Data Filtering, Smoothing, and Prediction

Kalman Filter + Particle Filter

Relation to This Course

- The famous KF is based on parametric estimation
- The advanced PK is based on density estimation (non-parametric estimation)
- Both use Bayesian frameworks we just discussed

Problem Statement

- A recurring theme in many online analysis and prediction tasks
- How can information
  - from different sources
  - with different accuracy (corrupted by noise)
  - may even be time varying

  Be integrated?
    - Estimating the position of a line from multiple sample points
    - Estimating shape using information from multiple sensors
    - Estimating moving robot location from sensor data and dead reckoning

Complication

- There are many more examples, where
  - not all data (observation) are gathered at the same time
  - not all data (observation) are equally reliable
  - the estimated quantities (state) change
  - The state variables may not be single-peaked (PF)

Progression -KF

- We will progress through a number of scenarios
  - Static state, observation data available all at once, of the same quality
  - Static state, observation data available all at once, of different quality
  - Static state, observation data not all at once, of the same or different quality
  - Dynamic state, observation data not all at once, of the same or different quality

General Principles

- Bayesian principle underlies all of the analysis
- Two things to remember (because we will use them over and over again)
  - Data should be trusted based on their expected accuracy
    - Weighted sum based on covariance
  - State should be trusted based on their ability to explain the sensor observation
    - Covariance can change

Simplest Case

- Two (or multiple) measurements with the same or different uncertainty
- States are directly measured

Batch

$$\hat{x} = x_1 + x_2$$

$$\hat{x} = \sigma_2^2 \frac{x_1}{\sigma_1^2 + \sigma_2^2} + \sigma_1^2 \frac{x_2}{\sigma_1^2 + \sigma_2^2}$$

$$\frac{1}{\sigma^2} = \frac{1}{\sigma_1^2} + \frac{1}{\sigma_2^2} \Rightarrow \sigma^2 \leq \sigma_1^2, \sigma^2 \leq \sigma_2^2$$

Iterative Gain innovation

$$\hat{x} = x_1 + \boxed{\frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2}} \boxed{(x_2 - x_1)}$$

$$\sigma^2 = \sigma_1^2 - \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2} \sigma_1^2$$

Some Important Intuition

- Information is good
  - Variance will always decrease
- All information can and should be used
  - The worst case is to ignore totally uncertain information
- Information integration can be incremental
  - In terms of innovation
  - Properly weighed innovation
  - Not all data at once, not saving all past data

Linear Least Squares

- Second simplest of all formulations
  - States (**X**) are not directly measured
  - Observation (**B**) or measurements relate to state linearly
  - Observation are equally reliable
  - gathered at the same time

$$\mathbf{A}_{\mathbf{m\times n}}\mathbf{X}_{\mathbf{n\times 1}} = \mathbf{B}_{\mathbf{m\times 1}} \Rightarrow \mathbf{A}^{\mathbf{T}}{}_{\mathbf{n\times m}}\mathbf{A}_{\mathbf{m\times n}}\mathbf{X}_{\mathbf{n\times 1}} = \mathbf{A}^{\mathbf{T}}{}_{\mathbf{n\times m}}\mathbf{B}_{\mathbf{m\times 1}}$$

$$\mathbf{X}_{\mathbf{n\times 1}} = (\mathbf{A}^{\mathbf{T}}{}_{\mathbf{n\times m}}\mathbf{A}_{\mathbf{m\times n}})^{-1}\mathbf{A}^{\mathbf{T}}{}_{\mathbf{n\times m}}\mathbf{B}_{\mathbf{m\times 1}}$$

m: number of constraints (observations)
n: number of parameters (states)
if m<n multiple solutions
if m=n exact solution
if m>n least-squares solution
e: noise (assume white and Gaussian)

$$\mathbf{B}_{\mathbf{m\times 1}} = \mathbf{A}_{\mathbf{m\times n}}\mathbf{X}_{\mathbf{n\times 1}} + e_{m\times 1}$$

Weighted Least Square

- Slightly more complicated
  - data are *not* equally reliable
  - gathered at the same time

$$\mathbf{W}_{m\times m}\mathbf{A}_{m\times n}\mathbf{X}_{n\times 1} = \mathbf{W}_{m\times m}\mathbf{B}_{m\times 1}$$

$$(\mathbf{W}_{m\times m}\mathbf{A}_{m\times n})^T\mathbf{W}_{m\times m}\mathbf{A}_{m\times n}\mathbf{X}_{n\times 1} = (\mathbf{W}_{m\times m}\mathbf{A}_{m\times n})^T\mathbf{W}_{m\times m}\mathbf{B}_{m\times 1}$$

$$\mathbf{X}_{n\times 1} = ((\mathbf{W}_{m\times m}\mathbf{A}_{m\times n})^T\mathbf{W}_{m\times m}\mathbf{A}_{m\times n})^{-1}(\mathbf{W}_{m\times m}\mathbf{A}_{m\times n})^T\mathbf{W}_{m\times m}\mathbf{B}_{m\times 1}$$

$$= \left(\mathbf{A}_{m\times n}{}^T\mathbf{W}^T{}_{m\times m}\mathbf{W}_{m\times m}\mathbf{A}_{m\times n}\right)^{-1}\mathbf{A}_{m\times n}{}^T\mathbf{W}^T{}_{m\times m}\mathbf{W}_{m\times m}\mathbf{B}_{m\times 1}$$

$$= \left(\mathbf{A}_{m\times n}{}^T\mathbf{C}_{m\times m}\mathbf{A}_{m\times n}\right)^{-1}\mathbf{A}_{m\times n}{}^T\mathbf{C}_{m\times m}\mathbf{B}_{m\times 1}$$

$$= \mathbf{L}_{m\times m}\mathbf{B}_{m\times 1}$$

$$\Rightarrow \hat{\mathbf{X}} = \mathbf{LB}$$

$$\mathbf{X}_{n\times 1} = (\mathbf{A}^T{}_{n\times m}\mathbf{A}_{m\times n})^{-1}\mathbf{A}^T{}_{n\times m}\mathbf{B}_{m\times 1}$$ (before)

Weighted Least Square (cont.)

- Weights (**W**) do not appear directly but only indirectly in **C**
- What are the right choice of weights?
- It can be shown that the right weights are inversely proportional to the standard deviation in the scalar case and covariance in the vector case
- Kind of make sense - the larger the uncertainty the less you will trust the data

BLUE
(best linear unbiased estimator)

- **C=V**$^{-1}$gives BLUE (V: variance of "noise" in the measurements)
  - Matrix operator is certainly linear
  - Unbiased means that expected error is neither positive or negative $E(\mathbf{X}-\hat{\mathbf{X}})=0$
  - Or an unbiased estimator must be such as it is the left inverse of **A**

$$E(\mathbf{X}-\hat{\mathbf{X}})=E(\mathbf{X}-\mathbf{LB})=E(\mathbf{X}-\mathbf{LAX}-\mathbf{Le})=E[(\mathbf{I}-\mathbf{LA})\mathbf{X}]=0$$
$$\Rightarrow \mathbf{I}=\mathbf{LA}$$

  - Non-square matrices can have multiple left inverse

Proof of Optimality

- Unbiased – all unbiased operators are similar and satisfy

$$\mathbf{L}_o = \left(\mathbf{A}^{\mathbf{T}}\mathbf{V}^{-1}\mathbf{A}\right)^{-1}\mathbf{A}^{\mathbf{T}}\mathbf{V}^{-1} = \mathbf{A}^{-1}\mathbf{V}(\mathbf{A}^{\mathbf{T}})^{-1}\mathbf{A}^{\mathbf{T}}\mathbf{V}^{-1} = \mathbf{A}^{-1}$$

