Kalman Filter

Dr. Elli Angelopoulou

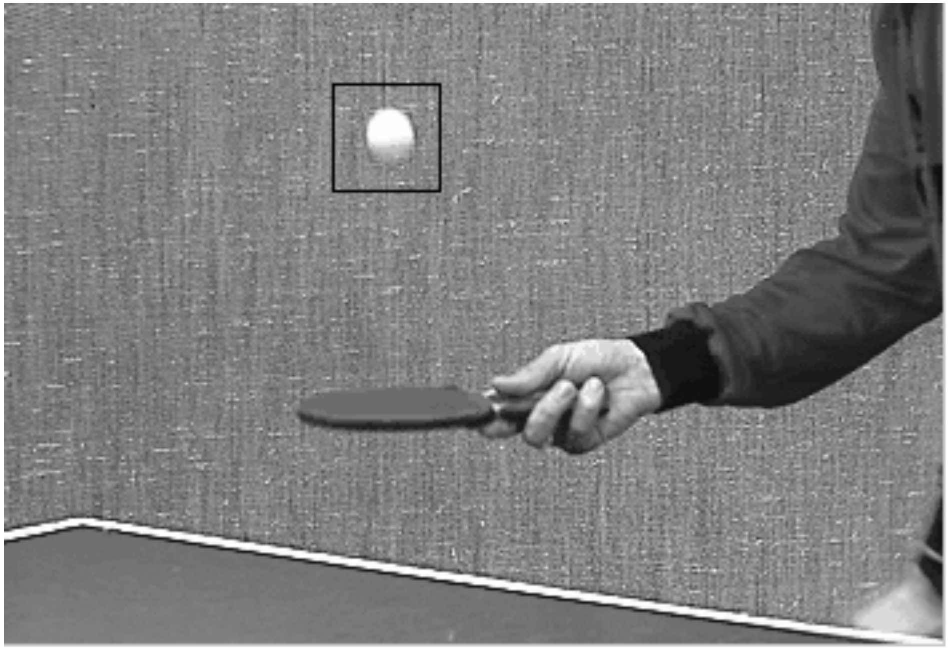
Pattern Recognition Lab (Computer Science 5)

University of Erlangen-Nuremberg

Optical Flow

The direction of the optical flow vectors is color coded as shown on this sphere.

Tracking Specific Objects



Tracking with Kalman Filter

New Paradigm - Prediction

- The image brightness equation does not explicitly incorporate previous knowledge.
- For example, based on what has been observed so far can we *predict* where the moving object will most probably be in the next frame?
- Such a method would work better:
  - If we observe the scene for more than 2 or 3 frames.
  - There are specific objects or regions whose motion is analyzed instead of estimating the motion of every pixel that has changed.

Tracking

- Tracking: the pursuit (of a person or animal) by following tracks or marks they left behind.
- Tracking in computer vision: following the motion of a particular object (or objects).
- Tracking in computer vision often involves predicting where the object(s) will appear in the next frame, based on:
  - Previous observations, up to the current frame, on how the object(s) move.
  - A model that describes how the motion of the object.

Dynamic System

- Motion is now analyzed in the context of a dynamic system.
- Typical attributes of such a system are:

1. We are dealing with a system that is **changing over time**, i.e. a dynamic system.
2. We have sensors observing the dynamic scene. The **measurements** of compute from them are **noisy**.
3. There is an **uncertainty** about how the system is changing. In other words we have an uncertain model of the system's dynamics.
4. We want to produce the best possible estimates of what is moving in which direction and at what speed. We want **optimal estimates** of the state of a dynamic system.

Optimality

- Our goal is to obtain optimal motion estimates.
- How do we know that our estimates are good approximations of what is really happening?
- Common method: Our estimates should come as close as possible to the real motion. The difference between the *true* and the *estimated* values should be as close to *zero* as possible.
- Soooo... out of all the possible solutions we want the one that minimizes the mean of the squared error (MSE).
- The idea of minimizing the mean squared error is not new. It has its roots as far back as Gauss (1795).
- R.E. Kalman introduced in 1960 an efficient recursive solution to minimizing least-squared error for discrete-data linear problems.

Rudolf Kalman

- Draper Prize by National Academy of Engineering 2008
- National Medal of Science on October 7th, 2009.

Kalman Filter

- His solution, known as *Kalman filter* is a set of mathematical equations that provides an efficient recursive solution to the least-squares method.
- It explicitly encompasses noise and uncertainty.
- Originally, Kalman filtering was designed as an **optimal Bayesian technique** to estimate **state variables** at time t based on:
  - the previous state of the variables, i.e. at time t-1
  - indirect and noisy measurements at time t
  - known statistical correlations between variables and time.
- Kalman filtering can also be used to estimate variables in a static (i.e. time-independent) system, if the system is appropriately modeled.

Kalman Filter Popularity

- Since its introduction in 1960, Kalman filtering (KF) has become a classical tool of optimal estimation theory and has been applied in areas as diverse as:
  - aerospace,
  - marine navigation,
  - nuclear power plant instrumentation,
  - demographic modeling,
  - manufacturing,
  - …
- Why did this method become so popular?
- The KF method is very powerful in several aspects:
  - it supports estimations of past, present and future states,
  - it can do so even when the precise nature of the modeled system is unknown.

Dynamic System Formulation

- We will view the problem in its more general formulation.
- Consider motion as a problem where we have to **estimate the values of the variables of some dynamic system**.
- A dynamic system is often described via:
- a **state vector** $\vec{x}$, also known as the state,
- a set of equations called the **system model**, which captures the evolution of the state vectors over time.

State Vector

- The state vector x is a time-dependent vector $\vec{x}(t) \in R^n$.
- The elements of the vector are variables of the dynamic system.

$$\vec{x}(t) = (q_1(t), q_2(t), \ldots, q_n(t))$$

- In case of motion, $\vec{x}(t) = (v_x(t), v_y(t))$.
- How big is *n*? As big as necessary in order to capture all the dynamic properties of the system.
- Example1: 3D motion $\vec{x}(t) = (v_x(t), v_y(t), v_z(t))$
- Example2: multiple moving objects, e.g. four objects moving on a plane (2D motion).

$$\begin{aligned}\vec{x}(t) &= (\vec{x}_1(t), \vec{x}_2(t), \vec{x}_3(t), \vec{x}_4(t)) \\ &= (v_{1x}(t), v_{1y}(t), v_{2x}(t), v_{2y}(t), v_{3x}(t), v_{3y}(t), v_{4x}(t), v_{4y}(t))\end{aligned}$$

Time

- Assume that we observe the system at discrete, equally spaced time intervals so that:

$$t_k = t_0 + k\delta t$$

where $k = 0,1,\ldots$ and δt is the sampling interval

- For simplicity $\vec{x}(t_k)$ is denoted as $\vec{x}_k$.
- Assumption: δt is small enough to capture the dynamics of the system. In other words, the system does not change much between consecutive time instants, i.e. during δt.

System Model

- Key Assumption: The system is **linear**. That means that the relationship between consecutive state-changes is linear.
- Then the system model can be written as:

$$\vec{x}_k = \mathbf{\Phi}_{k-1}\vec{x}_{k-1} + \vec{w}_{k-1}$$

- $\vec{w}_{k-1}$ is a vector describing the **random process noise**.
- $\mathbf{\Phi}_{k-1}$ is the **state transition matrix** that captures the relationship between the current state *k* and the previous state *k*-1 in the absence of noise.
- $\mathbf{\Phi}_{k-1}$ is an *n* x *n* matrix, $\vec{w}_{k-1}$ is an *n*-dimensional vector.

- The formulation so far does not consider the fact that we can have observations of the system.

Measurements

- At any time t_k, we have a vector $\vec{z}_k \in R^m$ of measurements of the system.
- Due to imperfections (e.g. noise) in our sensors, there is uncertainty in our measurements.
- The vector $\vec{\mu}_k$ describes the uncertainty associated with each measurement $\vec{z}_k$.
- The relationship between the true system state $\vec{x}_k$ and our measurements is given by the following equation:

$$\vec{z}_k = \mathbf{H}_k \vec{x}_k + \vec{\mu}_k$$

- $\mathbf{H}_k$ is the **measurement matrix** that captures the relationship between our measurements and the real system variables in the absence of noise. It is an *m* x *n* matrix.
- $\vec{\mu}_k$ is an *m*-dimensional vector known as the **measurement noise**.

