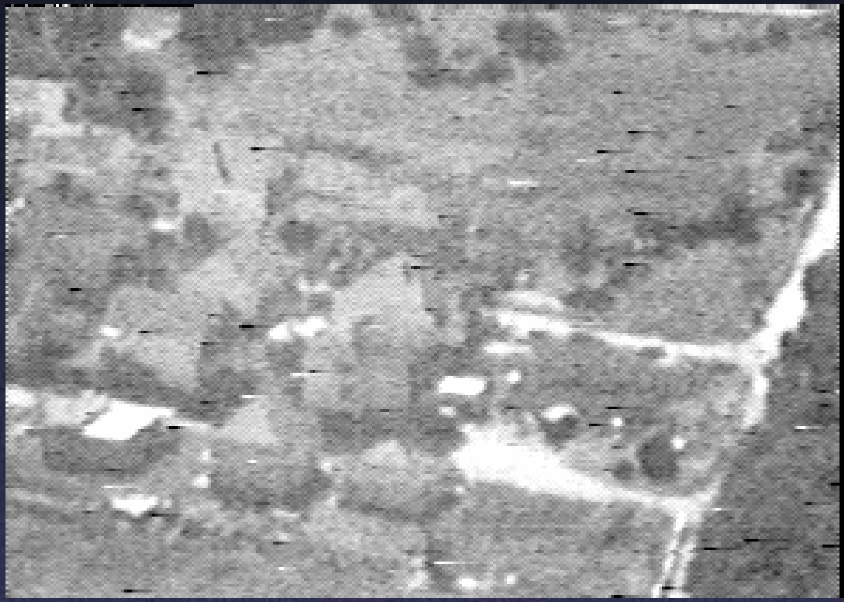
Image Alignment and Mosaicing
Feature Tracking and the Kalman Filter

Image Alignment Applications

- Local alignment:
  - Tracking
  - Stereo
- Global alignment:
  - Camera jitter elimination
  - Image enhancement
  - Panoramic mosaicing

Image Enhancement



Original

Enhanced

Anandan

Panoramic Mosaicing

Anandan

Correspondence Approaches

- Optical flow
- Correlation
- Correlation + optical flow
- Any of the above, iterated (e.g. Lucas-Kanade)
- Any of the above, coarse-to-fine

Correspondence Approaches

- Optical flow
- Correlation
- Correlation + optical flow
- Any of the above, iterated (e.g. Lucas-Kanade)
- Any of the above, coarse-to-fine

Optical Flow for Image Registration

- Compute local matches
- Least-squares fit to motion model
- Problem: outliers

Outlier Rejection

- Robust estimation: tolerant of outliers
- In general, methods based on absolute value rather than square:

$$\text{minimize } \Sigma|x_i - f|, \text{ not } \Sigma(x_i - f)^2$$

Correspondence Approaches

- Optical flow
- Correlation
- Correlation + optical flow
- Any of the above, iterated (e.g. Lucas-Kanade)
- Any of the above, coarse-to-fine

Correlation / Search Methods

- Assume translation only
- Given images I_1, I_2
- For each translation (t_x, t_y) compute

$$c(I_1, I_2, \mathbf{t}) = \sum_i \sum_j \psi\big(I_1(i,j), I_2(i - t_x, j - t_y)\big)$$

- Select translation that maximizes c
- Depending on window size, local or global

Cross-Correlation

- Statistical definition of correlation:

$$\psi(u,v) = uv$$

- Disadvantage: sensitive to local variations in image brightness

Normalized Cross-Correlation

- Normalize to eliminate brightness sensitivity:

$$\psi(u,v) = \frac{(u-\bar{u})(v-\bar{v})}{\sigma_u \sigma_v}$$

where

$$\bar{u} = \text{average}(u)$$

$$\sigma_u = \text{standard deviation}(u)$$

Sum of Squared Differences

- More intuitive measure:

$$\psi(u,v) = -(u-v)^2$$

- Negative sign so that higher values mean greater similarity
- Expand:

$$-(u-v)^2 = -u^2 - v^2 + 2uv$$

Local vs. Global

- Correlation with local windows not too expensive
- High cost if window size = whole image
- But computation looks like convolution
  - FFT to the rescue!