- Optimality

$$\mathbf{e} = \mathbf{B} - \mathbf{A}\hat{\mathbf{X}} = \mathbf{B} - \mathbf{A}(\hat{\mathbf{X}} - \mathbf{X} + \mathbf{X}) = \mathbf{B} - \mathbf{A}(\hat{\mathbf{X}} - \mathbf{X}) + \mathbf{A}\mathbf{X} = \mathbf{A}(\mathbf{X} - \hat{\mathbf{X}}) \Rightarrow (\mathbf{X} - \hat{\mathbf{X}}) = \mathbf{A}^{-1}\mathbf{e} = \mathbf{L}\mathbf{e}$$

$$\mathbf{P} = E[(\mathbf{X} - \hat{\mathbf{X}})(\mathbf{X} - \hat{\mathbf{X}})^T]$$

$$= E[\mathbf{L}\mathbf{e}(\mathbf{L}\mathbf{e})^T] = \mathbf{L}E[\mathbf{e}\mathbf{e}^T]\mathbf{L}^T = \mathbf{L}\mathbf{V}\mathbf{L}^T$$

$$\mathbf{L} = \mathbf{L}_o + (\mathbf{L} - \mathbf{L}_o)$$

$$\mathbf{P} = (\mathbf{L}_o + (\mathbf{L} - \mathbf{L}_o))\mathbf{V}(\mathbf{L}_o + (\mathbf{L} - \mathbf{L}_o))^T$$

$$= \mathbf{L}_o\mathbf{V}\mathbf{L}_o^{\ T} + \underbrace{(\mathbf{L} - \mathbf{L}_o)\mathbf{V}\mathbf{L}_o^{\ T}}_{0} + \underbrace{\mathbf{L}_o\mathbf{V}(\mathbf{L} - \mathbf{L}_o)^T}_{0} + (\mathbf{L} - \mathbf{L}_o)\mathbf{V}(\mathbf{L} - \mathbf{L}_o)^T$$

$$(\mathbf{L} - \mathbf{L}_o)\mathbf{V}\mathbf{L}_o^{\ T} = (\mathbf{L} - \mathbf{L}_o)\mathbf{V}[(\mathbf{A}^T\mathbf{V}^{-1}\mathbf{A})^{-1}\mathbf{A}^T\mathbf{V}^{-1}]^T$$

$$= (\mathbf{L} - \mathbf{L}_o)\mathbf{V}(\mathbf{V}^{-1})^T\mathbf{A}[(\mathbf{A}^T\mathbf{V}^{-1}\mathbf{A})^{-1}]^T$$

$$= (\mathbf{L} - \mathbf{L}_o)\mathbf{V}\mathbf{V}^{-1}\mathbf{A}(\mathbf{A}^T\mathbf{V}^{-1}\mathbf{A})^{-1}$$

$$= (\mathbf{L} - \mathbf{L}_o)\mathbf{A}(\mathbf{A}^T\mathbf{V}^{-1}\mathbf{A})^{-1} = (\mathbf{L}\mathbf{A} - \mathbf{L}_o\mathbf{A})(\mathbf{A}^T\mathbf{V}^{-1}\mathbf{A})^{-1}$$

$$= (\mathbf{I} - \mathbf{I})(\mathbf{A}^T\mathbf{V}^{-1}\mathbf{A})^{-1} = 0(\mathbf{A}^T\mathbf{V}^{-1}\mathbf{A})^{-1} = 0$$

Final Equations

$$\hat{\mathbf{x}} = \mathbf{L}_o\mathbf{B} = \left(\mathbf{A}^{\mathbf{T}}\mathbf{V}^{-1}\mathbf{A}\right)^{-1}\mathbf{A}^{\mathbf{T}}\mathbf{V}^{-1}\mathbf{B}$$

$$\mathbf{P} = E[(\mathbf{X}-\hat{\mathbf{X}})(\mathbf{X}-\hat{\mathbf{X}})^T] = E[(\mathbf{X}-\mathbf{LB})(\mathbf{X}-\mathbf{LB})^T]$$

$$= E[(\mathbf{X}-\mathbf{L}_o\mathbf{AX}-\mathbf{L}_o\mathbf{e}')(\mathbf{X}-\mathbf{L}_o\mathbf{AX}-\mathbf{L}_o\mathbf{e}')^T]$$

$$= E[\mathbf{L}_o\mathbf{e}'(\mathbf{L}_o\mathbf{e}')^T] = \mathbf{L}_o E[\mathbf{e}'\mathbf{e}'^T]\mathbf{L}_o{}^T = \mathbf{L}_o\mathbf{V}\mathbf{L}_o{}^T$$

$$= (\mathbf{A}^T\mathbf{V}^{-1}\mathbf{A})^{-1}\mathbf{A}^T\mathbf{V}^{-1}\mathbf{V}[(\mathbf{A}^T\mathbf{V}^{-1}\mathbf{A})^{-1}\mathbf{A}^T\mathbf{V}^{-1}]^T$$

$$= (\mathbf{A}^T\mathbf{V}^{-1}\mathbf{A})^{-1}\mathbf{A}^T\mathbf{V}^{-1}\mathbf{V}\mathbf{V}^{-1}\mathbf{A}(\mathbf{A}^T\mathbf{V}^{-1}\mathbf{A})^{-1}$$

$$= (\mathbf{A}^T\mathbf{V}^{-1}\mathbf{A})^{-1}\mathbf{A}^T\mathbf{V}^{-1}\mathbf{A}(\mathbf{A}^T\mathbf{V}^{-1}\mathbf{A})^{-1}$$

$$= (\mathbf{A}^T\mathbf{V}^{-1}\mathbf{A})^{-1}$$

Recursive Least Squares

- More complicated
  - data are *not* equally reliable (the same reliability is a special case)
  - gathered *not* at the same time
  - But for the **same** state
- How can we build estimates recursively without recomputing everything from scratch?

$$\mathbf{X}_o = \left(\mathbf{A}_o{}^T \mathbf{V}_o{}^{-1} \mathbf{A}_o\right)^{-1} \mathbf{A}_o{}^T \mathbf{V}_o{}^{-1} \mathbf{B}_o = \mathbf{P}_o \mathbf{A}_o{}^T \mathbf{V}_o{}^{-1} \mathbf{B}_o$$

$$\mathbf{P}_o = \left(\mathbf{A}_o{}^T \mathbf{V}_o{}^{-1} \mathbf{A}_o\right)^{-1}$$

$$\mathbf{V} = \begin{bmatrix} \mathbf{V}_o & \mathbf{0} \\ \mathbf{0} & \mathbf{V}_1 \end{bmatrix}$$

← If noise is uncorrelated over time

$$\mathbf{P}_1{}^{-1} = \begin{bmatrix} \mathbf{A}_o \\ \mathbf{A}_1 \end{bmatrix}^T \begin{bmatrix} \mathbf{V}_o & \mathbf{0} \\ \mathbf{0} & \mathbf{V}_1 \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{A}_o \\ \mathbf{A}_1 \end{bmatrix} = \mathbf{A}_o{}^T \mathbf{V}_o{}^{-1} \mathbf{A}_o + \mathbf{A}_1{}^T \mathbf{V}_1{}^{-1} \mathbf{A}_1 = \mathbf{P}_0{}^{-1} + \mathbf{A}_1{}^T \mathbf{V}_1{}^{-1} \mathbf{A}_1$$

$$\mathbf{X}_1 = \mathbf{P}_1 \begin{bmatrix} \mathbf{A}_o \\ \mathbf{A}_1 \end{bmatrix}^T \mathbf{V}^{-1} \begin{bmatrix} \mathbf{B}_o \\ \mathbf{B}_1 \end{bmatrix} = \mathbf{P}_1 (\mathbf{A}_o{}^T \mathbf{V}_o{}^{-1} \mathbf{B}_o + \mathbf{A}_1{}^T \mathbf{V}_1{}^{-1} \mathbf{B}_1)$$

$$= \mathbf{P}_1 (\mathbf{P}_o{}^{-1} \mathbf{X}_o + \mathbf{A}_1{}^T \mathbf{V}_1{}^{-1} \mathbf{B}_1) \qquad \because \mathbf{X}_o = \mathbf{P}_o \mathbf{A}_o{}^T \mathbf{V}_o{}^{-1} \mathbf{B}_o$$

