Goals:
- Review the Kalman filtering problem for state estimation and sensor fusion
- Describes extensions to KF: information filters, moving horizon estimation

Reading:
- OBC08, Chapter 4 - Kalman filtering
- OBC08, Chapter 5 - Sensor fusion
The State Estimation Problem

Problem Setup
• Given a dynamical system with noise and uncertainty, estimate the state

\[
\dot{x} = Ax + Bu + Fv \\
y = Cx + Du + Gw
\]

\[
\hat{x} = \alpha(\hat{x}, y, u) \quad \text{estimator}
\]

\[
\lim_{t \to \infty} E\{x - \hat{x}\} = 0 \quad \text{expected value}
\]

\[
\hat{x} \quad \text{estimate of } x
\]

Discrete-time systems
\[
x[k+1] = Ax[k] + Bu[k] + Fv[k] \\
y[k] = Cx[k] + w[k],
\]

\[
\hat{x}[k+1] = A\hat{x}[k] + Bu[k] + L(y[k] - C\hat{x}[k]) \quad \text{prediction correction}
\]

System description
\[
x[k+1] = Ax[k] + Bu[k] + Fv[k] \\
y[k] = Cx[k] + w[k],
\]

\[
E\{v[k]\} = 0 \\
E\{v[k]v[j]^T\} = \begin{cases} 0 & k \neq j \\ R_v & k = j \end{cases}
\]

• Disturbances and noise are multi-variable Gaussians with covariance \(R_v, R_w\)

Problem statement: Find the estimate that minimizes the mean square error

\[
E\{(x[k] - \hat{x}[k])(x[k] - \hat{x}[k])^T\}
\]

Proposition
• For Gaussian noise, optimal estimate is the expectation of the random process \(x\) given the constraint of the observed output:

\[
\hat{x}[k] = E\{X[k] \mid Y[l], l \leq k\}
\]

• Can think of this as a least squares problem: given all previous \(y[k]\), find the estimate \(\hat{x}[k]\) that satisfies the dynamics and minimizes the square error with the measured data.
Kalman Filter

Thm (Kalman, 1961) The observer gain $L$ that minimizes the mean square error is given by

$$L[k] = AP[k]C^T(R_w + CP[k]C^T)^{-1}$$

where

$$P[k+1] = (A - LC)P[k](A - LC)^T + R_v + LR_wL^T$$

$$P_0 = E\{X(0)X^T(0)\}.$$  

Proof (easy version). Let $P[k] = E\{(\hat{x}[k] - x[k])(\hat{x}[k] - x[k])^T\}$. By definition,

$$P[k+1] = E\{x[k+1|x[k+1]^T \}

= AP[k]A^T - AP[k]C^TR_e L^T - LCA^T + L(R_w + CP[k]C^T)L^T.$$

Letting $R_e = (R_w + CP[k]C^T)$,

$$P[k+1] = AP[k]A^T + (L - AP[k]C^TR_e^{-1})R_e(L - AP[k]C^TR_e^{-1})^T

- AP[k]C^T R_e^{-1} CP[k]^T A^T + R_w.$$  

to minimize covariance, choose $L = AP[k]C^T R_e^{-1}$

Kalman Filtering with Intermittent Data

Kalman filter has “predictor-corrector” form

$$\hat{x}[k+1] = A\hat{x}[k] + Bu[k] + L(y[k] - C\hat{x}[k])$$

$$P[k+1] = AP[k]A^T + R_w - AP[k]C^T R_e^{-1} CP[k]^T A^T$$

- Key idea: updated prediction on each iteration; apply correction when data arrives

Alternative formulation

- Prediction:
  $$\hat{x}[k+1|k] = A\hat{x}[k|k] + Bu[k]$$
  $$P[k+1|k] = AP[k|k]A^T + F R_v[k] F^T$$

- Correction:
  $$\hat{x}[k|k] = \hat{x}[k|k-1] + L[k](y[k] - C\hat{x}[k|k-1])$$
  $$P[k|k] = P[k|k-1] - P[k|k-1]C^T(CP[k|k-1]C^T + R_w[k])^{-1}CP[k|k-1]$$
Kalman-Bucy Filter

System dynamics: linear process + Gaussian white noise
\[ \dot{x} = Ax + Bu + Fv \quad E\{v(s)v^T(t)\} = Q(t)\delta(t-s) \]
\[ y = Cx + w \quad E\{w(s)w^T(t)\} = R(t)\delta(t-s) \]