Correlation

$$c = \sum_i \sum_j I_1(i, j), I_2(i - \Delta_x, j - \Delta_y)$$

$$\mathcal{F}(c) = \mathcal{F}(I_1) \cdot \mathcal{F}_{\text{translated}}(I_2)$$

Fourier Transform with Translation

$$\mathcal{F}\left(f(x+\Delta x, y+\Delta y)\right) = \mathcal{F}\left(f(x,y)\right)e^{i(\omega_x \Delta x + \omega_y \Delta y)}$$

Fourier Transform with Translation

- Therefore, if I_1 and I_2 differ by translation,

$$\mathcal{F}\big(I_1(x,y)\big) = \mathcal{F}\big(I_2(x,y)\big)e^{i(\omega_x \Delta x + \omega_y \Delta y)}$$

$$e^{i(\omega_x \Delta x + \omega_y \Delta y)} = \frac{F_1}{F_2}$$

- So, $\mathcal{F}^{-1}(F_1/F_2)$ will have a peak at $(\Delta x, \Delta y)$

Phase Correlation

- In practice, use cross power spectrum

$$\frac{F_1 F_2^*}{\left|F_1 F_2^*\right|}$$

- Compute inverse FFT, look for peaks
- [Kuglin & Hines 1975]

Phase Correlation

- Advantages
  - Fast computation
  - Low sensitivity to global brightness changes (since equally sensitive to all frequencies)

Phase Correlation

- Disadvantages
  - Sensitive to white noise
  - No robust version
  - Translation only
    - Extensions to rotation, scale
    - But not local motion
    - Not too bad in practice with small local motions

Correspondence Approaches

- Optical flow
- Correlation
- Correlation + optical flow
- Any of the above, iterated (e.g. Lucas-Kanade)
- Any of the above, coarse-to-fine

Correlation plus Optical Flow

- Use e.g. phase correlation to find average translation (may be large)
- Use optical flow to find local motions

Correspondence Approaches

- Optical flow
- Correlation
- Correlation + optical flow
- Any of the above, iterated (e.g. Lucas-Kanade)
- Any of the above, coarse-to-fine