Noise

- There are two types of noise:
  - Process noise $\vec{w}_k$
  - Measurement noise $\vec{\mu}_k$
- In Kalman filtering both types of noise are assumed to be white, zero-mean Gaussians.
- As such they are described by their corresponding covariance matrices:
  - Process noise covariance $\mathbf{Q}_k$
  - Measurement noise covariance $\mathbf{R}_k$

Notations

- State variable $\vec{x}_k$
- State transition matrix $\mathbf{\Phi}_k$
- Process noise $\vec{w}_k$
- Process noise covariance $\mathbf{Q}_k$

- Measurement $\vec{z}_k$
- Measurement matrix $\mathbf{H}_k$
- Measurement noise $\vec{\mu}_k$
- Measurement noise covariance $\mathbf{R}_k$

Kalman Filter Setup

- We are observing a dynamic system.
- We have a linear system model, but there is uncertainty about the accuracy of the employed model.

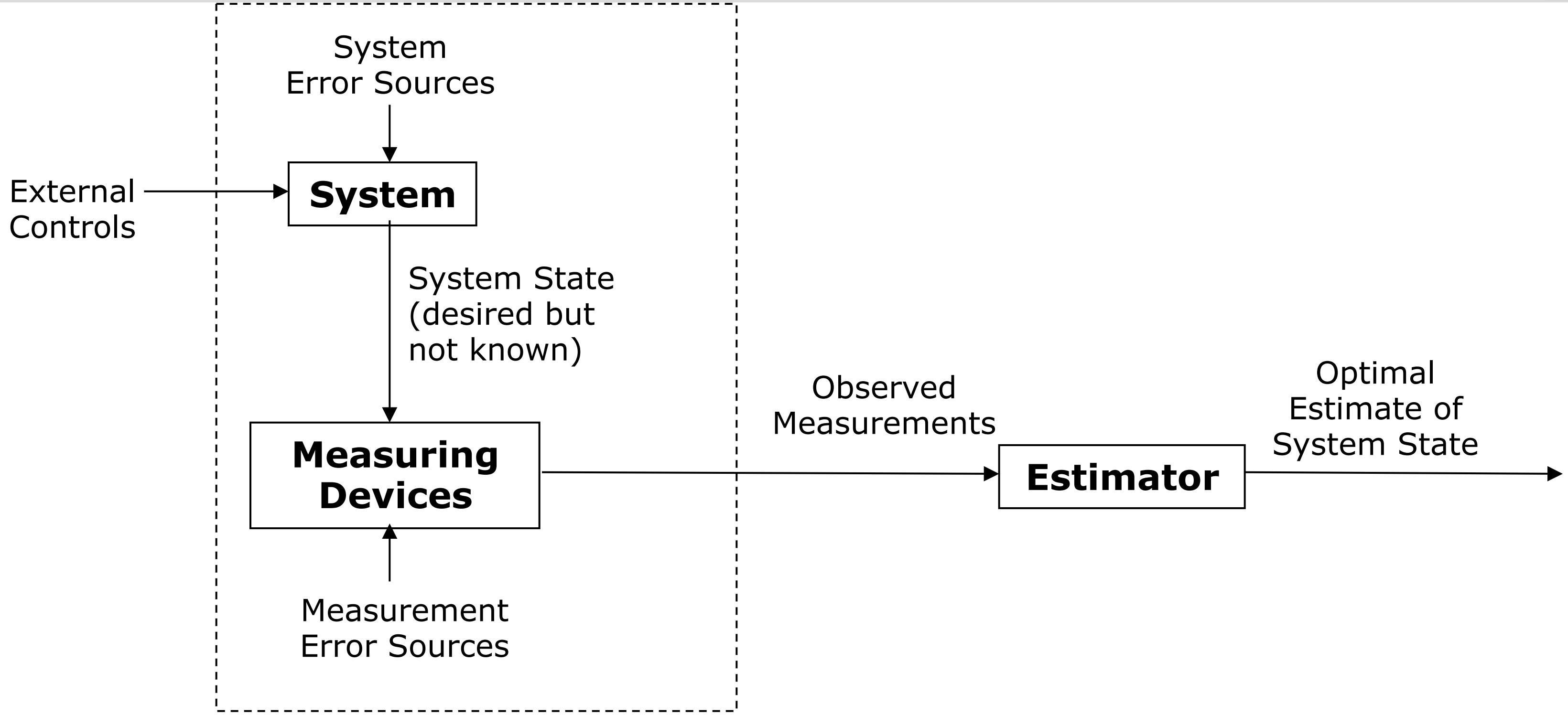
$$\vec{x}_k = \mathbf{\Phi}_{k-1}\vec{x}_{k-1} + \vec{w}_{k-1}$$

- We also have sensor(s) that measure how the dynamic system behaves.

$$\vec{z}_k = \mathbf{H}_k\vec{x}_k + \vec{\mu}_k$$

- The sensor(s) are noisy.
- The sensor noise is assumed to follow a white, zero-mean, Gaussian distribution.

The Problem



- So far we have setup our variables and equations to describe a linear dynamic system that is measured by some sensors.
- Goal: Compute the best estimate of the system state $\hat{\vec{x}}_k$ at time t_k given the previous state estimate $\hat{\vec{x}}_{k-1}$ and the current measurements $\vec{z}_k$.

Kalman Filter Setup

- We are observing a dynamic system.
- We have a linear system model, but there is uncertainty about the accuracy of the employed model.

$$\vec{x}_k = \mathbf{\Phi}_{k-1}\vec{x}_{k-1} + \vec{w}_{k-1}$$

- We also have sensor(s) that measure how the dynamic system behaves.

$$\vec{z}_k = \mathbf{H}_k\vec{x}_k + \vec{\mu}_k$$

- The sensor(s) are noisy.
- The sensor noise is assumed to follow a white, zero-mean, Gaussian distribution.

KF Idea

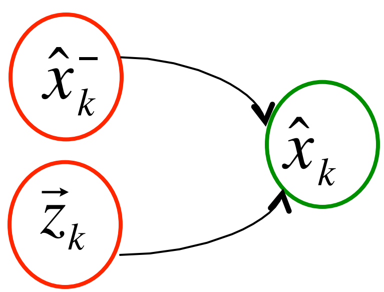
- An estimate of $\hat{\vec{x}}_k$ $(\hat{x}_k)$ is obtained from $\hat{\vec{x}}_{k-1}$ $(\hat{x}_{k-1})$ and $\vec{z}_k$ in a 2-step process:

1. First, obtain an intermediate estimate, $\hat{x}_k^-$, based on the previous estimates, but *without* using the newest measurements $\vec{z}_k$.

$$\hat{x}_k^- = \mathbf{\Phi}_{k-1}\hat{x}_{k-1}$$

- It is called the **prediction step**. It predicts what the state variable should be based purely on our model.

2. Use the intermediate estimate $\hat{x}_k^-$ and combine it, in the **update step**, with the newest measurements $\vec{z}_k$, to get $\hat{x}_k$.



KF Idea - continued

- This 2-step process is performed as a series of 4 (or 5) recursive equations.
- The 4 (or 5) Kalman Filter equations are characterized by:

1. The state covariance matrix $\mathbf{P}_k$. It is the covariance matrix of the estimate $\hat{x}_k$. It is also known as the **covariance of the estimates**. It is a measurement of the uncertainty in $\hat{x}_k$.
2. The state covariance matrix $\mathbf{P}_k^-$. It is the covariance matrix of the estimate $\hat{x}_k^-$. It is also known as the **covariance of the prediction error**. It is a measurement of the uncertainty in $\hat{x}_k^-$.
3. The **gain matrix** $\mathbf{K}_k$. It expresses the relative importance of the prediction $\hat{x}_k^-$ and the measurement $\vec{z}_k$.