$$= \mathbf{P}_1 (\mathbf{P}_1{}^{-1} \mathbf{X}_o - \mathbf{A}_1{}^T \mathbf{V}_1{}^{-1} \mathbf{A}_1 \mathbf{X}_o + \mathbf{A}_1{}^T \mathbf{V}_1{}^{-1} \mathbf{B}_1) \qquad \because \mathbf{P}_1{}^{-1} = \mathbf{P}_0{}^{-1} + \mathbf{A}_1{}^T \mathbf{V}_1{}^{-1} \mathbf{A}_1$$

$$= \mathbf{X}_o + \boxed{\mathbf{P}_1 \mathbf{A}_1{}^T \mathbf{V}_1{}^{-1}} \boxed{(\mathbf{B}_1 - \mathbf{A}_1 \mathbf{X}_o)}$$

gain innovation

Final Equations

$$\mathbf{P}_i^{-1} = \mathbf{P}_{i-1}^{-1} + \mathbf{A}_i^{\mathbf{T}} \mathbf{V}_i^{-1} \mathbf{A}_i$$

$$\mathbf{X}_i = \mathbf{X}_{i-1} + \mathbf{P}_i \mathbf{A}_i^{\mathbf{T}} \mathbf{V}_i^{-1} (\mathbf{B}_i - \mathbf{A}_i \mathbf{X}_{i-1})$$

Dynamic States

- State evolves over time
- Two mechanisms
  - Observation: noise white and Gaussian
  - State propagation: noise white and Gaussian

$$\mathbf{A}_i\mathbf{X}_i = \mathbf{B}_i$$
$$\mathbf{F}_i\mathbf{X}_i = \mathbf{X}_{i+1}$$

$$\begin{bmatrix} \mathbf{A}_0 & & \\ -\mathbf{F}_0 & I & \\ & \mathbf{A}_1 & \\ & -\mathbf{F}_1 & I \\ & & \mathbf{A}_2 \end{bmatrix} \begin{bmatrix} \mathbf{X}_0 \\ \mathbf{X}_1 \\ \mathbf{X}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{B}_0 \\ 0 \\ \mathbf{B}_1 \\ 0 \\ \mathbf{B}_2 \end{bmatrix}$$

$$\mathbf{A}_i\mathbf{X}_i + \mathbf{e}_1 = \mathbf{B}_i$$
$$\mathbf{F}_i\mathbf{X}_i + \mathbf{e}_2 = \mathbf{X}_{i+1}$$

$$\begin{bmatrix} \mathbf{A}_0 & & \\ -c\mathbf{F}_0 & cI & \\ & \mathbf{A}_1 & \\ & -c\mathbf{F}_1 & cI \\ & & \mathbf{A}_2 \end{bmatrix} \begin{bmatrix} \mathbf{X}_0 \\ \mathbf{X}_1 \\ \mathbf{X}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{B}_0 \\ 0 \\ \mathbf{B}_1 \\ 0 \\ \mathbf{B}_2 \end{bmatrix}$$

Dynamic States

- Each time instance
  - Add one column ($\mathbf{x_i}$)
  - Add one row $\mathbf{Ax_i=B_i}$
- Solution
  - Gauss said least square
  - Kalman said recursive
  - Kalman wins
  - Do remember that x_o, x_1, x_2, etc are affected by new data b_2
    - x_o and x_1 given b_0, b_1, b_2 a smoothing problem
    - x_2 given b_o, b_1, b_2 a filtering problem

Kalman's Iterative Formulation

- To understand it, you actually need to remember just two things
  - Rule 1: Linear operations on Gaussian random variables remain Gaussian
  - Rule 2: Linear combinations of jointly Gaussian random variables are also Gaussian

$$\mathbf{Y} = \mathbf{AX}$$

$$\mathbf{m}_y = \mathbf{Am}_x$$

$$\mathbf{P}_{yy} = \mathbf{AP}_{xx}\mathbf{A}^T$$

$$\mathbf{Z} = \mathbf{AX} + \mathbf{BY} + \mathbf{C}$$

$$\mathbf{m}_z = \mathbf{Am}_x + \mathbf{Bm}_y + \mathbf{C}$$

$$\mathbf{P}_{zz} = \mathbf{AP}_{xx}\mathbf{A}^T + \mathbf{AP}_{xy}\mathbf{B}^T + \mathbf{BP}_{yx}\mathbf{A}^T + \mathbf{BP}_{yy}\mathbf{B}^T$$

X: states
Y: observations
Z: prediction based on states + observations
A, B, C: linear prediction mechanism (from **X, Y** to **Z**)
P: covariance matrix

More Rules

- Rule 3: Any portion of a Gaussian random vector is still a Gaussian

$$\mathbf{Z} = \begin{bmatrix} \mathbf{X} \\ \mathbf{Y} \end{bmatrix}$$

$$\mathbf{m}_z = \begin{bmatrix} \mathbf{m}_x \\ \mathbf{m}_y \end{bmatrix}$$

$$\mathbf{P}_{zz} = \begin{bmatrix} \mathbf{P}_{xx} & \mathbf{P}_{xy} \\ \mathbf{P}_{yx} & \mathbf{P}_{yy} \end{bmatrix}$$

Intuition

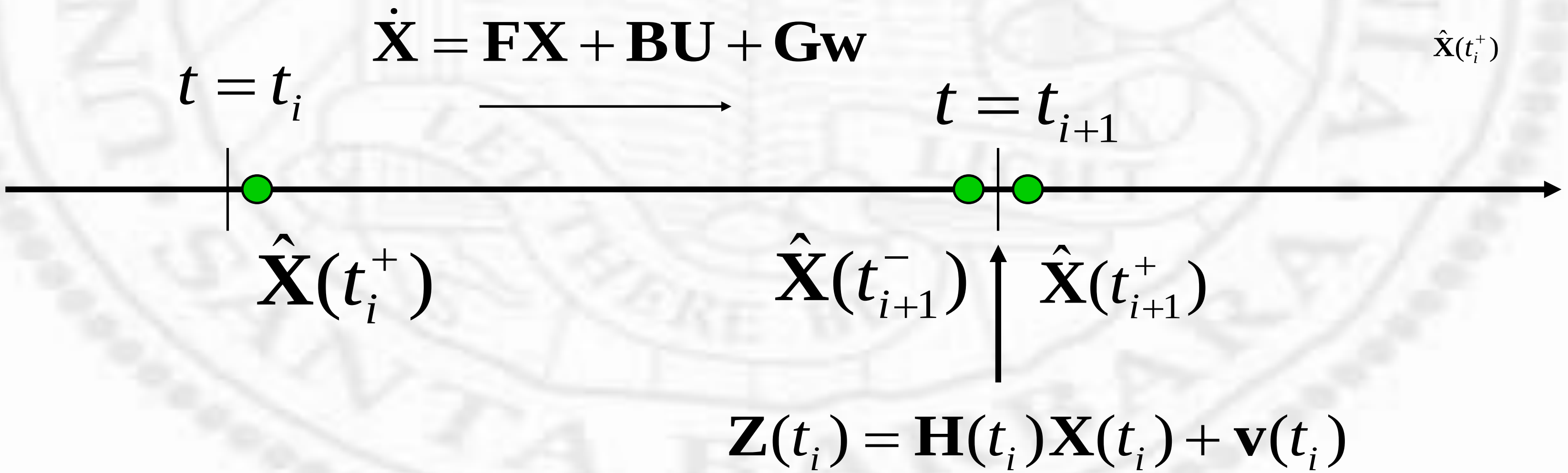
- Initial state estimate is Gaussian
- State propagation mechanism is linear
- Propagation of state over time is corrupted by Gaussian noise
- Sensor measurement is linearly related to state
- Sensor measurement also corrupted by Gaussian noise
- Updated state estimate is again Gaussian