Correspondence Approaches

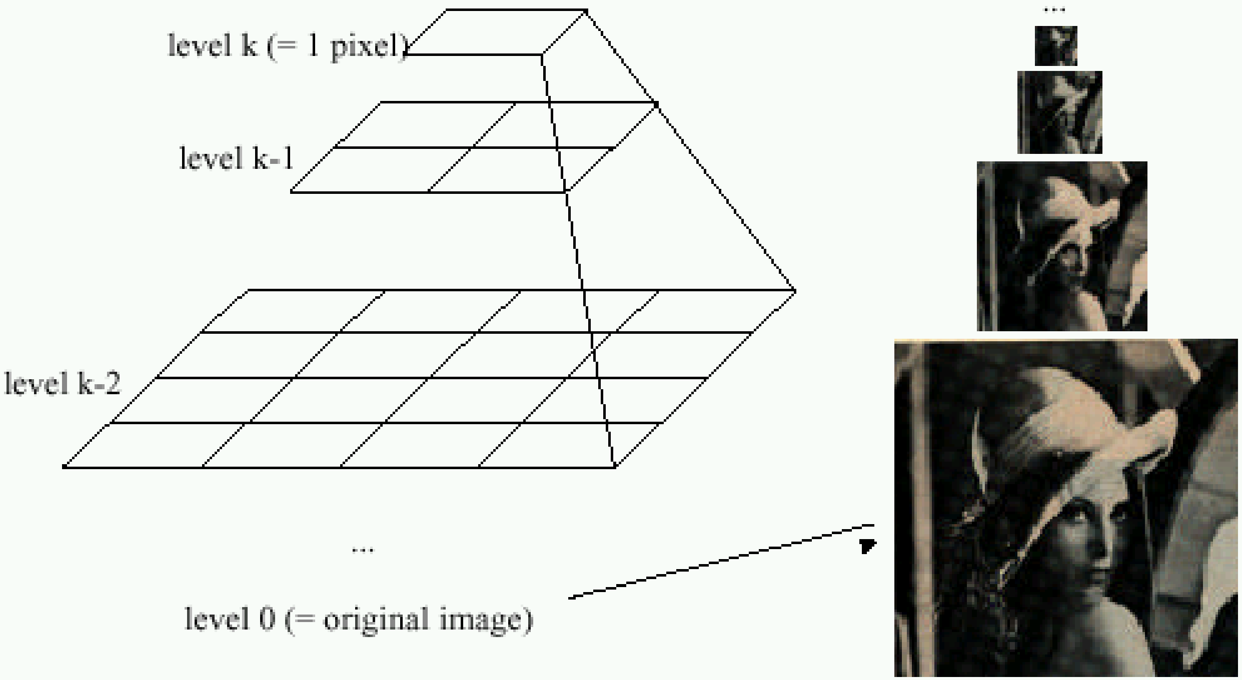
- Optical flow
- Correlation
- Correlation + optical flow
- Any of the above, iterated (e.g. Lucas-Kanade)
- Any of the above, coarse-to-fine

Image Pyramids

- Pre-filter images to collect information at different scales
- More efficient computation, allows larger motions

Image Pyramids

Idea: Represent NxN image as a “pyramid” of 1×1, 2×2, 4×4,…, $2^k\times2^k$ images (assuming $N=2^k$)



Szeliski

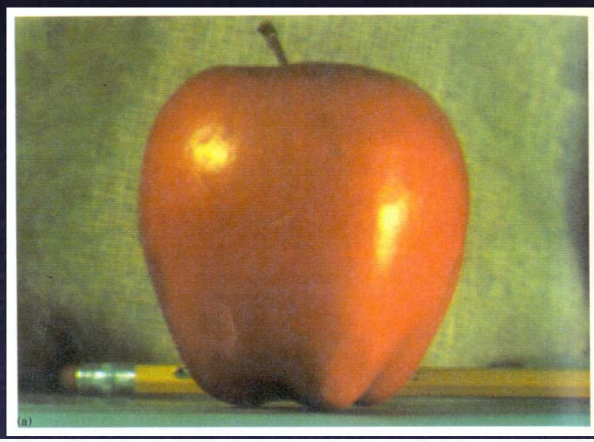
Blending

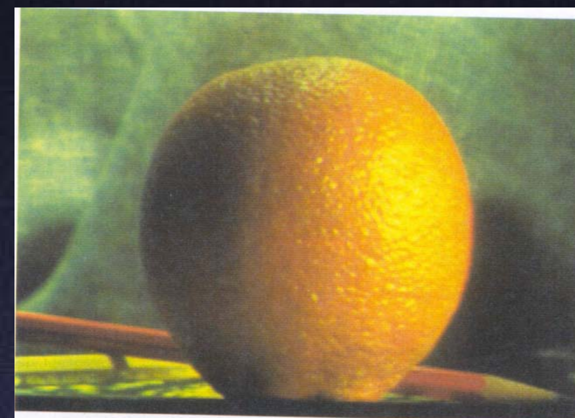
- Blend over too small a region: seams
- Blend over too large a region: ghosting