Notations for KF equations

- State variable $\vec{x}_k$
- State transition matrix $\mathbf{\Phi}_k$
- Process noise $\vec{w}_k$
- Process noise covariance $\mathbf{Q}_k$
- Covariance of the estimates $\mathbf{P}_k$
- Covariance of the prediction $\mathbf{P}_k^-$
- Gain Matrix $\mathbf{K}_k$
- Measurement $\vec{z}_k$
- Measurement matrix $\mathbf{H}_k$
- Measurement noise $\vec{\mu}_k$
- Measurement noise covariance $\mathbf{R}_k$

Kalman Filter

- Prediction equations

$$\hat{x}_k^- = \mathbf{\Phi}_{k-1}\hat{x}_{k-1}$$

$$\mathbf{P}_k^- = \mathbf{\Phi}_{k-1}\mathbf{P}_{k-1}\mathbf{\Phi}_{k-1}^T + \mathbf{Q}_k$$

  - Project state and covariance estimates forward in time

- Update equations

$$\mathbf{K}_k = \mathbf{P}_k^-\mathbf{H}_k^T(\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T + \mathbf{R}_k)^{-1}$$

$$\hat{x}_k = \hat{x}_k^- + \mathbf{K}_k(\vec{z}_k - \mathbf{H}_k\hat{x}_k^-)$$

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^-(\mathbf{I} - \mathbf{K}_k)^T + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T$$

  - Compute Kalman gain K
  - Include the measurement
  - Compute a posteriori estimate
  - Compute a posteriori covariance of the estimate

Kalman Filter Equations

Predict

Correct

$$\mathbf{P}_k^- = \mathbf{\Phi}_{k-1}\mathbf{P}_{k-1}\mathbf{\Phi}_{k-1}^T + \mathbf{Q}_k$$

$$\mathbf{K}_k = \mathbf{P}_k^-\mathbf{H}_k^T(\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T + \mathbf{R}_k)^{-1}$$

$$\hat{x}_k = \mathbf{\Phi}_{k-1}\hat{x}_{k-1} + \mathbf{K}_k(\vec{z}_k - \mathbf{H}_k\mathbf{\Phi}_{k-1}\hat{x}_{k-1})$$

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^-(\mathbf{I} - \mathbf{K}_k)^T + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T$$

$\mathbf{\Phi}_{k-1}\hat{x}_{k-1}$ is the prediction

$\vec{z}_k - \mathbf{H}_k\mathbf{\Phi}_{k-1}\hat{x}_{k-1}$ is the innovation

$\hat{x}_k = \mathbf{\Phi}_{k-1}\hat{x}_{k-1} + \mathbf{K}_k(\vec{z}_k - \mathbf{H}_k\mathbf{\Phi}_{k-1}\hat{x}_{k-1})$ is the update

KF Remarks

- Let’s take a closer look at the computation of the gain matrix and the update equation:

$$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{H}_k^T (\mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T + \mathbf{R}_k)^{-1}$$

$$\hat{x}_k = \hat{x}_k^- + \mathbf{K}_k (\vec{z}_k - \mathbf{H}_k \hat{x}_k^-)$$

- If the measurement noise is much greater than the process noise,

$$\mathbf{R}_k >> \mathbf{Q}_k$$

$\mathbf{K}_k$ will be small (that is, we won't give much credence to the measurement).

- If the measurement noise is much smaller than the process noise,

$$\mathbf{R}_k << \mathbf{Q}_k$$

$\mathbf{K}_k$ will be large (that is, we don’t trust our model too much).

KF Remarks - continued

- The method assumes initial estimates of $\mathbf{P}_0$ and $\hat{x}_0$.
- Typically, the entries in $\mathbf{P}_0$ are set to arbitrary high values. We set $\mathbf{P}_0$ to arbitrarily high values because we don't trust our initial estimates. Hence, the estimate error is expected to be high.
- For $\hat{x}_0$, if we have some data, we use it, otherwise we set that, too, to arbitrary values.

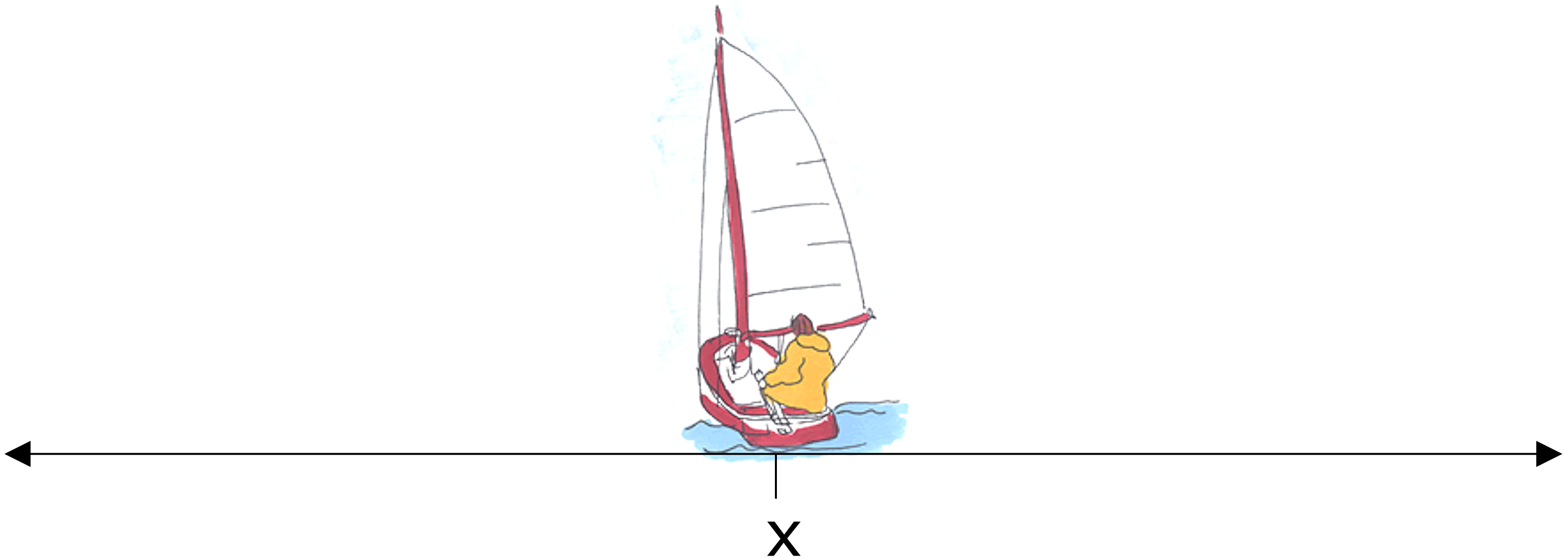
Filter Parameters and Tuning

- Most of the times we assume stable R_k and Q_k over time.
- R: measurement noise covariance can be measured a priori. If we know our sensor we can analyze its noise behavior. Similarly, we can estimate the accuracy of our algorithm that extracts the measurement from the sensed data.
- Q: process noise covariance. Can not be measured, because we can not directly observe the process we are measuring. If we choose Q large enough (lots of uncertainty), a poor process model can still produce acceptable results.
- Parameter tuning: We can increase filter performance by tuning the parameters R and Q.

Filter Parameters and Tuning

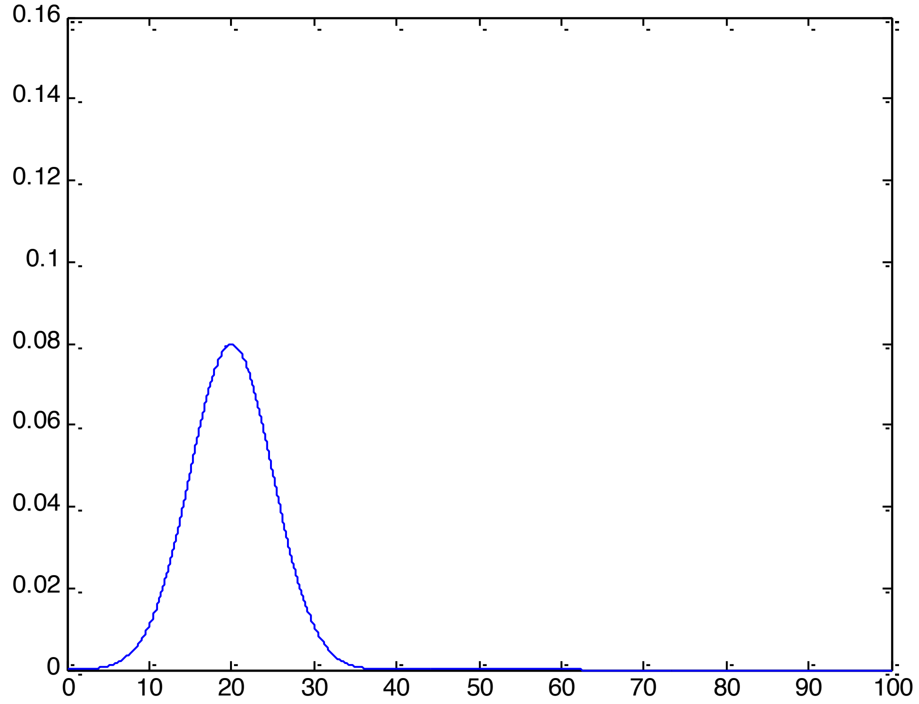
- If we measure directly what we are trying to predict, then we can set H to the identity matrix I.
- If R and Q are constant, the estimation error covariance P_k and the Kalman gain K_k will stabilize quickly and stay constant. In this case, P_k and K_k can be precomputed.

