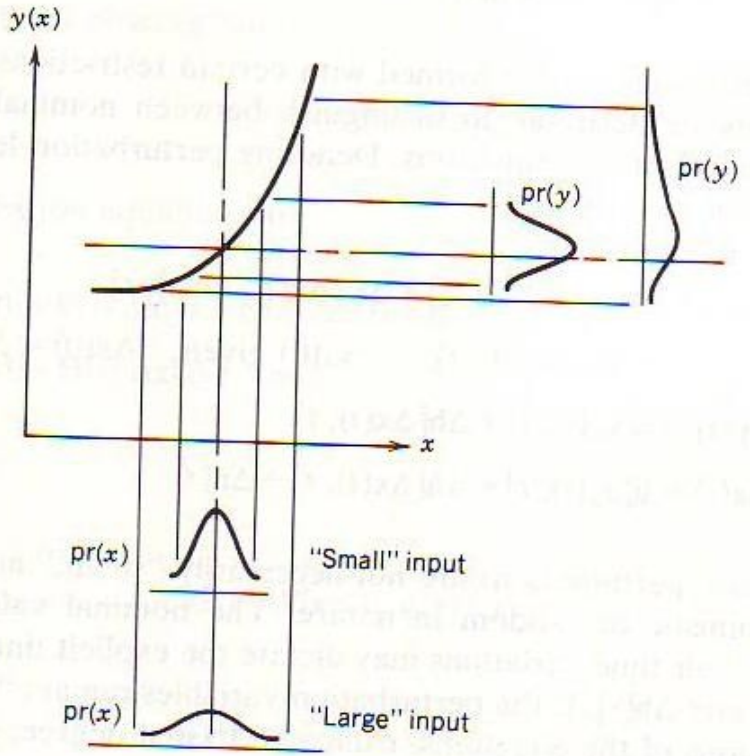
$$\dot{P} = F(t)P(t) + P(t)F'(t) + L(t)Q_c(t)L'(t) - P(t)H'(t)R_c^{-1}(t)H(t)P(t)$$

From this, the continuous time Kalman gain is derived as,

$$K_c(t) = P(t)H'(t)R_c^{-1}(t)$$

Dealing with Nonlinearities





Extended Kalman Filter - Setup

We have the nonlinear system,

$$x_k = f(x_{k-1}, u_k, w_{k-1})$$
$$z_k = h(x_k, v_k)$$

Consider the *a posteriori* noise-free approximation,

$$\tilde{x}_k = f(x_{k-1}, u_k, 0)$$
$$\tilde{z}_k = h(x_k, 0)$$

This then yields the linearization,

$$x_k \approx \tilde{x}_k + A(x_{k-1} - \hat{x}_{k-1}) + W w_{k-1}$$
$$z_k \approx \tilde{z}_k + H(x_k - \tilde{x}_k) + V v_k$$

where A is the Jacobian matrix,

$$A_{[i,j]} = \frac{\partial f_{[i]}}{\partial x_{[j]}}(\hat{x}_{k-1}, u_k, 0)$$

and similarly for W, H, V with respect to w, x, v respectively.

Extended Kalman Filter - Computation

Given the linearization,

$$x_k \approx \tilde{x}_k + A(x_{k-1} - \hat{x}_{k-1}) + W w_{k-1}$$
$$z_k \approx \tilde{z}_k + H(x_k - \tilde{x}_k) + V v_k$$