Kalman Filter Properties

- For linear system and white Gaussian errors, Kalman filter is “best” estimate based on all previous measurements
- For non-linear system optimality is ‘qualified’ (EKF, SKF, etc.)
- Doesn’t need to store all previous measurements and reprocess all data each time step

Graphic Illustration

- When noise is white and uncorrelated
- Starting out as a Gaussian process the evolution will stay a Gaussian process



Math Details

- If Gaussian assumption is assumed, all we need to derive are the mechanisms for propagating mean and variance Using the now familiar update equation of
  - New = old + gain * innovation
  - Goal: determine the right gain expression

$$\mathbf{X}_i = \mathbf{X}_{i-1}^{+} + \mathbf{K}_i(\mathbf{Z}_i - \mathbf{H}_i\mathbf{X}_i^{-})$$

Starting Condition

$$\mathbf{X}_{i+1} = \mathbf{\Phi}_i \mathbf{X}_i + \mathbf{w}_i$$

$$\mathbf{z}_i = \mathbf{H}_i \mathbf{X}_i + \mathbf{v}_i$$

$$E(\mathbf{w}_i \mathbf{w}_j^T) = \begin{cases} \mathbf{Q}_i & i = j \\ 0 & i \neq j \end{cases}$$

$$E(\mathbf{v}_i \mathbf{v}_j^T) = \begin{cases} \mathbf{R}_i & i = j \\ 0 & i \neq j \end{cases}$$

$$E(\mathbf{w}_i \mathbf{v}_j^T) = 0$$

State Propagation

$$
\begin{aligned}
\hat{\mathbf{X}}^{-}_{i+1} &= \mathbf{\Phi}_i \hat{\mathbf{X}}^{+}_i \\
\mathbf{P}^{-}_{i+1} &= E\left[(\mathbf{X}^{-}_{i+1} - \hat{\mathbf{X}}^{-}_{i+1})(\mathbf{X}^{-}_{i+1} - \hat{\mathbf{X}}^{-}_{i+1})^T\right] \\
&= E\left[(\mathbf{\Phi}_i \mathbf{X}^{+}_i + \mathbf{w}_i - \mathbf{\Phi}_i \hat{\mathbf{X}}^{+}_i)(\mathbf{\Phi}_i \mathbf{X}^{+}_i + \mathbf{w}_i - \mathbf{\Phi}_i \hat{\mathbf{X}}^{+}_i)^T\right] \\
&= E\left[(\mathbf{\Phi}_i (\mathbf{X}^{+}_i - \hat{\mathbf{X}}^{+}_i) + \mathbf{w}_i)(\mathbf{\Phi}_i (\mathbf{X}^{+}_i - \hat{\mathbf{X}}^{+}_i) + \mathbf{w}_i)^T\right] \\
&= E\left[\mathbf{\Phi}_i (\mathbf{X}^{+}_i - \hat{\mathbf{X}}^{+}_i)(\mathbf{X}^{+}_i - \hat{\mathbf{X}}^{+}_i)^T \mathbf{\Phi}^T_i\right] + E(\mathbf{w}_i \mathbf{w}^T_j) \\
&= \mathbf{\Phi}_i \mathbf{P}^{+}_i \mathbf{\Phi}^T_i + \mathbf{Q}_i
\end{aligned}
$$

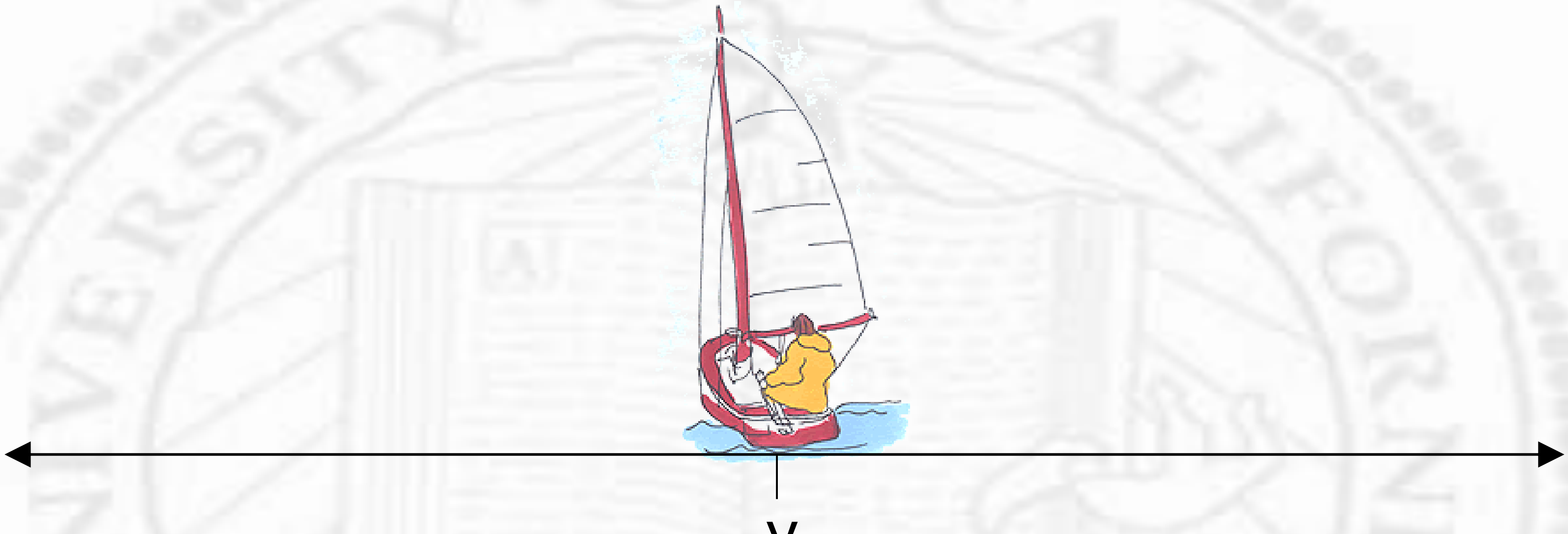
State Update

$$\hat{\mathbf{X}}^{+}_{i+1} = \hat{\mathbf{X}}^{-}_{i+1} + \mathbf{K}_{i+1}(\mathbf{Z}_{i} - \mathbf{H}_{i+1}\mathbf{X}^{-}_{i+1})$$

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

$$\mathbf{P}^{+}_{i+1} = (\mathbf{I} - \mathbf{K}_{i+1}\mathbf{H}_{i+1})\mathbf{P}^{-}_{i+1}$$

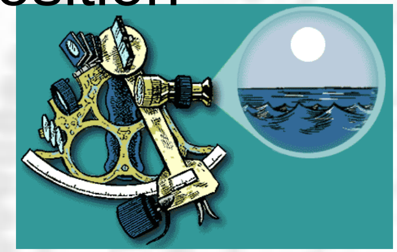
Conceptual Overview

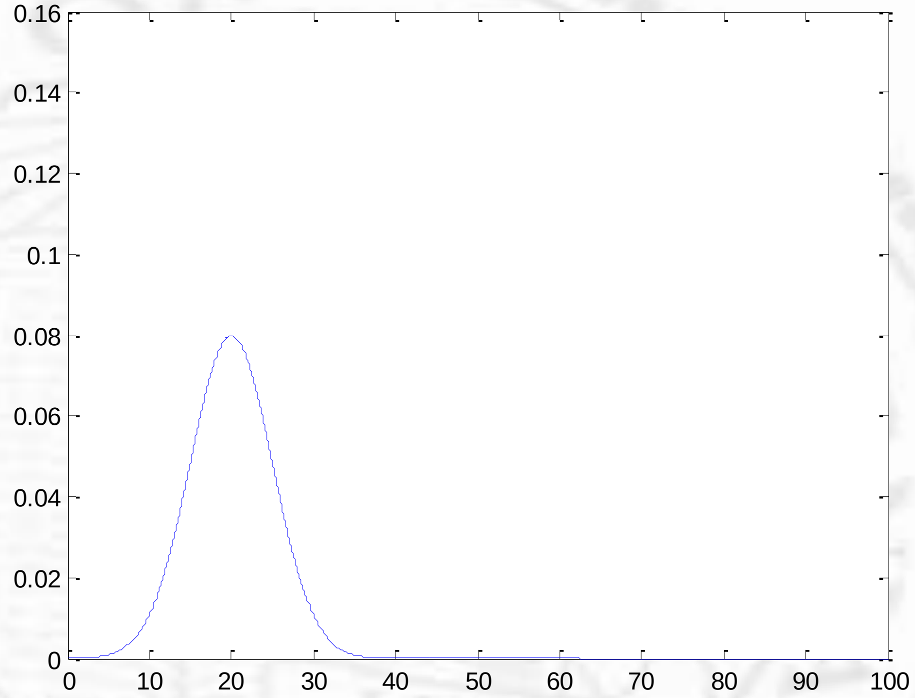


- Lost on the 1-dimensional line (imagine that you are guessing your position by looking at the stars using sextant)
- Position – y(t)
- Assume Gaussian distributed measurements

