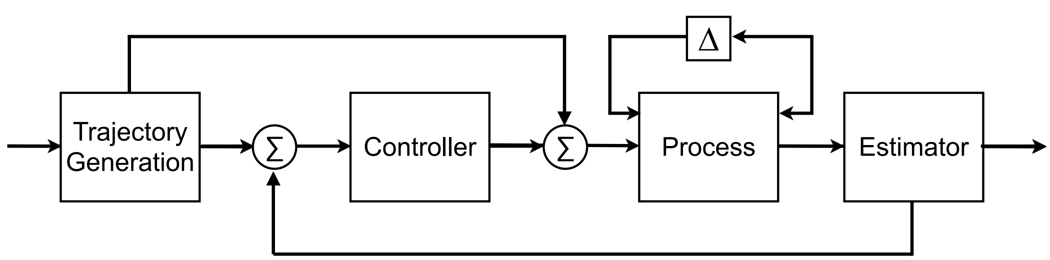
CDS 110b: Lecture 5-1
KF Extensions and Applications

Richard M. Murray
29 January 2007

Goals
- Review modern (optimization-based) control systems structure
- Describe nonlinear, non-Gaussian extensions to KF: MHE + particle filters
- Introduce course project possibilities (as applications of estimation)

Reading (optional)
- Cremean et al, “Alice: A Networked Control System for Autonomous Desert Driving”, *J. Field Robotics,* 2006. (Available at http://gc.caltech.edu)

Modern Control System Design

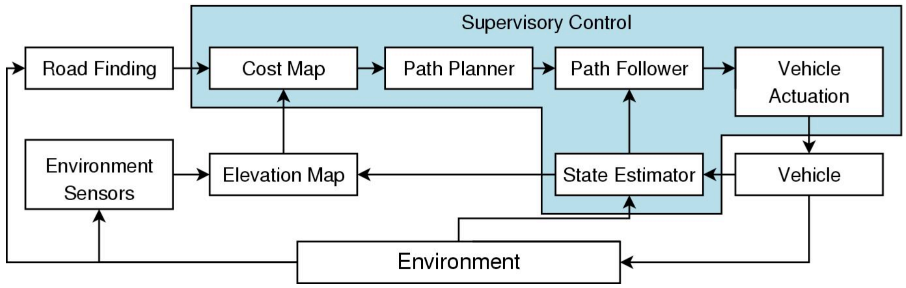


Traditional Control System: controller + process
- Corresponds to “inner loop” of most control system designs

Modern Control System: optimization-based design + robust analysis
- Replace reference with reference trajectory (Weeks 5-9)
- Replace process output with estimated output (Weeks 1-4)
- Replace “inner loop” controller with robust controller (Week 10 + CDS 212/213)

Cremean et al, 2006
J. Field Robotics

Example: Autonomous Driving



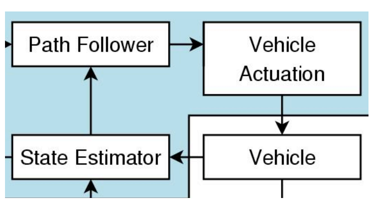
Computing

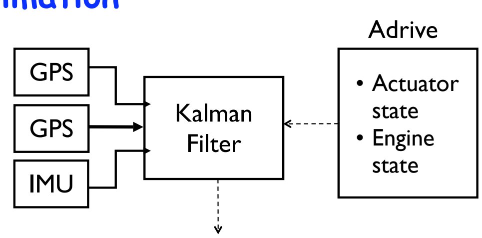
- 6 Dell 750 PowerEdge Servers (P4, 3GHz)
- 1 IBM Quad Core AMD64 (fast!)
- 1 Gb/s switched ethernet

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





Vehicle position, orientation, velocities, accelerations

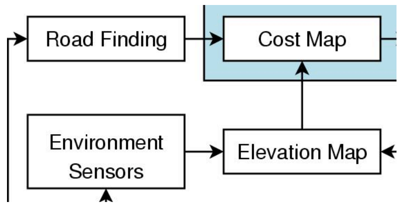
State estimation: astate

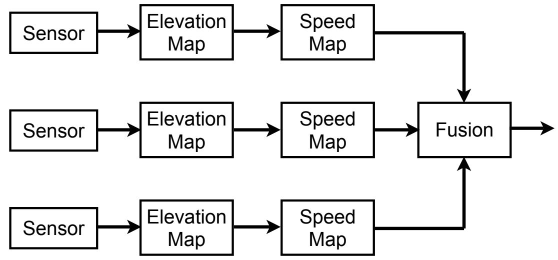
- Broadcast current vehicle state to all modules that require it (many)
- Timing of state signal is critical - use to calibrate sensor readings
- Quality of state estimate is critical: use to place terrain features in global map
- Issue: GPS jumps
  - Can get 20-100 cm jumps as satellites change positions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