Estimator: prediction + correction
\[ \hat{x} = A\hat{x} + Bu + L(y - C\hat{x}) \]
\[ L(t) = P(t)C^TR^{-1} \]
\[ P(t) = E\{(x(t) - \hat{x}(t))(x(t) - \hat{x}(t))^T\} \]

Covariance update
\[ \dot{P} = AP + PA^T - PC^TR^{-1}(t)CP + FQ(t)F^T \]
\[ P(0) = E\{x(0)x^T(0)\} \]

Variation #1: Sensor Fusion

What happens if we have redundant sensors?
- Kalman filter “fuses” data measurements according to covariance
\[ \hat{x} = A\hat{x} + Bu + L(y - C\hat{x}) \]
\[ L(t) = P(t)C^TR^{-1} \]
\[ P(t) = E\{(x(t) - \hat{x}(t))(x(t) - \hat{x}(t))^T\} \]

- Assume R is diagonal, expand out gain:
\[ L(t) = P(t)C^T \begin{bmatrix} R_{11}^{-1} & \cdots \\ \vdots & \ddots \\ R_{nn}^{-1} \end{bmatrix} \]

- Process disturbances
\[ \dot{P} = AP + PA^T - PC^TR^{-1}(t)CP + FQ(t)F^T \]

- Steady state (ARE): optimal balance of dynamics and uncertainty
Variation #2: Extended Kalman Filter (EKF)

Consider a nonlinear system
\[
\dot{x} = f(x, u, v) \quad x \in \mathbb{R}^n, u \in \mathbb{R}^m
\]
\[
y = Cx + w \quad v, w \text{ Gaussian white noise processes with}
\]
\[
\text{covariance matrices } Q \text{ and } R.
\]

Form estimator using nonlinear model + linear feedback
\[
\hat{x} = f(\hat{x}, u, 0) + L(y - C\hat{x})
\]

Compute estimator gain based on linearization at current estimated state:
\[
\begin{align*}
\dot{x} &= f(\hat{x}, u, 0) + L(y - C\hat{x}) \\
\dot{P} &= (\ddot{A} - LC)P + P(\ddot{A} - LC)^T + \dot{F}Q\dot{F}^T + LRL^T \\
L &= PC^T R^{-1} \\
P(t) &= E\{x(t_0)x^T(t_0)\}
\end{align*}
\]
\[
\ddot{A} = \left. \frac{\partial F}{\partial x} \right|_{(\hat{x}, u, 0)} \quad \ddot{F} = \left. \frac{\partial F}{\partial u} \right|_{(\hat{x}, u, 0)}
\]

- Little formal theory, but works very well as long as estimated state is close
- Very important for tracking problems (might operate far from equilibrium)

Example: GPS + IMU localization in Alice

Nonlinear dynamics (simplified)
\[
\dot{x} = \cos \theta \, v \\
\dot{y} = \sin \theta \, v \\
\dot{\theta} = \frac{\dot{v}}{\ell} + \frac{v}{\ell} \cos^2 \phi
\]

Results
- If only \( x \) and \( y \) are measured, get larger errors in state estimate
- Adding angular rate measurement improves performance (right)
Variation #3: Parameter Estimation

Suppose dynamics depend on unknown parameter $\xi$

\[
\dot{x} = A(\xi)x + B(\xi)u + Fv \quad \xi \in \mathbb{R}^p
\]
\[
y = C(\xi)x + w
\]

Rewrite dynamics using added state $\xi$

\[
\dot{x} = A(\xi)x + B(\xi)u + Fv
\]
\[
\dot{\xi} = 0
\]

Now use extended Kalman filter to estimate state and parameter:

\[
\frac{d}{dt} \begin{bmatrix} x \\ \xi \end{bmatrix} = f\left(\begin{bmatrix} x \\ \xi \end{bmatrix}, u, v\right) + \begin{bmatrix} A(\xi) \\ 0 \end{bmatrix} \begin{bmatrix} x \\ \xi \end{bmatrix} + \begin{bmatrix} B(\xi) \\ 0 \end{bmatrix} u + \begin{bmatrix} F \\ 0 \end{bmatrix} v
\]
\[
y = C(\xi)x + w
\]
\[
h(\begin{bmatrix} x \\ \xi \end{bmatrix}, w)
\]