Conceptual Overview



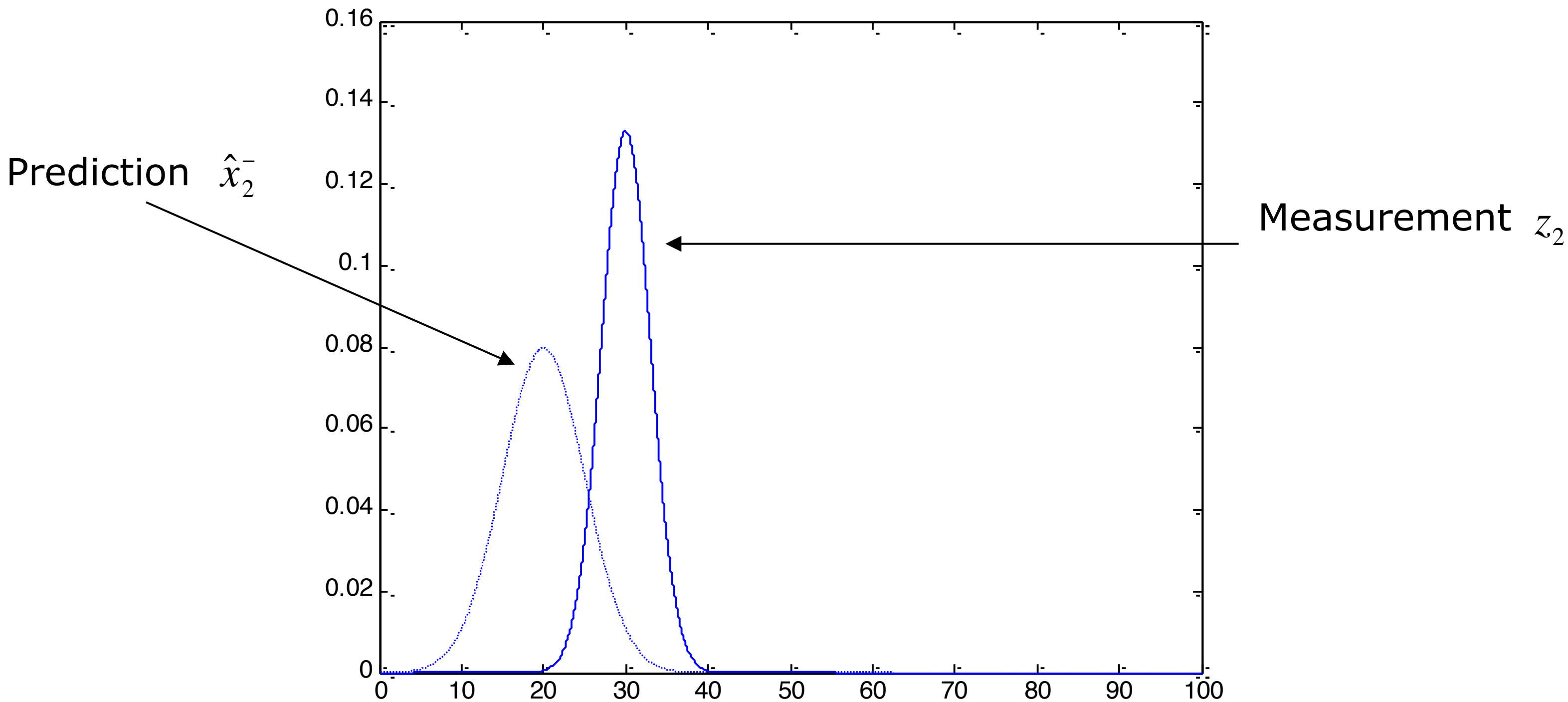
- Lost on the 1-dimensional line
- Position – x(t)
- Assume Gaussian distributed measurements

Conceptual Overview



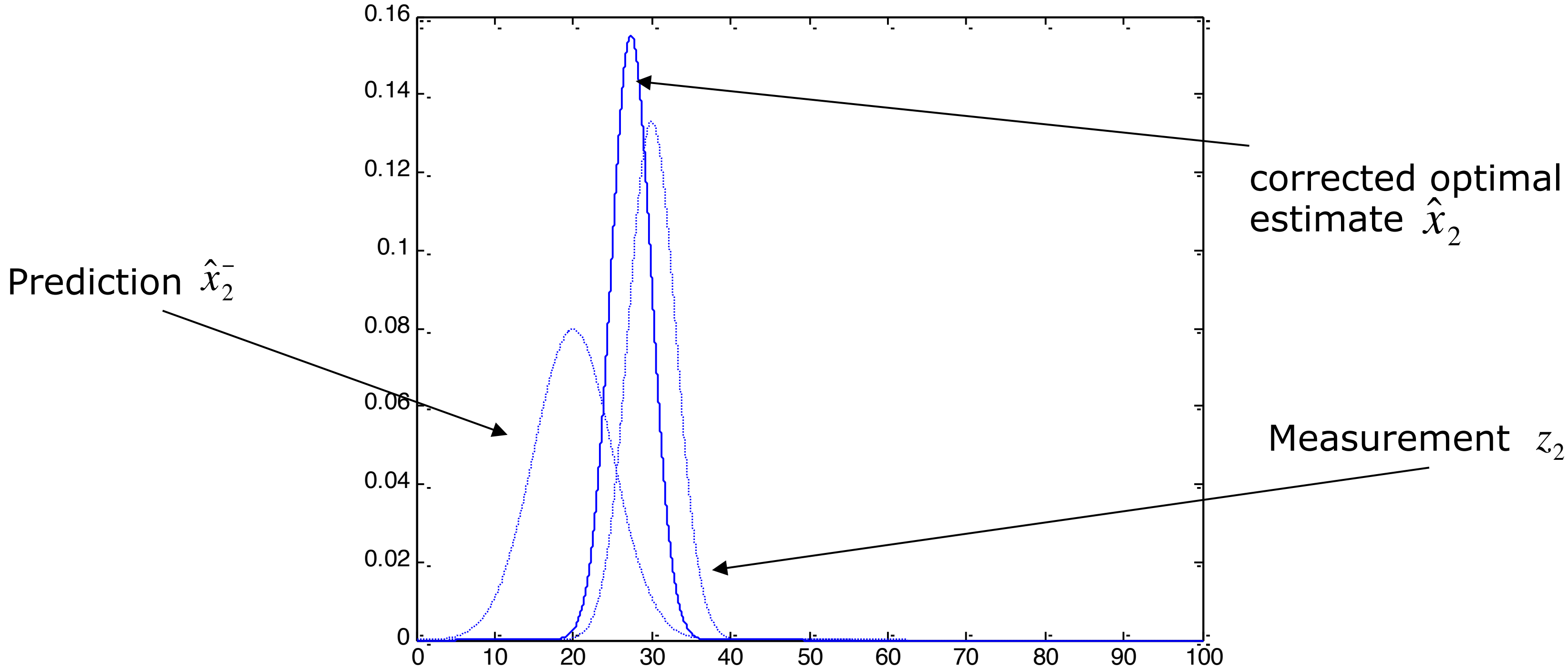
- GPS (or sextant) measurement at t_1: Mean = z_1 and Variance = σ_{z1}
- Optimal estimate of position is: $\hat{x}(t_1) = z_1$
- Variance of error in estimate: $\sigma^2_x(t_1) = \sigma^2_{z1}$
- If the boat stays in the same position at time t_2, then then the <u>Predicted</u> position is $\hat{x}_2^- = z_1$

Conceptual Overview



- So we have the prediction $\hat{x}_2^-$
- GPS Measurement at t_2: Mean = z_2 and Variance = σ_{z2}
- Need to correct the prediction due to measurement to get $\hat{x}_2$
- Closer to more trusted measurement – linear interpolation?