Astate

- HW: 2 GPS units (2-10 Hz update), 1 inertial measurement unit (gryo, 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:

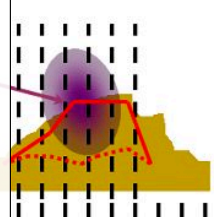
Propagation Equations:

$$\hat{z}_{i,j}(k+1|k) = \hat{z}_{i,j}(k|k)$$
$$P_{i,j}(k+1|k) = P_{i,j}(k|k)$$

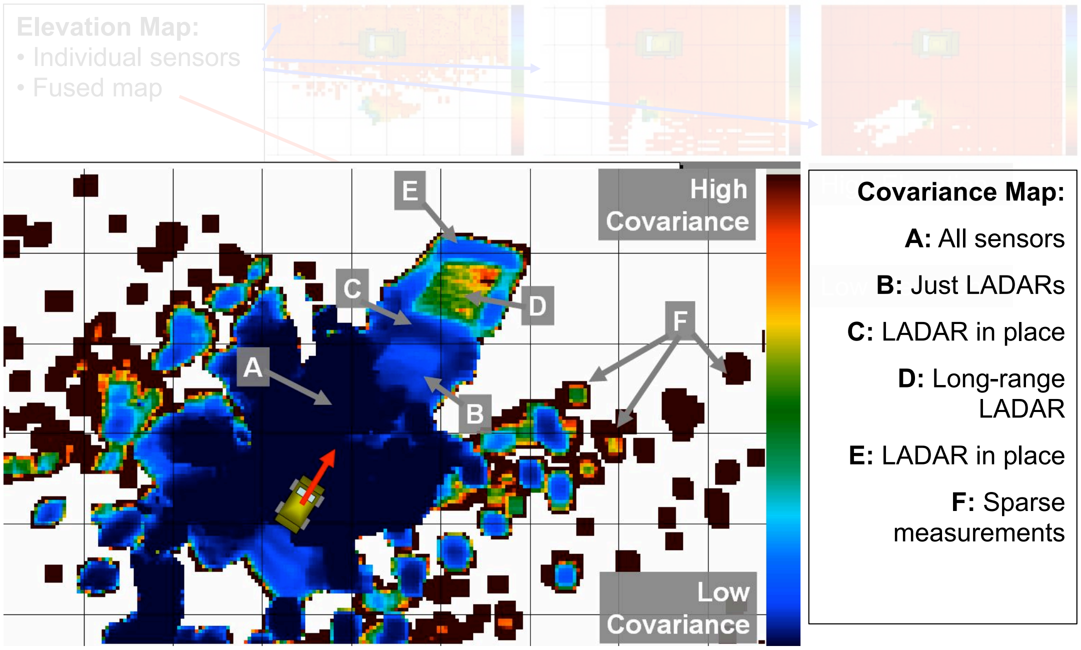
Update Equations:

$$\hat{z}_{i,j}(k+1|k+1) = \frac{R\hat{z}_{i,j}(k+1|k) + P_{i,j}(k+1|k)z_m}{P_{i,j}(k+1|k) + R}$$

$$P_{i,j}(k+1|k+1) = \frac{P_{i,j}(k+1|k)R}{P_{i,j}(k+1|k) + R}$$



The Results – Accurate Elevation and Covariance



Henrik Sandberg, 2005

Extension: Moving Horizon Estimation

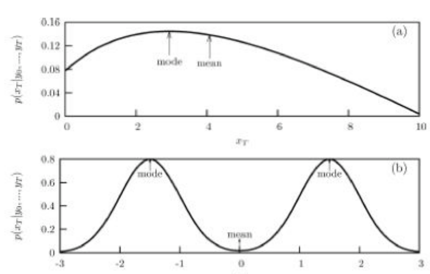
System description:

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

The problem: *Given* the data

$$Y_k = \{y_i : 0 \le i \le k\},$$

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.