Example: estimate wheel base, sensor locations on Alice

Example: Autonomous Driving

Computing
- 6 Dell 750 PowerEdge Servers (P4, 3GHz)
- 1 IBM Quad Core AMD64 (fast!)
- 1 Gb/s switched ethernet

Sensing
- 5 cameras: 2 stereo pairs, roadfinding
- 5 LADARs: long, med*2, short, bumper
- 2 GPS units + 1 IMU (LN 200)
- 0.5-1 Gb/s raw data rates
State Estimation

State estimation: a state
- Broadcast current vehicle state to all modules that require it (many)
- Timing of state signal is critical - use to calibrate sensor readings
- Quality of state estimate is critical: use to place terrain features in global map
- Issue: GPS jumps
  - Can get 20-100 cm jumps as satellites change positions
  - Maintain continuity of state at same time as insuring best accuracy

A state
- HW: 2 GPS units (2-10 Hz update), 1 inertial measurement unit (gyro, accel @ 400 Hz)
- In: actuator commands, actuator values, engine state
- Out: time-tagged position, orientation, velocities, accelerations
- Use vehicle wheel speed + brake command/position to check if at rest

Terrain Estimation

Sensor processing
- Construct local elevation based on measurements and state estimate
- Compute speed based on gradients

Sensor fusion
- Combine individual speed maps
- Process “missing data” cells

Road finding
- Identify regions with road features
- Increase allowable speed along roads

LadarFeeder, StereoFeeder
- HW: LADAR (serial), stereo (firewire)
- In: Vehicle state
- Out: Speed map (deltas)
- Multiple computers to maintain speed

FusionMapper
- In: Sensor speed maps (deltas)
- Output: fused speed map
- Run on quadcore AMD64
Example: Kalman Filtering for Terrain (Gillula)

KF Framework:
• State to estimate is elevation of each cell
• Elevation is static – so no time updates!

Kalman Filtering:
Propagation Equations:
\[
\begin{align*}
\hat{z}_{i,j}(k+1|k) &= \hat{z}_{i,j}(k|k) \\
P_{i,j}(k+1|k) &= P_{i,j}(k|k)
\end{align*}
\]

Update Equations:
\[
\begin{align*}
\hat{z}_{i,j}(k+1|k+1) &= \frac{R \hat{z}_{i,j}(k+1|k) + P_{i,j}(k+1|k)z_{m}}{P_{i,j}(k+1|k) + R} \\
P_{i,j}(k+1|k+1) &= \frac{P_{i,j}(k+1|k)R}{P_{i,j}(k+1|k) + R}
\end{align*}
\]

The Results – Accurate Elevation and Covariance

Elevation Map:
• Individual sensors
• Fused map

Covariance Map:
A: All sensors
B: Just LADARs
C: LADAR in place
Extension: Information Filter

Idea: rewrite Kalman filter in terms of inverse covariance

\[ I[k | k] := P^{-1}[k | k], \quad \hat{Z}[k | k] := P^{-1}[k | k] \hat{X}[k | k] \]
\[ \Omega_i[k] := C_i^T R_i^{-1} [k] C_i, \quad \Psi_i[k] := C_i^T R_i^{-1} [k] C_i \hat{X}[k | k] \]

Resulting update equations become linear:

\[ \hat{X}[k | k - 1] = (1 - \Gamma[k] F^T) A^{-T} \hat{X}[k - 1 | k - 1] + I[k | k - 1] B u \]
\[ I[k | k] = I[k | k - 1] + \sum_{i=1}^{q} \Omega_i[k] \]
\[ \hat{Z}[k | k] = \hat{Z}[k | k - 1] + \sum_{i=1}^{q} \Psi_i[k] \]
\[ M[k] = A^{-T} P^{-1} [k - 1 | k - 1] A^{-1} \]
\[ \Gamma[k] = M[k] F \sigma^{-1} [k] \]
\[ \Sigma[k] = F^T M[k] F + R^{-1} \]

Remarks

- Information form allows simple addition for correction step: “additional measurements add information”
- Sensor fusion: each additional sensor increases the information
- Multi-rate sensing: whenever new information arrives, add it to the scaled estimate, information matrix; no date => prediction update only
- Derivation of the information filter is non-trivial; not easy to derive from Kalman filter

Extension: Moving Horizon Estimation

System description:

\[ x_{k+1} = f_k(x_k, w_k) \]
\[ y_k = h_k(x_k) + v_k \]
\[ x_k \in \mathbb{X}_k, \quad w_k \in \mathbb{W}_k, \quad v_k \in \mathbb{V}_k. \]

The problem: Given the data

\[ Y_k = \{ y_i : 0 \leq i \leq k \}, \]

find the “best” (to be defined) estimate \( \hat{x}_{k+m} \) of \( x_{k+m} \).