Conceptual Overview



- The corrected mean is the new optimal estimate of position $\hat{x}_2$
- The variance of the new estimate is smaller than either of the previous two variances.

Conceptual Overview

- So far:

We made a prediction based on previous data: $\hat{x}_k^-$, σ^-

↓

Took a measurement: z_k, σ_z

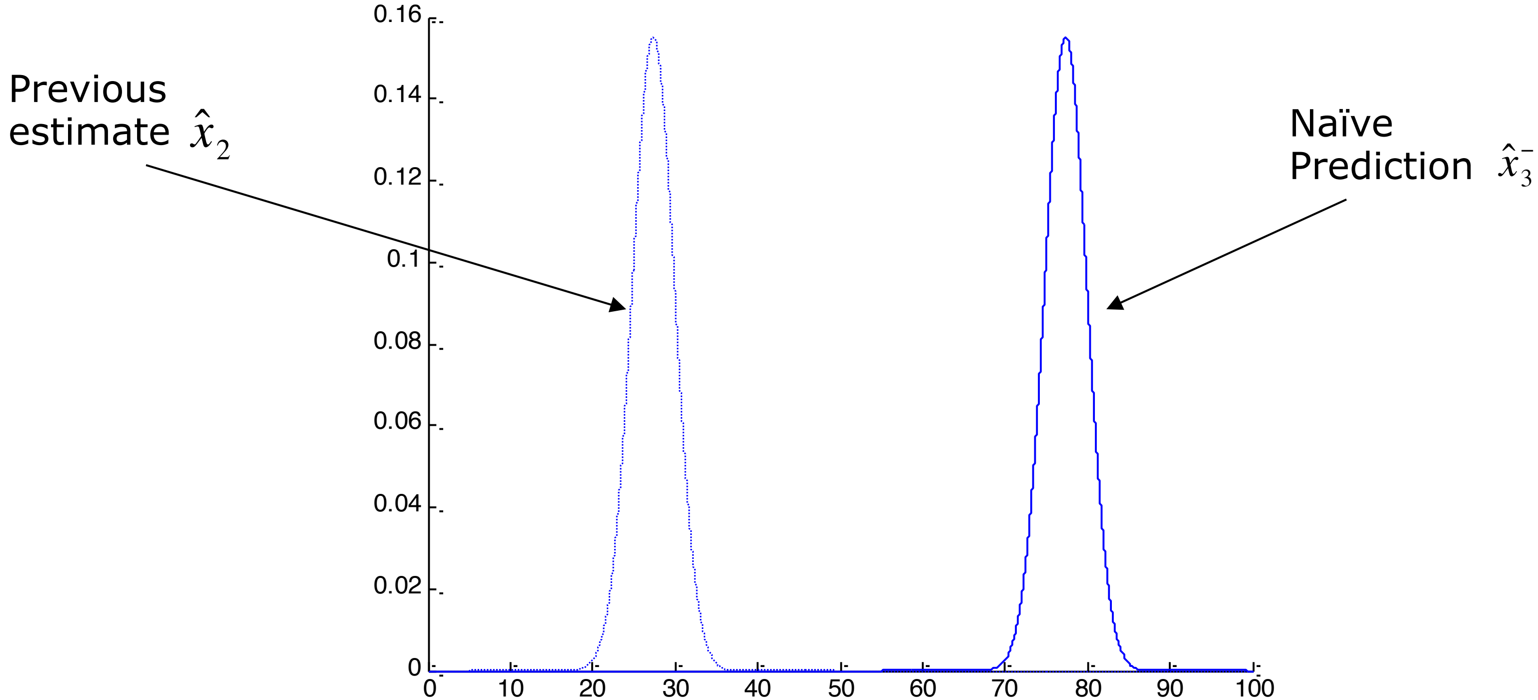
↓

Combined our prediction and our measurement to get a new optimal estimate and its variance

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

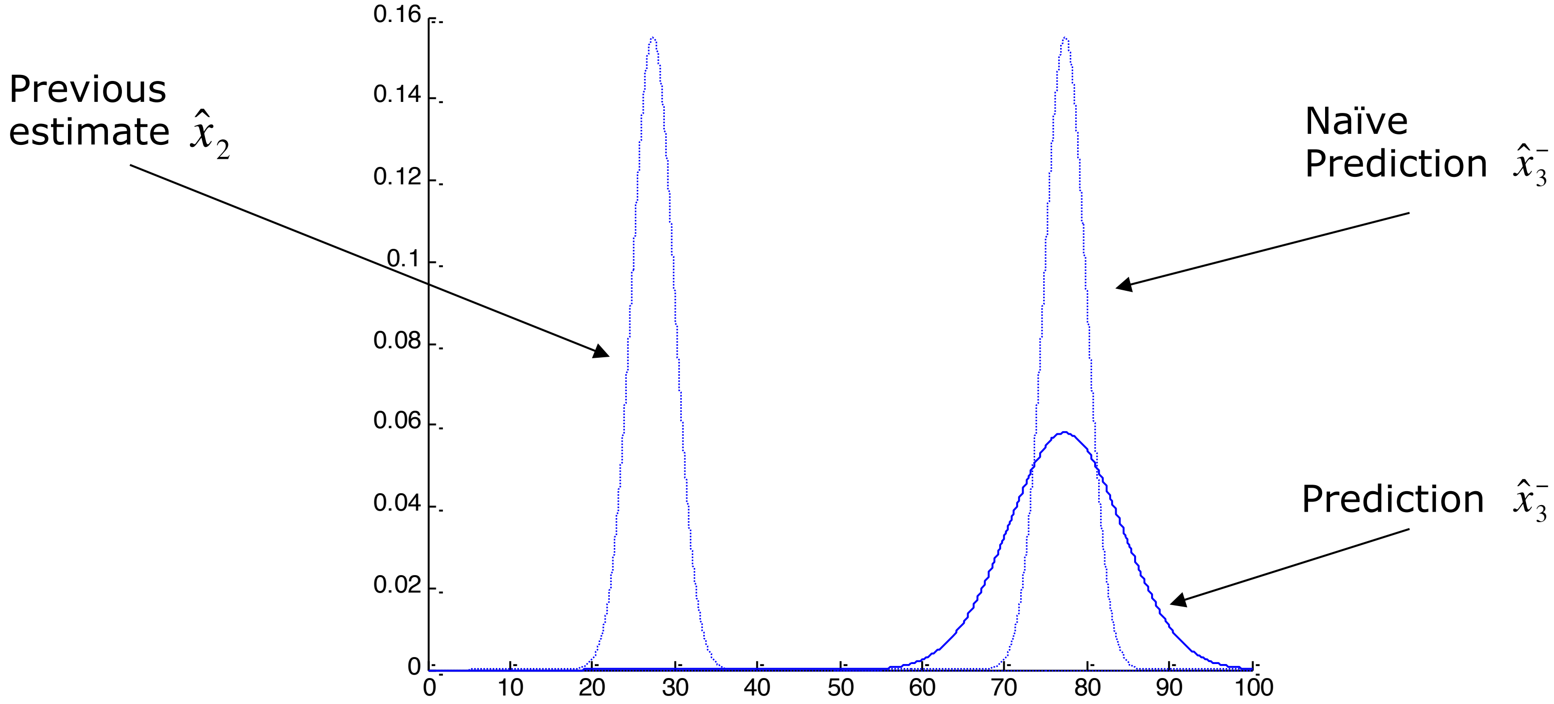
$$\sigma_k = \sigma^-(1-K) + K\sigma_z$$