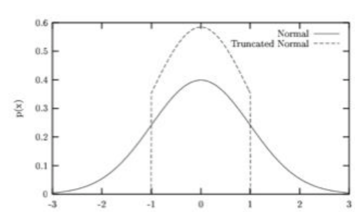


Pose as optimization problem:

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

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

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

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

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

Special case: Gaussian noise

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

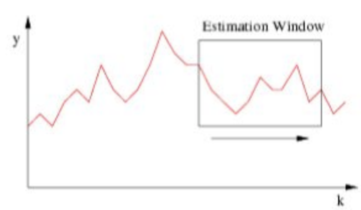
- Log of the probabilities sum of squares for noise terms
- Note: switched use of w and v from Friedland (and course notes)

Extension: Moving Horizon Estimation

Key idea: estimate over a finite window in the past

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

$$= \min_{z \in \mathcal{R}_{T-N}, \{w_k\}_{k=T-N}^{T-1}} \left(\sum_{k=T-N}^{T-1} L_k(w_k, v_k) + \mathcal{Z}_{T-N}(z) \right).$$



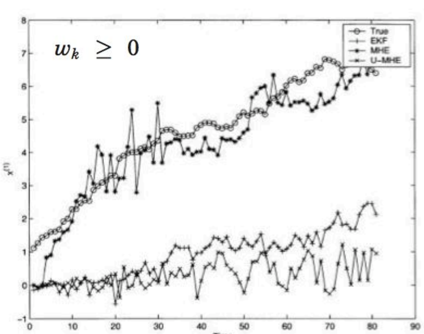
Example (Rao et al, 2003): nonlinear model with positive disturbances

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

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

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

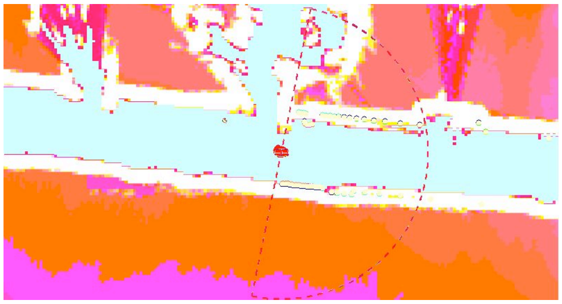
- EKF handles nonlinearity, but assumes noise is zero mean => misses positive drift

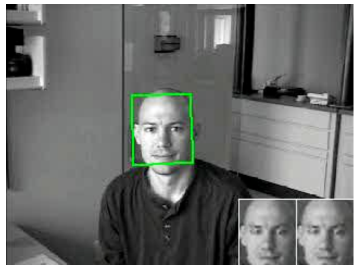


Extension: Particle Filters

Sequential Monte Carlo

- Rough idea: keep track of many possible states of the system via individual "particles"
- Propogate 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

Optional Course Project

Control System Implementation

- Course work focuses on design techniques, analysis, simulation
- Project will focus on implementation of estimators on Alice

Project administration

- Project reports (written and oral) in lieu of midterm and final
- Total time required for implementation: about 30-40 hours (over 10 weeks)
- Selected homework problems are aligned with project schedule
- Do ~1/2 of homework problems in second half of the course

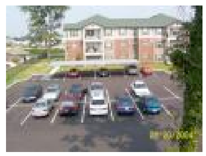
Sample projects

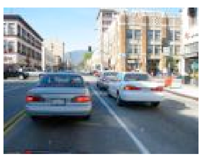
- Lane estimation and tracking using Kalman or particle filters
- Local versus global frame estimation (to cancel effects of GPS drifts, jumps)