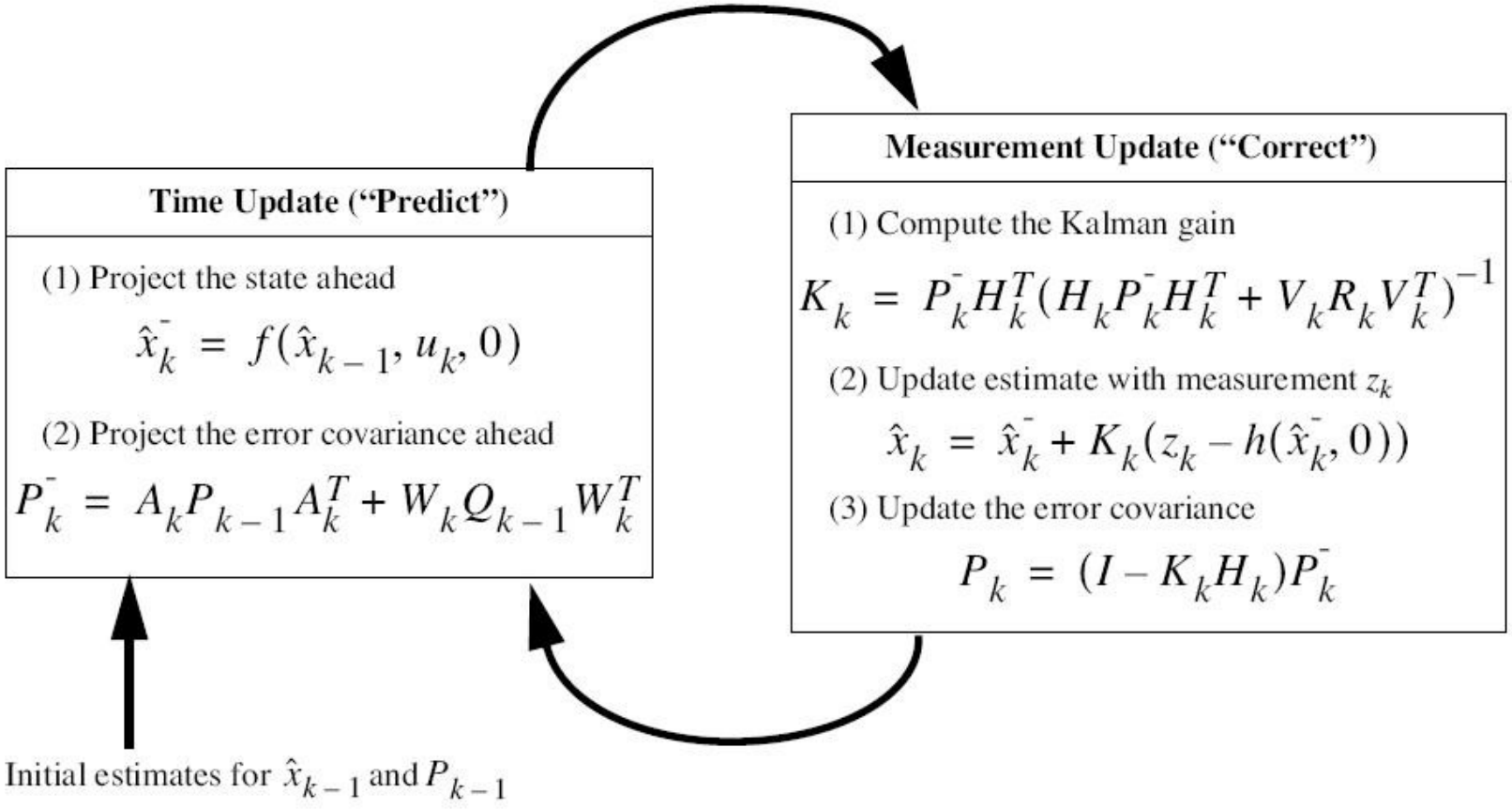
we define prediction error and measurement residual as,

$$\tilde{e}_{x_k} \equiv x_k - \tilde{x}_k \approx A(x_{k-1} - \hat{x}_{k-1}) + \epsilon_k$$
$$\tilde{e}_{z_k} \equiv z_k - \tilde{z}_k \approx H\tilde{e}_{x_k} + \eta_k$$

where ϵ_k, η_k are 0-mean random variables with covariance WQW' and VRV'.

- Notice that the above equations look a lot like the linear system dynamics and measurement equations
- So, we can define a Kalman filter over this error
- And then, use that linear error estimate to drive the main estimation loop