Conceptual Overview



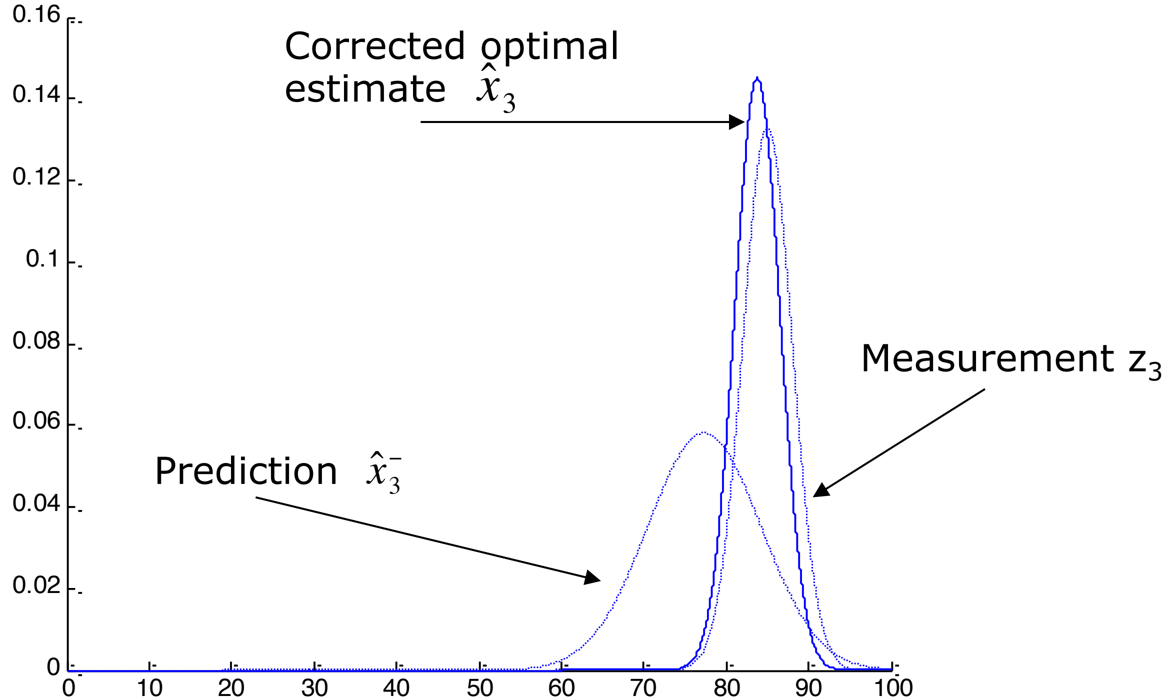
- At time t_3, boat moves with velocity v=dx/dt
- Naïve approach: Shift probability to the right, according to the speed of the boat, to predict its position.
- This would work *if* we knew the velocity exactly, i.e. we had a perfect model.

Conceptual Overview



- Better to assume imperfect model by adding Gaussian noise.
- v= dx/dt +/- w
- The distribution for prediction not only moves according to the speed of the boat but also spreads out.

Conceptual Overview



- Take another GPS (sextant) measurement at t_3: Mean = z_3 and Variance = σ_{z3}
- Correct the prediction by linearly interpolating the pure prediction with the measurement.

Conceptual Overview

So what have we done?

We made a prediction based on previous data:

$$\hat{x}_k^- = \mathbf{\Phi}_{k-1}\hat{x}_{k-1}$$

$$\mathbf{P}_k^- = \mathbf{\Phi}_{k-1}\mathbf{P}_{k-1}\mathbf{\Phi}_{k-1}^T + \mathbf{Q}_k$$

Took a measurement: $\vec{z}_k, \mathbf{R}_k$

Combined our prediction and our measurement to get a new optimal estimate and its variance:

$$\mathbf{K}_k = \mathbf{P}_k^-\mathbf{H}_k^T(\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T + \mathbf{R}_k)^{-1}$$

$$\hat{x}_k = \hat{x}_k^- + \mathbf{K}_k(\vec{z}_k - \mathbf{H}_k\hat{x}_k^-)$$

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^-(\mathbf{I} - \mathbf{K}_k)^T + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T$$

Optimality of Kalman Filter

- It can be proven that for a linear system under white zero-mean Gaussian noise, Kalman filtering gives an optimal solution. (Optimal in the statistical sense, i.e. the most probable estimate.)
- Even if the noise is not Gaussian, KF provably is the best linear unbiased filter.
- A Kalman filter computes the optimal $\hat{x}_k$ state estimate, as the maximum probability density of x_k given the past estimates, the past measurements and the current measurement.

$$\hat{x}_k = \max_{\vec{x}_k} p(\vec{x}_k | \vec{x}_1, \vec{x}_2, \ldots, \vec{x}_{k-1}, \vec{z}_1, \vec{z}_2, \ldots, \vec{z}_{k-1}, \vec{z}_k)$$

Optimality of Kalman Filter - continued

- The probability density function is assumed to be Gaussian so its max. coincides with its mean.

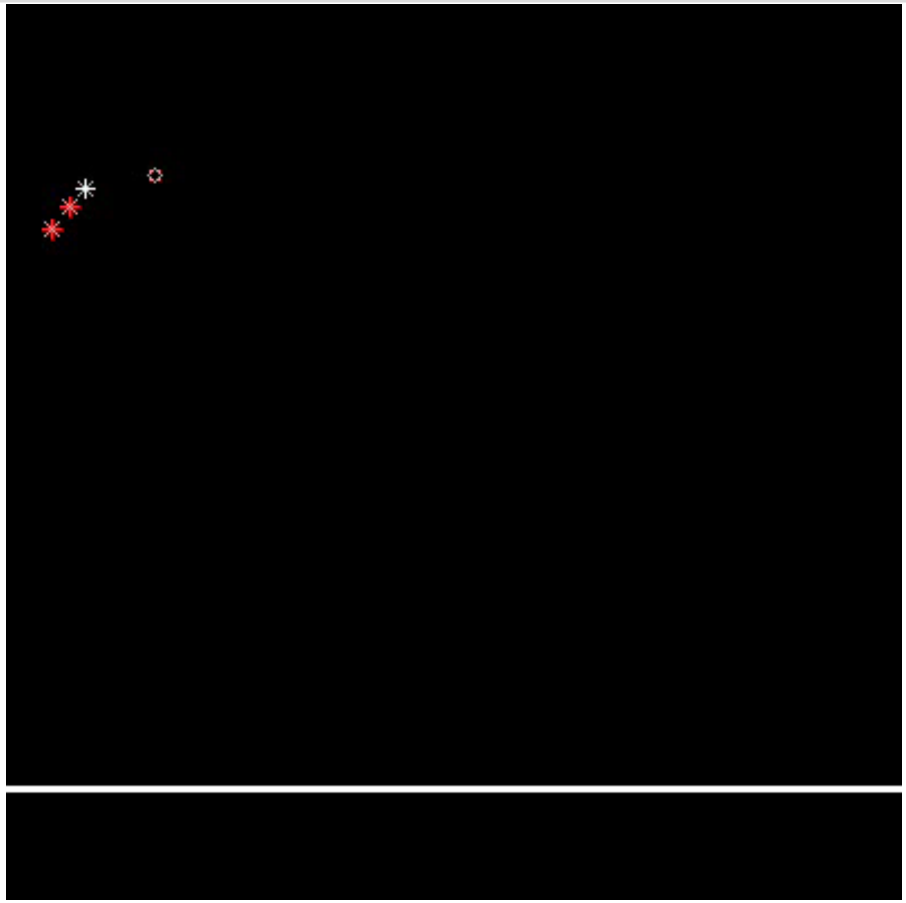
$$p(\vec{x}_k | \vec{x}_1, \vec{x}_2, \ldots, \vec{x}_{k-1}, \vec{z}_1, \vec{z}_2, \ldots, \vec{z}_{k-1}, \vec{z}_k) \sim \mathcal{N}(\vec{x}_k, \mathbf{P}_k)$$

- In reality, the true state lies with a probability c^2 within an ellipse centered at $\hat{x}_k$, where the ellipse is given by