Conceptual Overview

State space – position
Measurement - position



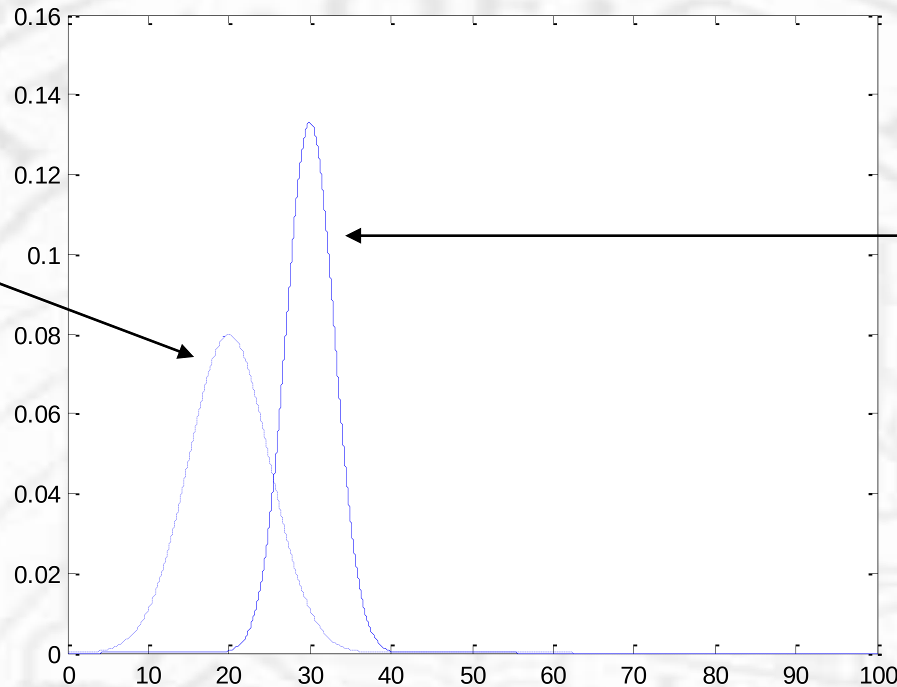


- Sextant Measurement at t_1: Mean = z_1 and Variance = σ_{z1}
- Optimal estimate of position is: $\hat{y}(t_1) = z_1$
- Variance of error in estimate: $\sigma^2_x(t_1) = \sigma^2_{z1}$
- **Boat in same position** at time t_2 - Predicted position is z_1

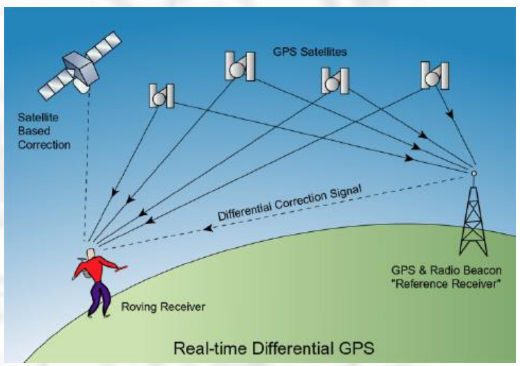
Sextant is not perfect

Conceptual Overview

prediction $\hat{y}^{-}(t_2)$
State (by looking at the stars at t2)

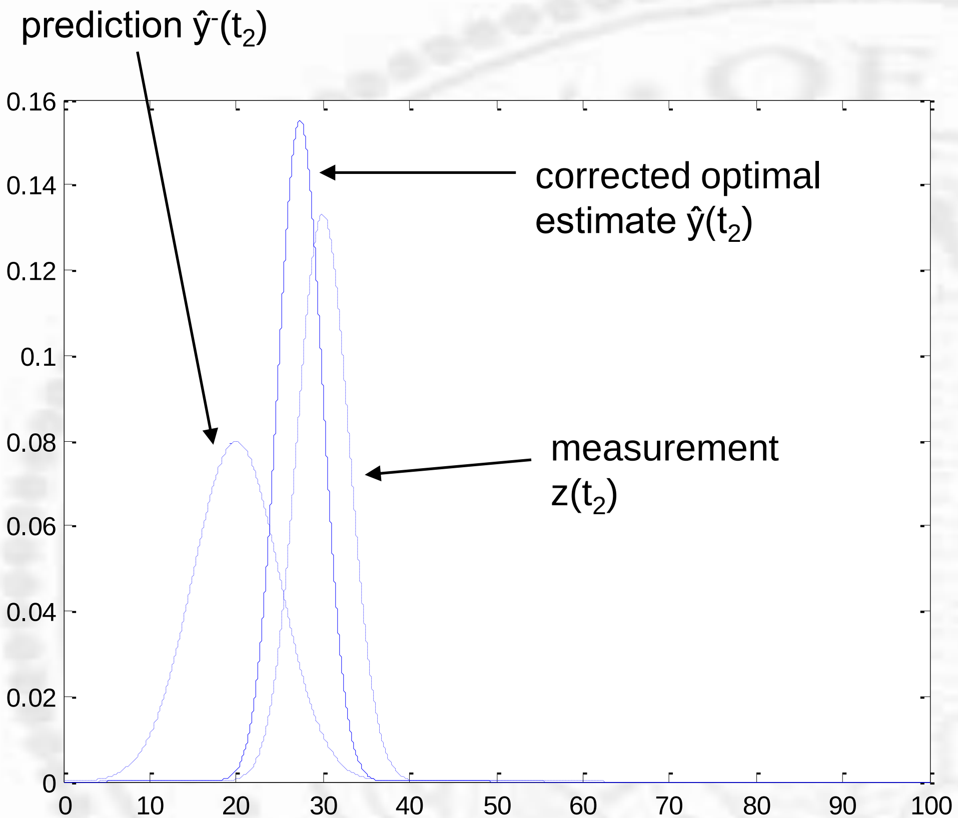


Measurement usign GPS $z(t_2)$



- So we have the prediction $\hat{y}^{-}(t_2)$
- GPS Measurement at t_2: Mean = z_2 and Variance = σ_{z2}
- Need to <u>correct</u> the prediction by Sextant due to measurement to get $\hat{y}(t_2)$
- Closer to more trusted measurement – should we do linear interpolation?