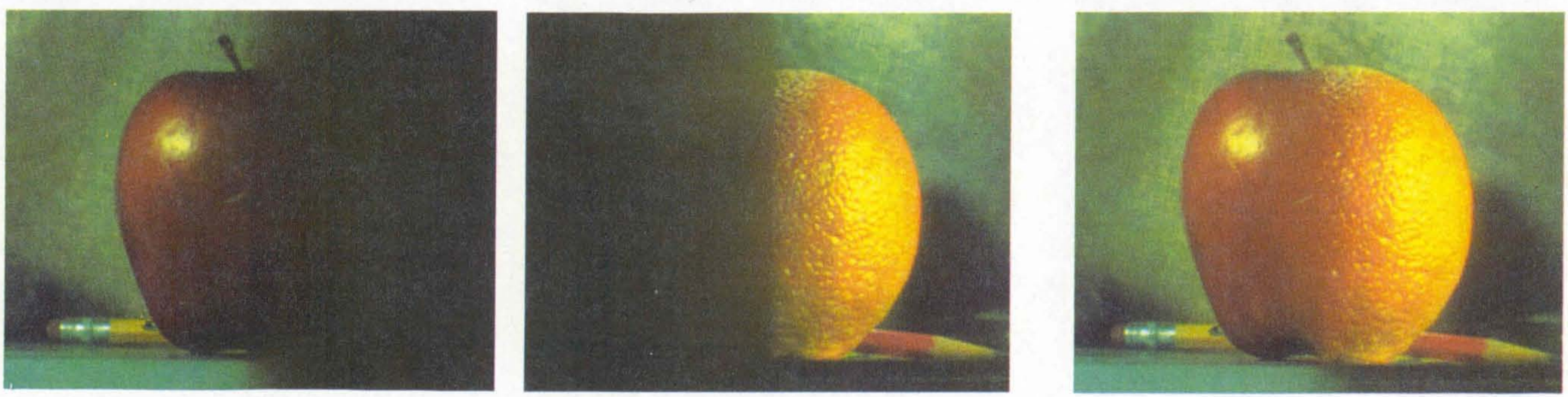
Multiresolution Blending

- Different blending regions for different levels in a pyramid [Burt & Adelson]
  - Blend low frequencies over large regions (minimize seams due to brightness variations)
  - Blend high frequencies over small regions (minimize ghosting)

Pyramid Blending







(d) (h) (l)

Szeliski

Minimum-Cost Cuts

- Instead of blending high frequencies along a straight line, blend along line of minimum differences in image intensities

Minimum-Cost Cuts

Moving object, simple blending → blur

[Davis 98]

Minimum-Cost Cuts

Minimum-cost cut → no blur

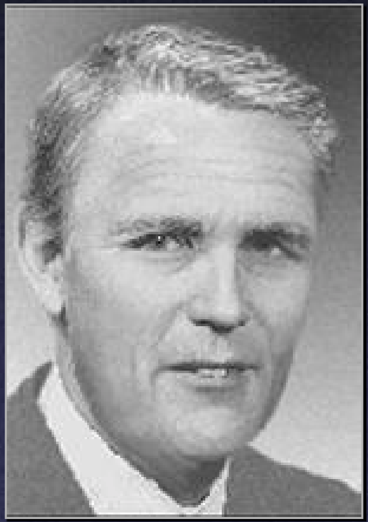
[Davis 98]

Feature Tracking

- Local region
- Take advantage of many frames
  - Prediction, uncertainty estimation
  - Noise filtering
  - Handle short occlusions

Kalman Filtering

- Assume that results of experiment (i.e., optical flow) are noisy measurements of system state
- Model of how system evolves
- Optimal combination of system model and observations
- Prediction / correction framework