$$(x_k - \hat{x}_k)\mathbf{P}_k^{-1}(x_k - \hat{x}_k)^T \leq c^2$$

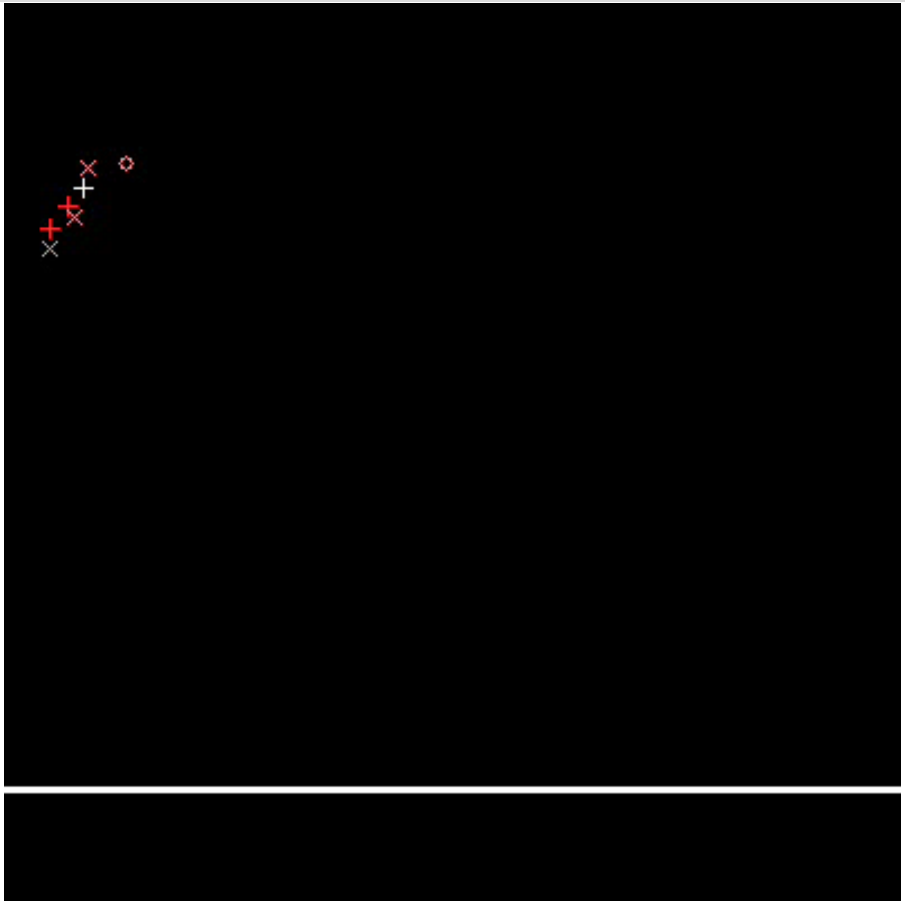
- The axes of the ellipse are the eigenvectors of $\mathbf{P}_k$.
- The true state lies with probability c^2 inside the covariance ellipse of $\hat{x}_k$.
- In tracking features we use the uncertainty ellipses to reduce the search space for locating a feature in the next frame.

Tracking Example – no Noise



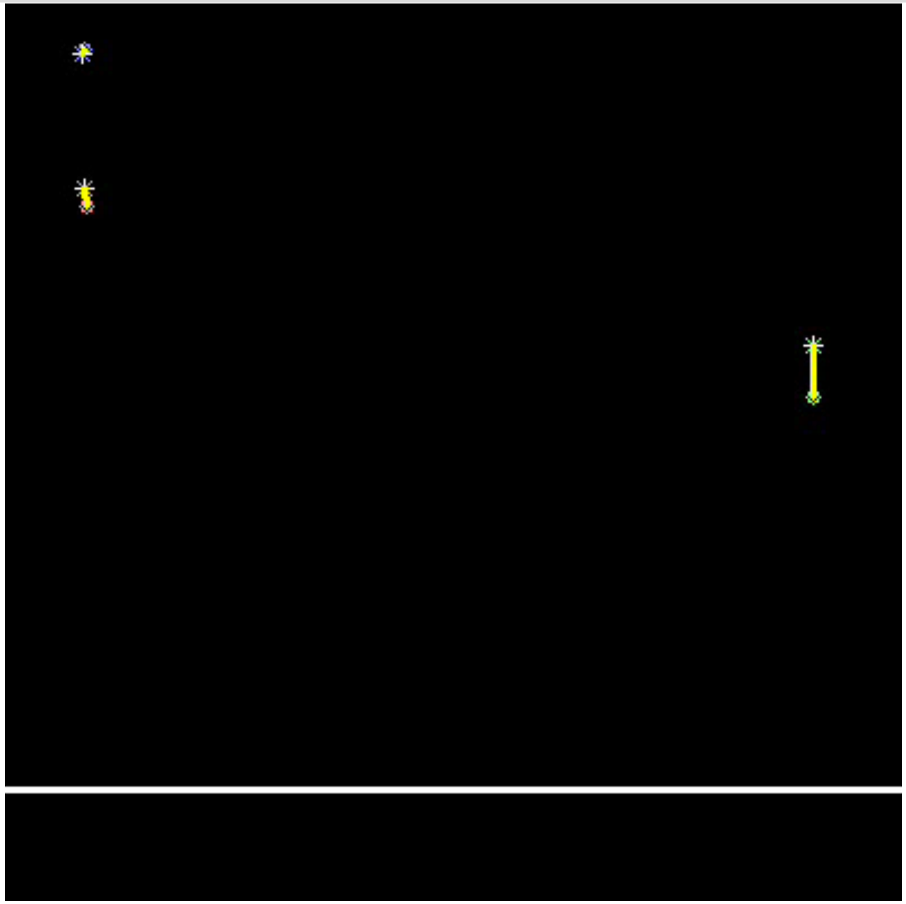
- Synthetic data without any added noise.
- True ball position shown with star. The estimated position shown with circles.
- Notice the estimate overshoots the "floor" and then overcompensates before settling down.

Tracking Example – Added Noise



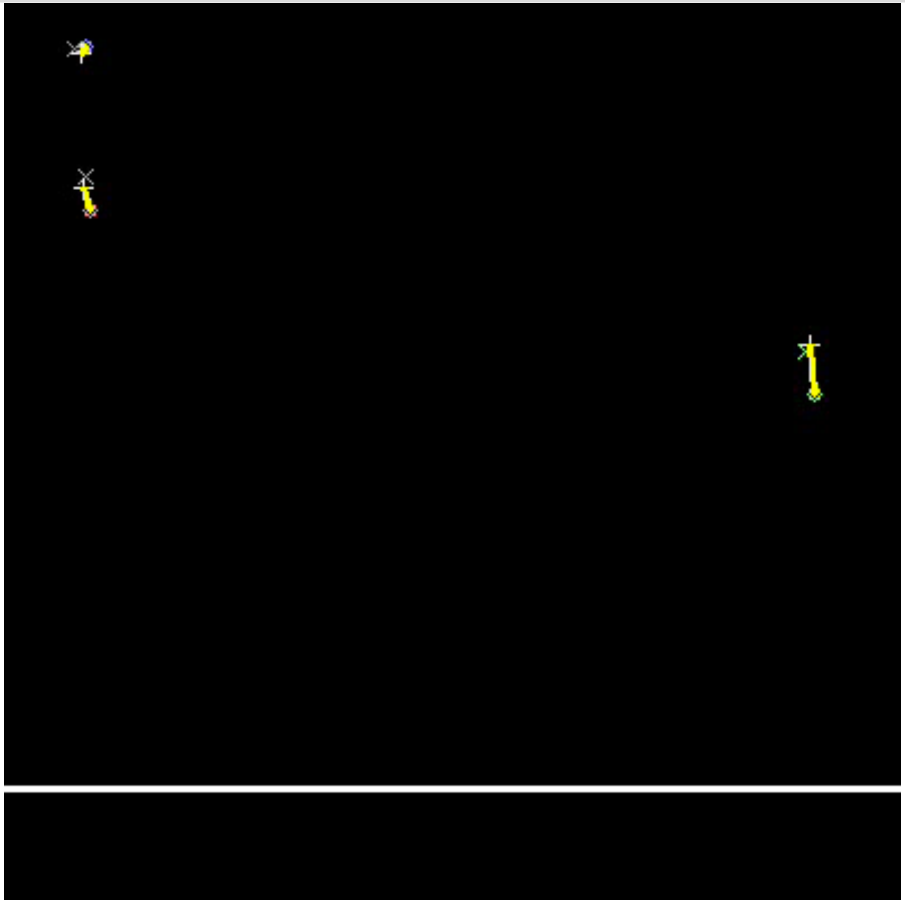
- Synthetic data with and without any added noise (Added 10% noise).
- Ideal ball position in +. Noisy ball data in x. Estimated position in o.
- The overshoot is still present. At the more linear parts of the motion KF compensates for the presence of noise.