Conceptual Overview



Kalman filter helps you fuse measurement and prediction on the basis of how much you trust each

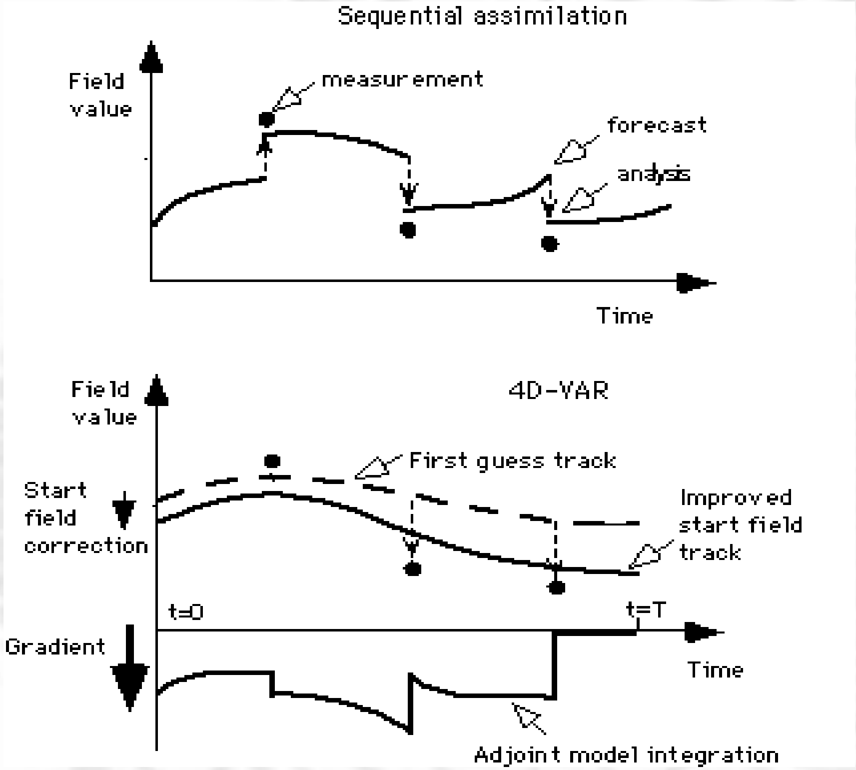
(I would trust the GPS more than the sextant)

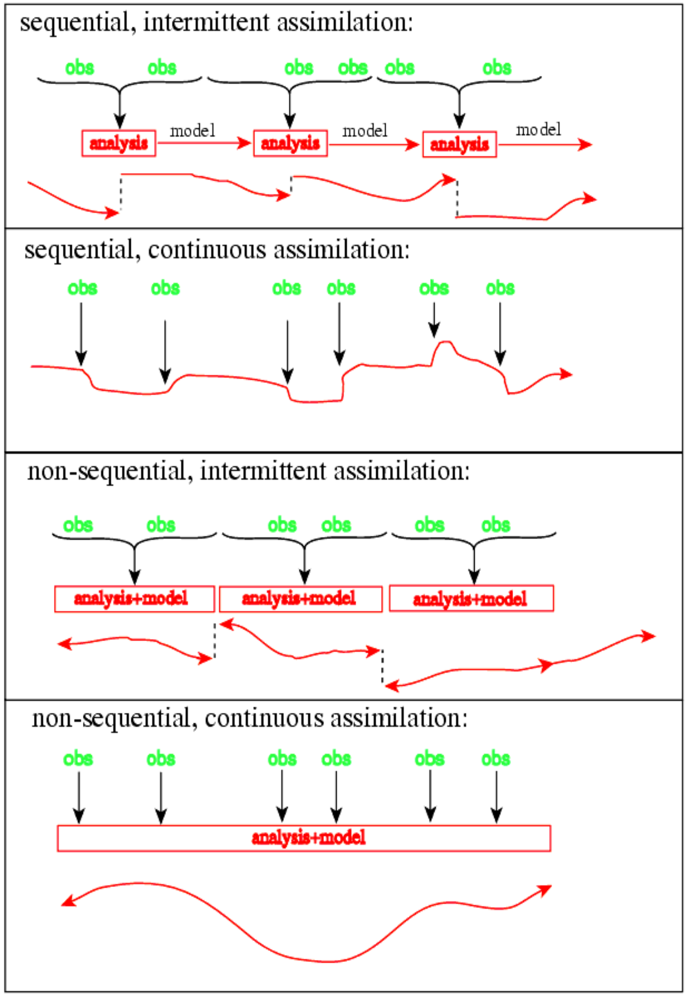
- Corrected mean is the new optimal estimate of position (basically you've 'updated' the predicted position by Sextant using GPS
- New variance is smaller than either of the previous two variances

More Example

- Suppose you have a hydrologic model that predicts river water level every hour (using the usual inputs).
- You know that your model is not perfect and you don't trust it 100%. So you want to send someone to check the river level in person.
- However, the river level can only be checked once a day around noon and not every hour.
- Furthermore, the person who measures the river level can not be trusted 100% either.
- So how do you combine both outputs of river level (from model and from measurement) so that you get a 'fused' and better estimate? – Kalman filtering