(\( m = 0 \) filtering, \( m > 0 \) prediction, and \( m < 0 \) smoothing)

Pose as optimization problem:

\[ \{ \hat{x}_0, \ldots, \hat{x}_T \} = \arg \max_{\{x_0, \ldots, x_T\}} p(x_0, \ldots, x_T | Y_{T-1}) \]

Remarks:

- Basic idea is to compute out the "noise" that is required for data to be consistent with model and penalize noise based on how well it fits its distribution
Extension: Moving Horizon Estimation

Solution: write out probability and maximize

$$\arg \max_{\{x_0, \ldots, x_T\}} p(x_0, \ldots, x_T | y_0, \ldots, y_{T-1})$$

$$= \arg \max_{\{x_0, \ldots, x_T\}} p_0(x_0) \prod_{k=0}^{T-1} p_k(y_k - h(x_k))p(x_{k+1} | x_k)$$

$$= \arg \max_{\{x_0, \ldots, x_T\}} \sum_{k=0}^{T-1} \log p_k(y_k - h(x_k)) + \log p(x_{k+1} | x_k) + \log p_0(x_0)$$

Special case: Gaussian noise

$$\min_{x_0, \{w_0, \ldots, w_{T-1}\}} \sum_{k=0}^{T-1} \| y_k - h_k(x_k) \|^2_{R_k} + \| w_k \|^2_{Q_k} + \| x_0 - \bar{x}_0 \|^2_{P_0}$$

• Log of the probabilities sum of squares for noise terms
• Note: switched use of w and v from Friedland (and course notes)

Extension: Moving Horizon Estimation

Key idea: estimate over a finite window in the past

$$\Phi_T = \min_{x_k, \{w_k\}_{k=0}^{T-N}} \left( \sum_{k=0}^{T-N-1} L_k(w_k, v_k) + \sum_{k=0}^{T-N} L_k(w_k, v_k) + \Gamma(x_0) \right)$$

$$= \min_{x_k, \{w_k\}_{k=0}^{T-N}} \left( \sum_{k=0}^{T-N} L_k(w_k, v_k) + Z_{T-N}(x) \right).$$

Example (Rao et al, 2003): nonlinear model with positive disturbances

$$x_{1,k+1} = 0.99x_{1,k} + 0.2x_{2,k}$$

$$x_{2,k+1} = -0.1x_{1,k} + \frac{0.5x_{2,k}}{1 + x_{2,k}^2} + w_k$$

$$y_k = x_{1,k} - 3x_{2,k} + v_k$$

• EKF handles nonlinearity, but assumes noise is zero mean => misses positive drift
Extension: Particle Filters

Sequential Monte Carlo

- Rough idea: keep track of many possible states of the system via individual “particles”
- Propagate each particle (state estimate + noise) via the system model with noise
- Truncate those particles that are particularly unlikely, redistribute weights

Remarks

- Can handle nonlinear, non-Gaussian processes
- Very computationally intensive; typically need to exploit problem structure
- Being explored in many application areas (eg, SLAM in robotics)
- Lots of current debate about information filters versus MHE versus particle filters

2007 Urban Challenge - 3 November 2007

Autonomous Urban Driving

- 60 mile course, less than 6 hours
- City streets, obeying traffic rules
- Follow cars, maintain safe distance
- Pull around stopped, moving vehicles
- Stop and go through intersections
- Navigate in parking lots (w/ other cars)
- U turns, traffic merges, replanning
- Prizes: $2M, $500K, $250K
Sensing and Decision Making

Video from 29 Jun 06 field test
- Front and side views from Tosin
- Rendered at 320x240, 15 Hz
- Manually synchronize

Some challenges
- Moving obstacle detection, separation, tracking and prediction
- Decision-making
- Lane markings (w/ shadows)

Sensing System

Sensing hardware
- 6 horizontal LADAR (overlapping)
- 1 pushbroom LADAR; 1 sweeping (PTU)
- 3 stereo pairs (color; 640x480 @ ~10 Hz)
- 2 road finding cameras (B&W)
- 2 RADAR units (PTU mounted)
- 10 blade cPCI high speed computing
Networked Control Systems

Next: effects of the network...