Multiple Tracking Example – No Noise



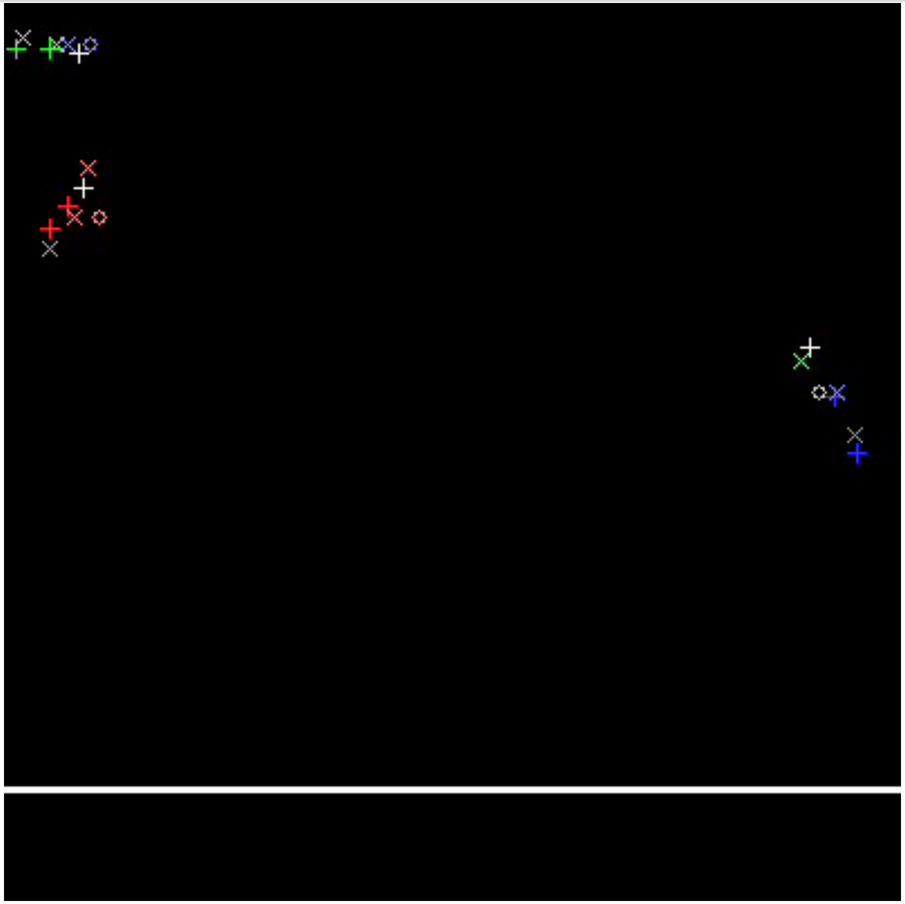
- Synthetic data without any added noise.
- Ideal ball position in +. Estimated position in o.
- Two filters end up getting associated with one set of measurements leaving another set abandoned.

Multiple Tracking Example – Little Noise



- Synthetic data with added noise of a factor of 5.
- Ideal ball position in +. Noisy ball data in x. Estimated position in o.
- The tracking still works best on the more linear parts of the motion.

Multiple Tracking Example – More Noise



- Synthetic data with added noise of a factor of 5.
- Ideal ball position in +. Noisy ball data in x. Estimated position in o.
- Notice that a different ball gets abandoned.

Challenges of Kalman Filter

- We have assumed that the system is linear. What if it is nonlinear?
- What if the measurement noise and process noise are:
  - not Gaussian,
  - not zero-mean,
  - not independent of each other?
- What if the statistics (for example, the covariance matrix) of the noise is not known?
- Matrix calculations can impose a large computational burden for high-dimensional systems. Is there a way to approximate the Kalman filter for large systems, in order to reduce the computational load while still approaching the theoretical optimum of the Kalman filter?

Kalman Filter: Good or Bad?

- Kalman Filtering is highly efficient. It has a polynomial time complexity, $O(m^{2.376} + n^2)$, where $n=dim(\mathbf{x})$ and $m=dim(\mathbf{z})$.
- It is optimal for linear Gaussian systems.
- Many systems exhibit Gaussian noise. It is a widely-used assumption.
- Most robotic systems and human motion are non-linear.

Extended Kalman Filter

- Suppose the state-evolution and measurement equations are non-linear but still differentiable:

$$\hat{x}_k = f(\hat{x}_{k-1}) + \vec{w}_{k-1}$$

$$\vec{z}_k = h(\hat{x}_k) + \vec{\mu}_k$$

  - The process noise **w** follows a zero-mean Gaussian distribution with covariance matrix **Q**.
  - The measurement noise μ follows a zero-mean Gaussian distribution with covariance matrix **R**.
- Function *f* can be used to compute the predicted state from the previous estimate.
- Function *h* can be used to compute the predicted measurement from the predicted state.
- However, *f* and *h* can *not* be directly applied on the covariance. We need a linear approximation of *f* and *h* which we get through the *Jacobian* matrix.

Jacobian Matrix

- For a scalar function $y=f(x)$,

$$\Delta y = f'(x)\Delta x$$

- For a vector function $\mathbf{y}=f(\mathbf{x})$,

$$\Delta \mathbf{y} = \mathbf{J}\Delta \mathbf{x} = \begin{bmatrix} \Delta y_1 \\ \vdots \\ \Delta y_n \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1}(\mathbf{x}) & \cdots & \frac{\partial f_1}{\partial x_n}(\mathbf{x}) \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1}(\mathbf{x}) & \cdots & \frac{\partial f_n}{\partial x_n}(\mathbf{x}) \end{bmatrix} \cdot \begin{bmatrix} \Delta x_1 \\ \vdots \\ \Delta x_n \end{bmatrix}$$

Linearize using the Jacobian

- Let $\mathbf{\Phi}$ be the Jacobian of *f* with respect to **x**.

$$\mathbf{\Phi}_{ij} = \frac{\partial f_i}{\partial x_j}(\mathbf{x}_{k-1})$$

- Let **H** be the Jacobian of *h* with respect to **x**.

$$\mathbf{H}_{ij} = \frac{\partial h_i}{\partial x_j}(\mathbf{x}_k)$$

- Then the Kalman Filter equations are almost the same as before.

EKF Equations

- Predictor step: $\hat{x}_k^- = f(\hat{x}_{k-1})$

$$\mathbf{P}_k^- = \mathbf{\Phi}_{k-1}\mathbf{P}_{k-1}\mathbf{\Phi}_{k-1}^T + \mathbf{Q}_k$$

- Kalman gain: $\mathbf{K}_k = \mathbf{P}_k^-\mathbf{H}_k^T(\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T + \mathbf{R}_k)^{-1}$

- Corrector step: $\hat{x}_k = \hat{x}_k^- + \mathbf{K}_k(\vec{z}_k - h(\hat{x}_k^-))$

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^-(\mathbf{I} - \mathbf{K}_k)^T + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T$$

Remarks on EKF

- It is still highly efficient. Similar time complexity as Kalman Filter.
- EKF does not recover optimal estimates.
- May not converge if the system is significantly non-linear.
- Computing the Jacobian can be complex.
- Still works well, even when the assumptions are violated.
- Next version for handling non-linearities: Unscented Kalman Filter.

Unscented Kalman Filtering

- EKF uses the 1st term of the Taylor series expansion.
- UKF uses the 1st two terms of the Taylor series expansion.
- UKF bases its computations on a subset of points. It uses a deterministic sampling technique known as the unscented transform to pick a minimal set of sample points (called *sigma points*) around the mean.
- The sigma points are propagated through non-linear functions and are used to obtain the mean and covariance of the estimate.
- UKF uses no Jacobians.
- It is still non-optimal.

More Kalman Filter Challenges

- What if, rather minimizing the "average" estimation error, we desire to minimize the "worst case" estimation error? This is known as the minimax or H-infinity estimation problem.

- What if, rather than estimating the state of a system as measurements are made, we already have all the measurements and we want to reconstruct a time history of the state? Can we do better than a Kalman filter? It would seem that we could since we have more information available (that is, we have future measurements) to estimate the state at a given time. This is called the *smoothing problem*.

Image Sources

1. The optical flow demo is courtesy of T. Brox http://www.cs.berkeley.edu/~brox/videos/index.html
2. The tracking ball movies are courtesy of T. Petrie http://www.marcad.com/cs584/Tracking.html
3. The person tracking example is courtesy of TUM http://www.mmk.ei.tum.de/demo/tracking/track3.gif
4. The conceptual overview slides were adapted from the presentation of M. Williams, http://users.cecs.anu.edu.au/~hartley/Vision-Reading-Course/Kalman-filters.ppt
5. The layout of a few slides was inspired by the slides of D. Hall http://www-prima.inrialpes.fr/perso/Hall/Courses/FAI05/Session7.ppt
6. The material on Extended Kalman filters is courtesy of B. Kuipers http://userweb.cs.utexas.edu/~pstone/Courses/395Tfall05/resources/week11-ben-kalman.ppt