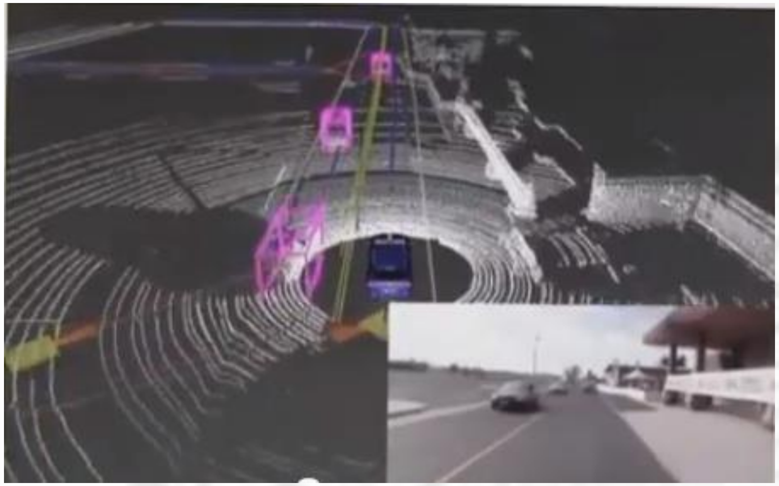
Graphically speaking

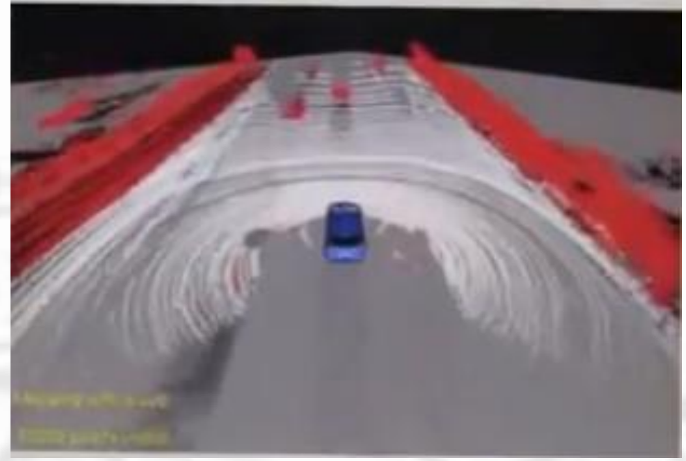


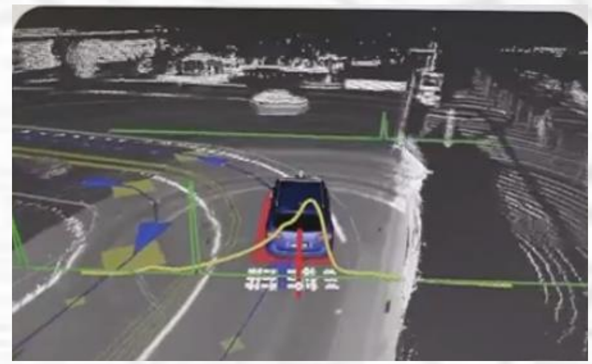


Navigation using PF

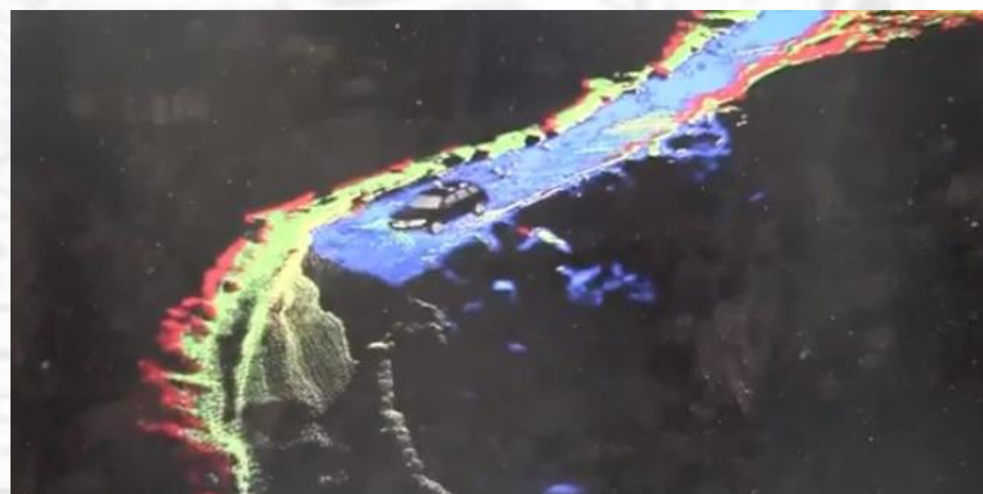
- Autonomous Land Vehicle (ALV), Google's Self-Driving Car, etc.
- One important requirement: track the position of the vehicle
- Kalman Filter, loop of
  - (Re)initialization
  - Prediction
  - Observation
  - Correction








intel
MDV
NXP



Interesting YouTube Videos

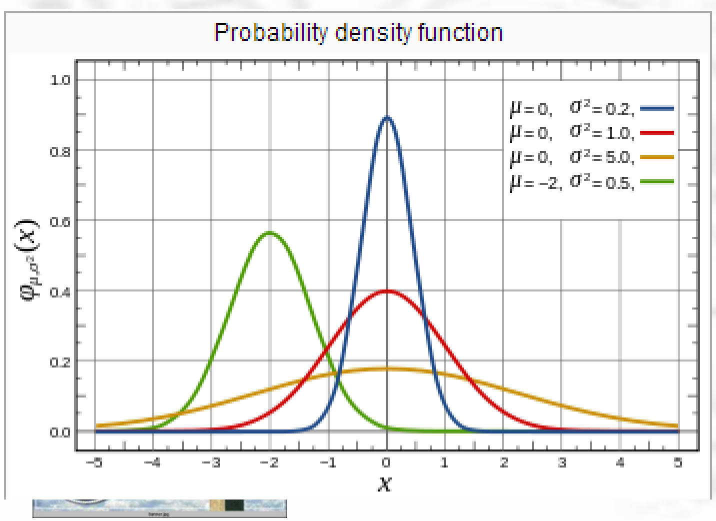
- Introduction to Autonomous Vehicle
- Introduction to Robot Localization
- Introduction to Particle Filters
- Example of Probabilistic Localization
- Example of Probabilistic Localization Using Particle Filters
- Monte Carlo Localization Formulation for Vehicle Localization
- Particle Filters Algorithms

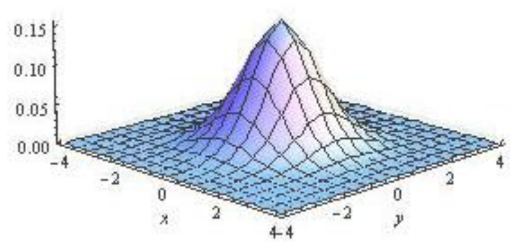
Navigation

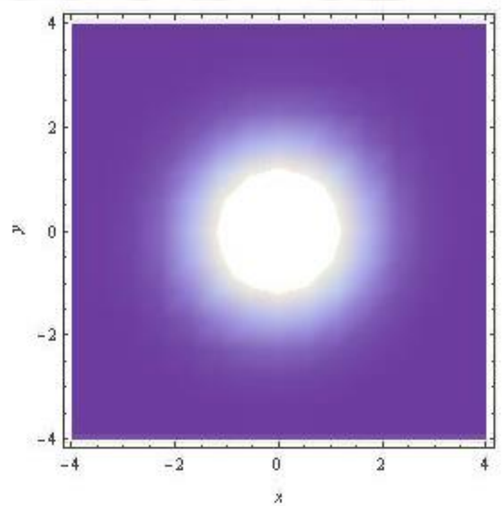
- Hypothesis and verification
  - Classic Approach like Kalman Filter maintains a single hypothesis
  - Newer approach like particle filter maintains multiple hypotheses (Monte Carlo sampling of the state space)

Single Hypothesis

- If the "distraction" – noise – is white and Gaussian
- State-space probability profile remains Gaussian (a single dominant mode)
- Evolving and tracking the mean, not a whole distribution







Multi-Hypotheses

- The distribution can have multiple modes
- Sample the probability distribution with “importance” rating
- Evolve the whole distribution, instead of just the mean

Key – Baeys Rule

$$P(s_i \mid o) = \frac{p(o, s_i)}{p(o)} = \frac{p(o \mid s_i)P(s_i)}{p(o)} \approx p(o \mid s_i)P(s_i)$$

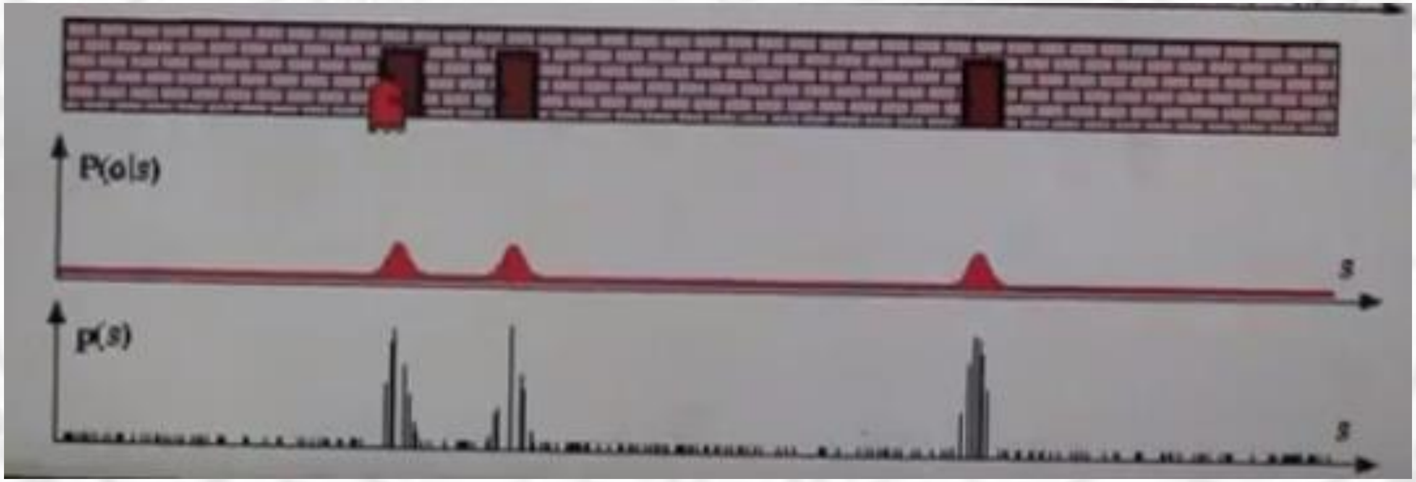
$s: state$

$o: observation$

- In the day time, some animal runs in front of you on the bike path, you know exactly what it is (p(o|si) is sufficient)
- In the night time, some animal runs in front of you on the bike path, you can hardly distinguish the shape (p(o|si) is low for all cases, but you know it is probably a squirrel, not a lion because of p(si))