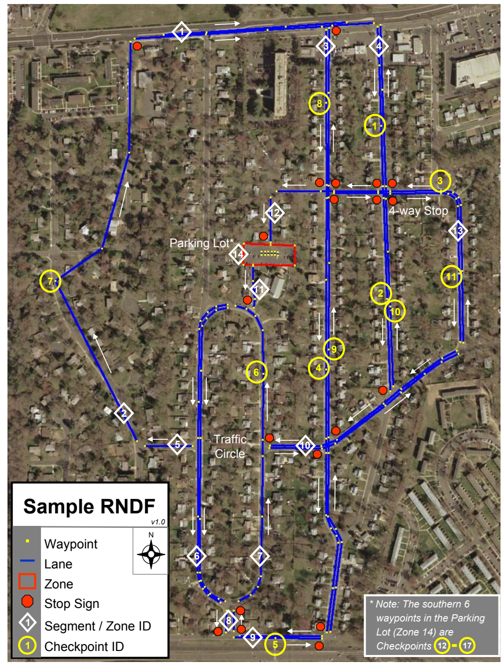
2007 Urban Challenge - 3 November 2007

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 synchronized

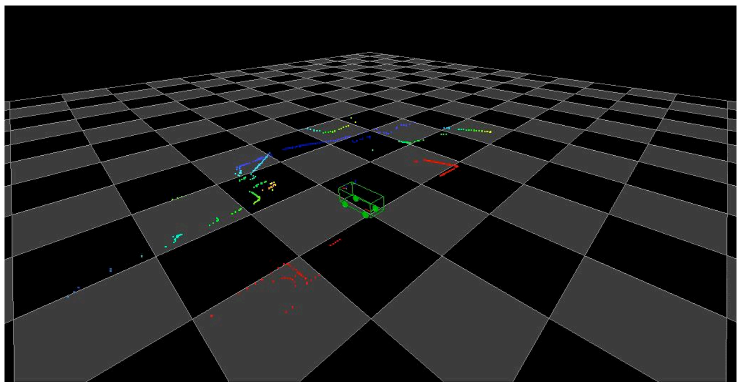
Some challenges

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

Project Idea: Terrain Estimation

Estimate the "terrain" (obstacles, curbs, speed bumps) around the vehicle

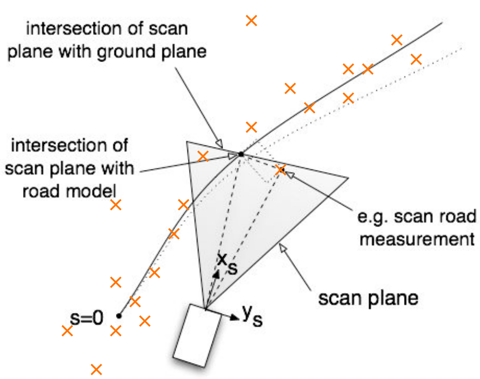
- Use data from multiple LADARs and stereo cameras
- Could be formulated as EKF (ala Gillula), MHE or particle filter
- Would like both estimated height as well as uncertainty estimates

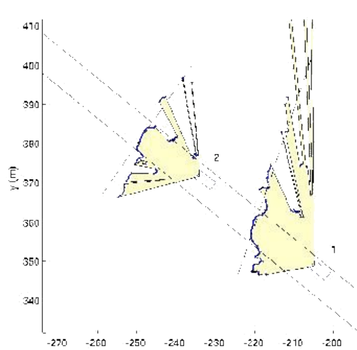


Project Idea: Lane Detection

Create estimate of road location based on measurements of lane features

- Measurements: locations of curbs, lane markings
- State: simple road model (eg, splines, clothoides)
- Starting point: Lars Cremean PhD thesis work (including initial code)
- EKF, IF, MHE, PF implementations possible



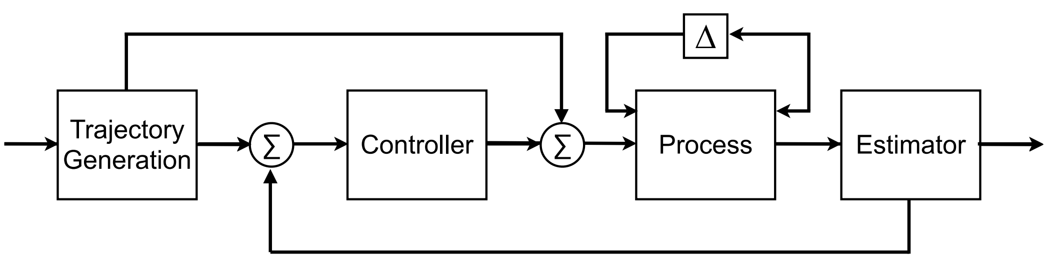


Project Idea: Moving Vehicle Tracking

Compute current and predicted position of a moving vehicle

- Assume measurements of current vehicle location (Laura Lindzey)
- Use model of vehicle motion, with uncertainty corresponding to driver
- Predict future states, with uncertainty according to model

Modern Control System Design



Traditional Control System: controller + process

- Corresponds to “inner loop” of most control system designs

Modern Control System: optimization-based design + robust analysis

- Replace reference with reference trajectory (Weeks 5-9)
- Replace process output with estimated output (Weeks 1-4)
- Replace “inner loop” controller with robust controller (Week 10 + CDS 212/213)