Rudolf Emil Kalman

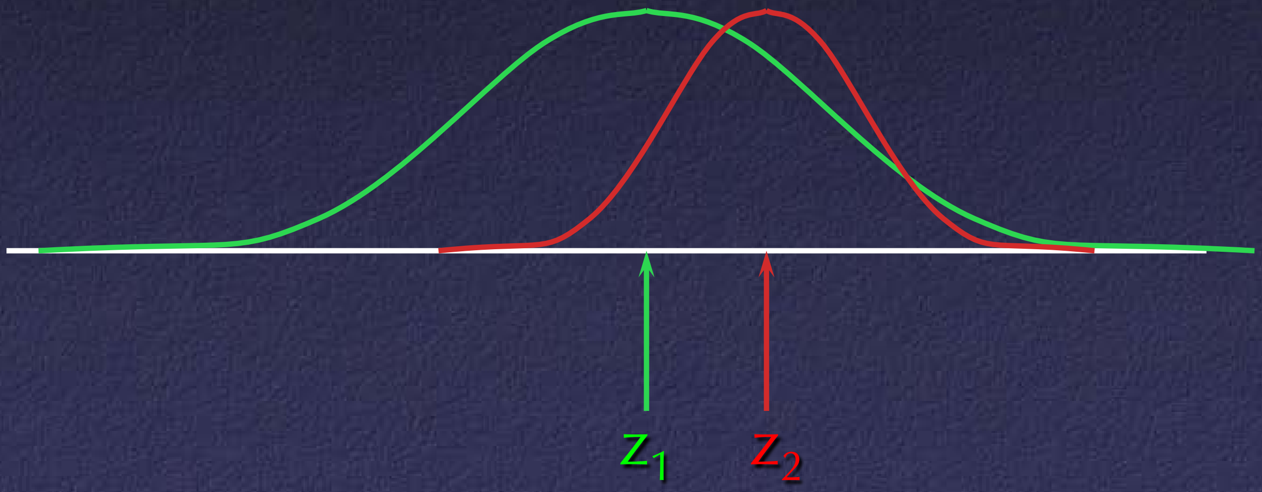
Acknowledgment: much of the following material is based on the SIGGRAPH 2001 course by Greg Welch and Gary Bishop (UNC)

Simple Example

- Measurement of a single point z_1
- Variance $\sigma_1{}^2$ (uncertainty σ_1)
  - Assuming Gaussian distribution
- Best estimate of true position $\hat{x}_1 = z_1$
- Uncertainty in best estimate $\hat{\sigma}_1^2 = \sigma_1^2$

Simple Example

- Second measurement z_2, variance σ_2^2
- Best estimate of true position?



Simple Example

- Second measurement z_2, variance $\sigma_2{}^2$
- Best estimate of true position: weighted average

$$\hat{x}_2 = \frac{\frac{1}{\sigma_1^2} z_1 + \frac{1}{\sigma_2^2} z_2}{\frac{1}{\sigma_1^2} + \frac{1}{\sigma_2^2}}$$

$$= \hat{x}_1 + \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2}\left(z_2 - \hat{x}_1\right)$$

- Uncertainty in best estimate $\hat{\sigma}_2^2 = \dfrac{1}{\frac{1}{\hat{\sigma}_1^2} + \frac{1}{\sigma_2^2}}$

Online Weighted Average

- Combine successive measurements into constantly-improving estimate
- Uncertainty decreases over time
- Only need to keep current measurement, last estimate of state and uncertainty

Terminology

- In this example, position is *state* (in general, any vector)
- State can be assumed to evolve over time according to a *system model* or *process model* (in this example, “nothing changes”)
- Measurements (possibly incomplete, possibly noisy) according to a *measurement model*
- Best estimate of state $\hat{x}$ with covariance P

Linear Models

- For "standard" Kalman filtering, everything must be linear
- System model:

$$x_k = \Phi_{k-1} x_{k-1} + \xi_{k-1}$$

- The matrix Φ_k is *state transition matrix*
- The vector ξ_k represents *additive noise*, assumed to have covariance Q

Linear Models

- Measurement model:

$$z_k = H_k x_k + \mu_k$$

- Matrix H is *measurement matrix*
- The vector μ is *measurement noise,* assumed to have covariance R