Initialization: before observation and measurement

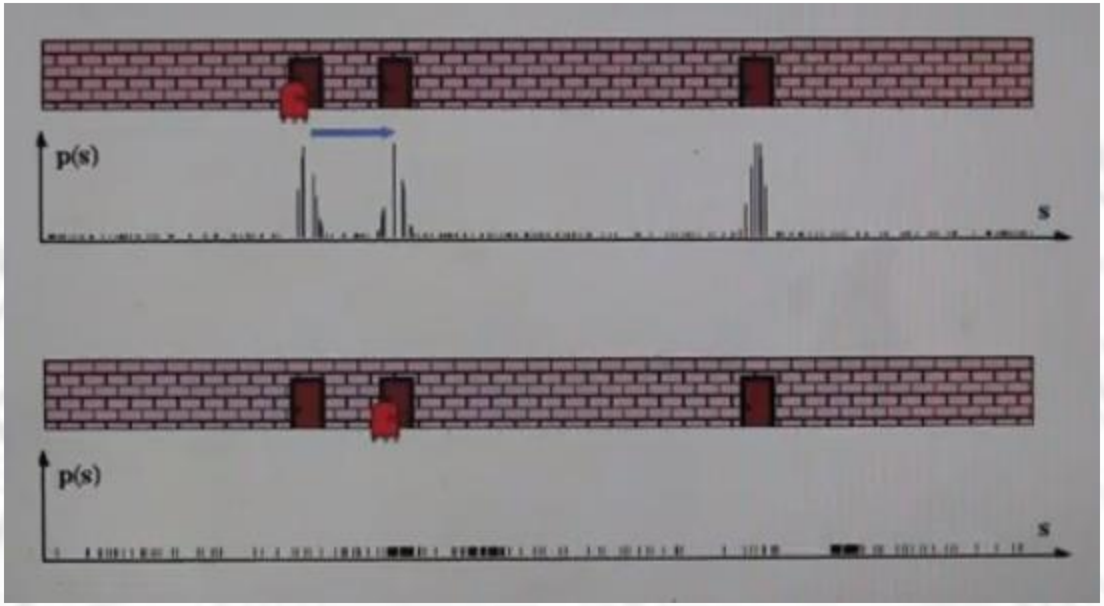
Observation: after seeing a door



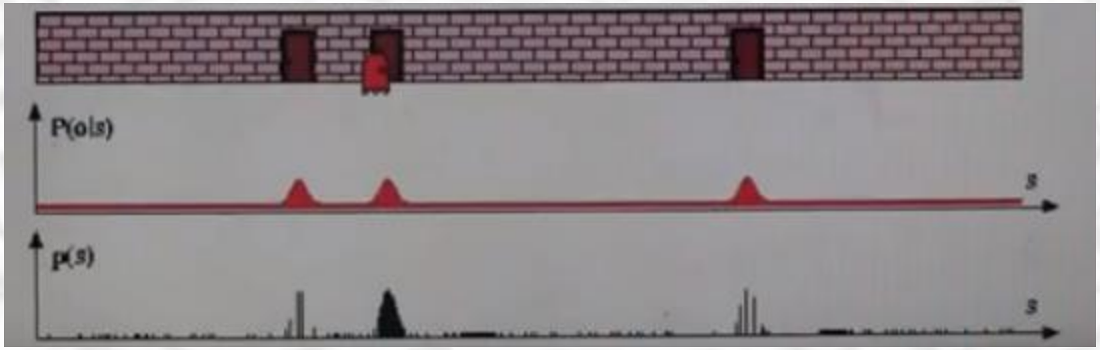
P(s): probability of state

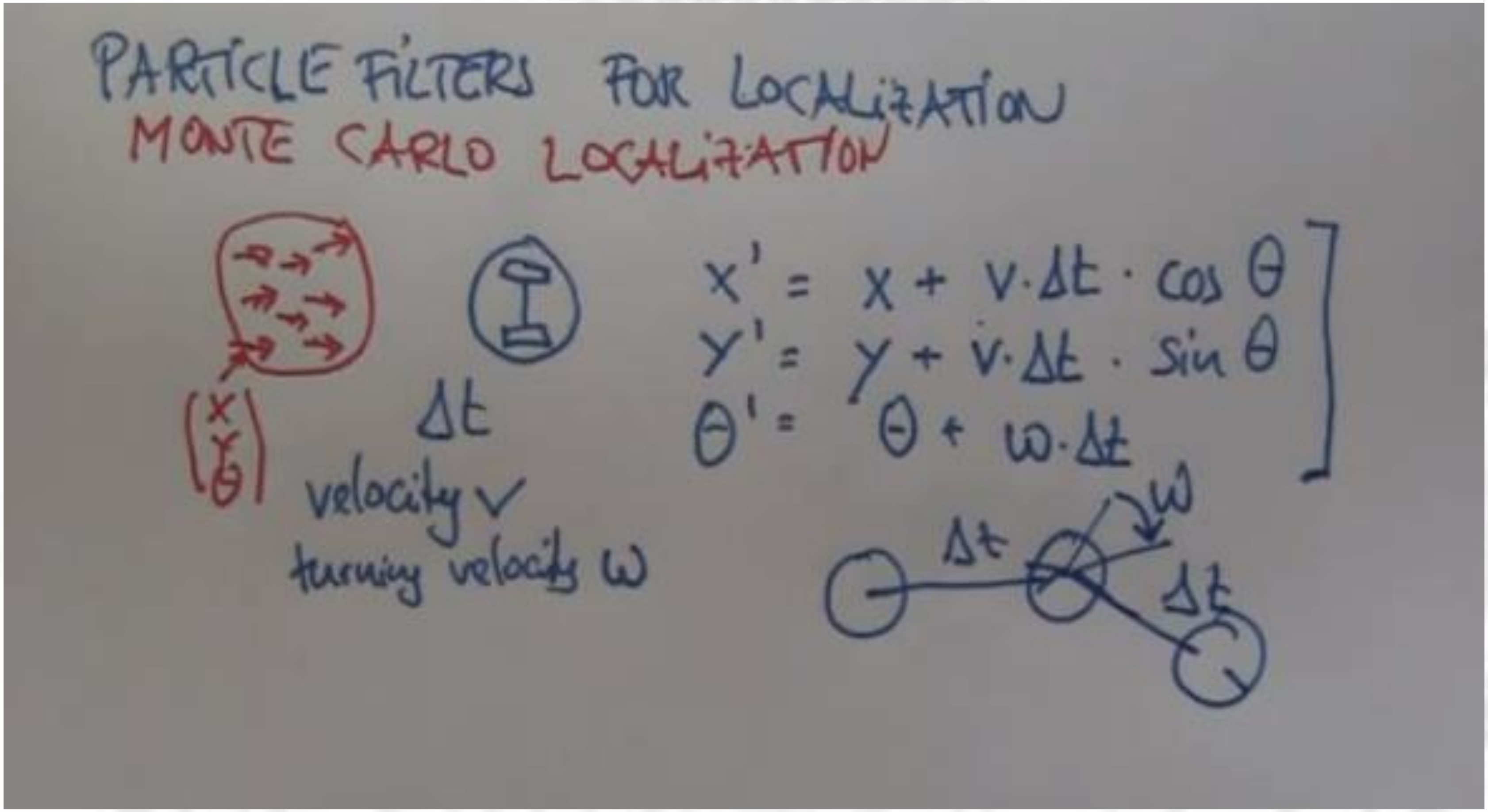
P(o|s): probably of observation given current state

Prediction : internal mechanism saying that robot moves right



Correction : prediction is weighed by confirmation with observation




PARTICLE FILTERS FOR LOCALIZATION
MONTE CARLO LOCALIZATION
x' = x + v·Δt · cos θ
y' = y + v·Δt · sin θ
θ' = θ + ω·Δt
Δt
velocity v
turning velocity ω
ω
Δt
Δt

Particles + weights

controls

measurements

ALGORITHM PARTICLE FILTER (S , U, Z)

new particles + weights

$S' = \emptyset$ $\quad \eta = 0$ ← Total weights

For $i = 1 \ldots n$

Sample $j \sim \{w\}$ with replacement

$x' \sim p(x' \mid u, s_j)$

$w' = p(z \mid x')$

$\eta = \eta + w'$

$S' = S' \cup \{ \langle x', w' \rangle \}$

end

For $i = 1 \ldots n$

$w_i = \frac{1}{\eta} w_i$