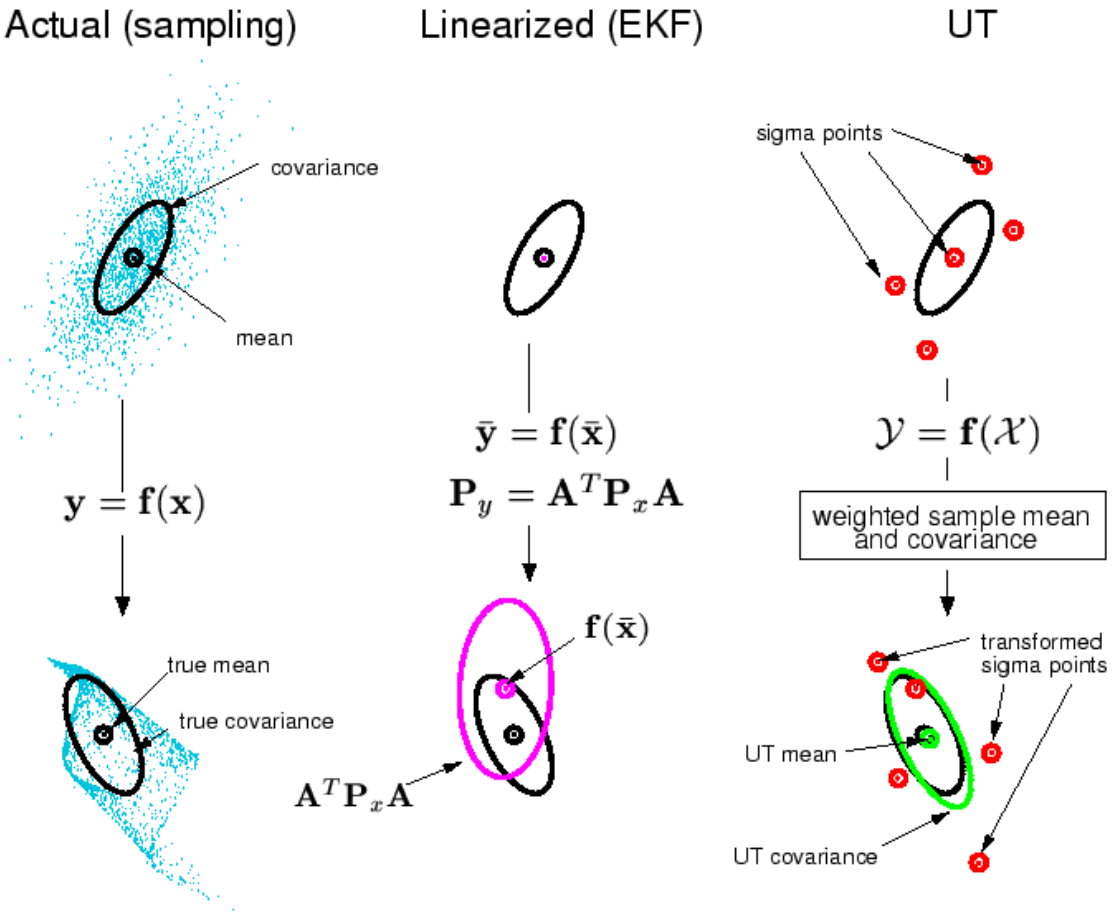
Extended Kalman Filter: The Loop



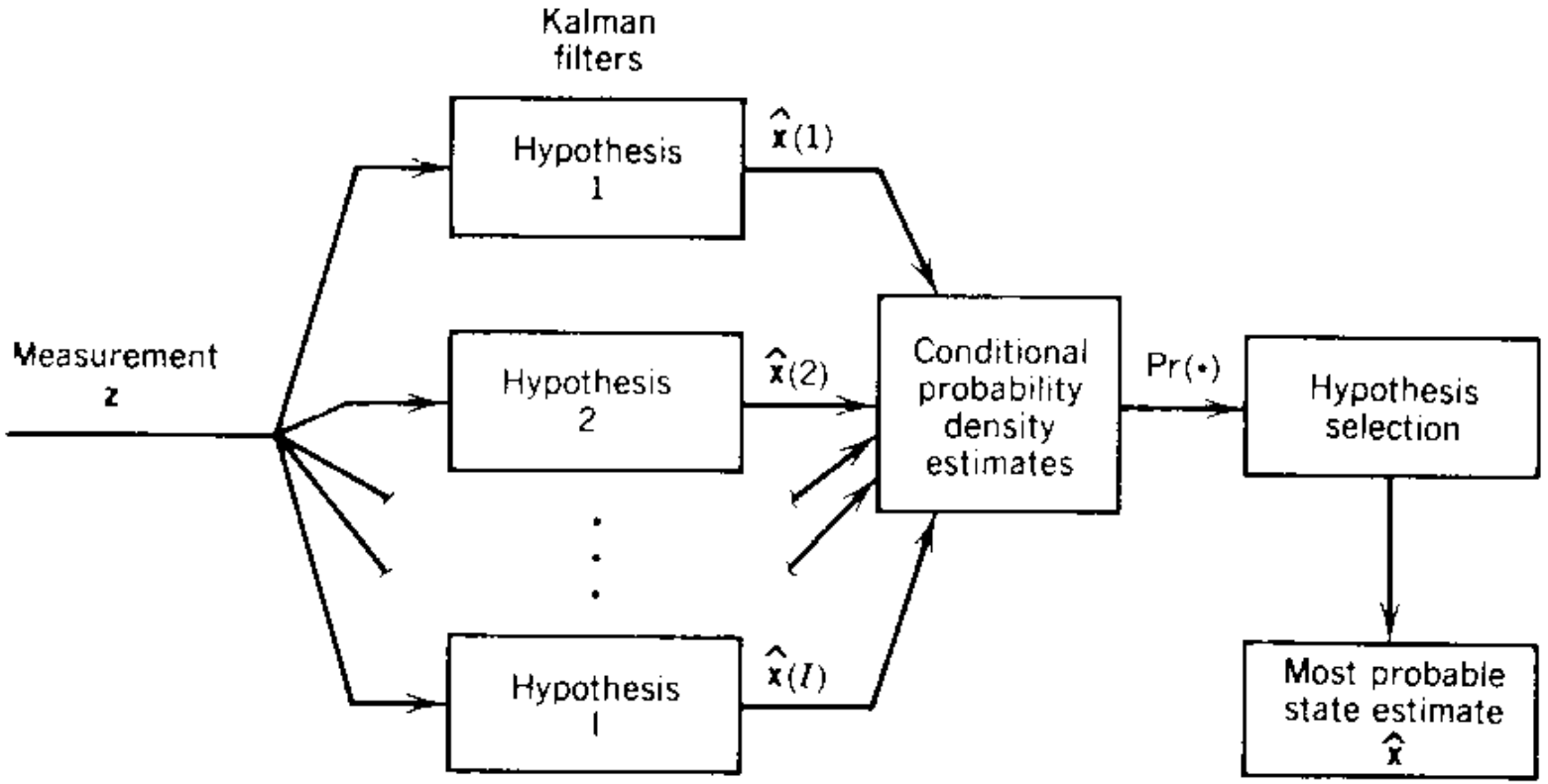
Limitations of the EKF

- If the initial state estimate is wrong, or if the process is incorrectly modeled, the filter may quickly diverge (due to linearization)
- Covariance estimate is an underestimate of the true matrix - risks becoming inconsistent in a statistical sense without the addition of "stabilizing noise“
- Computational complexity associated with Jacobian/Hessian calculation – can be an issue in high-dimensional setting

Unscented Kalman Filter: Basic Idea



Multiple Model Filters



Organization for multiple-model estimation.

Interesting Recent Hypothesis

INSTITUTE OF PHYSICS PUBLISHING JOURNAL OF NEURAL ENGINEERING
J. Neural Eng. **2** (2005) S219–S234 doi:10.1088/1741-2560/2/3/S06

Evolution of the cerebellum as a neuronal machine for Bayesian state estimation

M G Paulin

Department of Zoology and Centre for Neuroscience, University of Otago, Dunedin, New Zealand

E-mail: mike.paulin@stonebow.otago.ac.nz

Received 23 January 2005
Accepted for publication 15 July 2005
Published 31 August 2005
Online at stacks.iop.org/JNE/2/S219

Abstract
The cerebellum evolved in association with the electric sense and vestibular sense of the earliest vertebrates. Accurate information provided by these sensory systems would have been essential for precise control of orienting behavior in predation. A simple model shows that individual spikes in electrosensory primary afferent neurons can be interpreted as measurements of prey location. Using this result, I construct a computational neural model in which the spatial distribution of spikes in a secondary electrosensory map forms a Monte Carlo approximation to the Bayesian posterior distribution of prey locations given the sense data. The neural circuit that emerges naturally to perform this task resembles the cerebellar-like hindbrain electrosensory filtering circuitry of sharks and other electrosensory vertebrates. The optimal filtering mechanism can be extended to handle dynamical targets observed from a dynamical platform; that is, to construct an optimal dynamical state estimator using spiking neurons. This may provide a generic model of cerebellar computation. Vertebrate motion-sensing neurons have specific fractional-order dynamical characteristics that allow Bayesian state estimators to be implemented elegantly and efficiently, using simple

Reference:

S. Thrun et al., Probabilistic Robotics, MIT Press, 2005.

Several pictures and equations in this presentation are taken from the tutorial by Welch and Bishop (http://www.cs.unc.edu/~welch/kalman/)