PV Model

- Suppose we wish to incorporate velocity

$$\mathbf{x}_k = \begin{bmatrix} x \\ dx/dt \end{bmatrix}$$

$$\Phi_k = \begin{bmatrix} 1 & \Delta t_k \\ 0 & 1 \end{bmatrix}$$

$$H = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

Prediction/Correction

- Predict new state

$$x'_k = \Phi_{k-1}\hat{x}_{k-1}$$

$$P'_k = \Phi_{k-1}P_{k-1}\Phi^{\mathrm{T}}_{k-1} + Q_{k-1}$$

- Correct to take new measurements into account

$$\hat{x}_k = x'_k + K_k\left(z_k - H_k x'_k\right)$$

$$P_k = \left(I - K_k H_k\right)P'_k$$

Kalman Gain

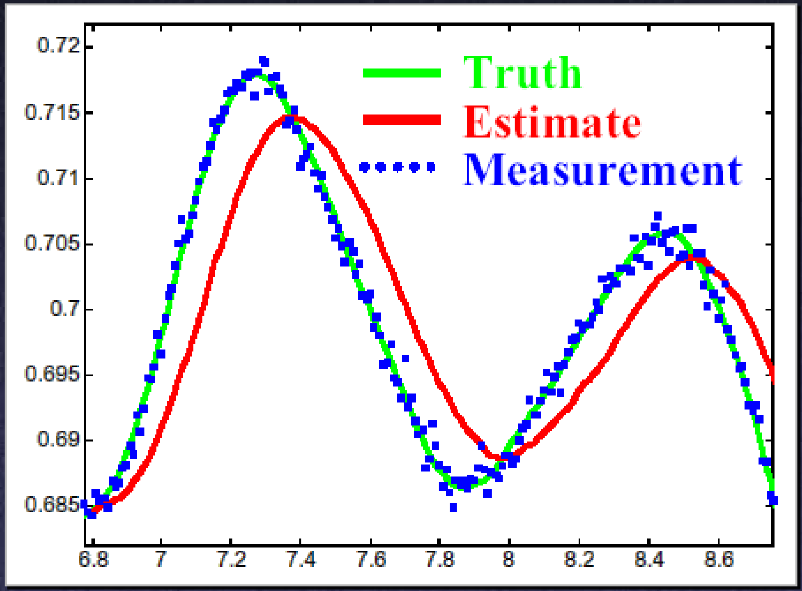
- Weighting of process model vs. measurements

$$K_k = P'_k H_k^{\mathrm{T}} \left(H_k P'_k H_k^{\mathrm{T}} + R_k \right)^{-1}$$

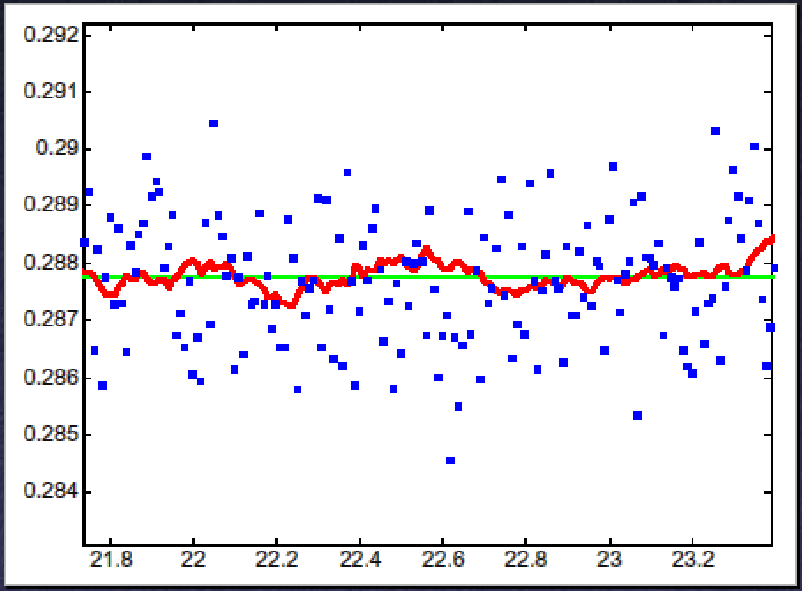
- Compare to what we saw earlier:

$$\frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2}$$

Results: Position-Only Model



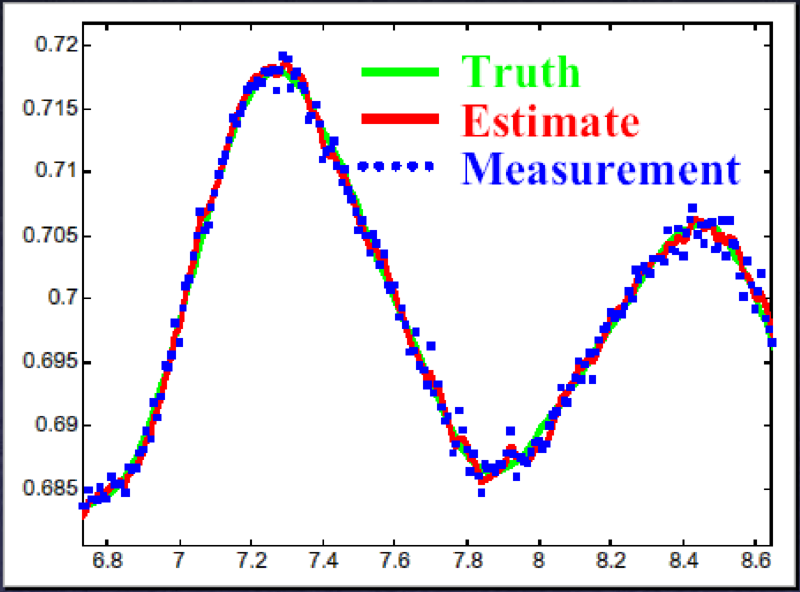
Moving



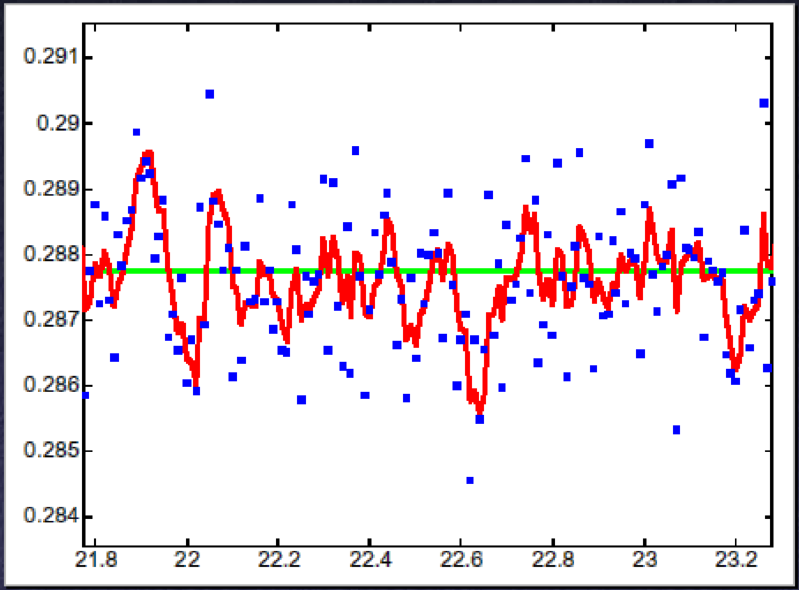
Still

[Welch & Bishop]

Results: Position-Velocity Model



Moving



Still

[Welch & Bishop]

Extension: Multiple Models

- Simultaneously run many KFs with different system models
- Estimate probability each KF is correct
- Final estimate: weighted average

Probability Estimation

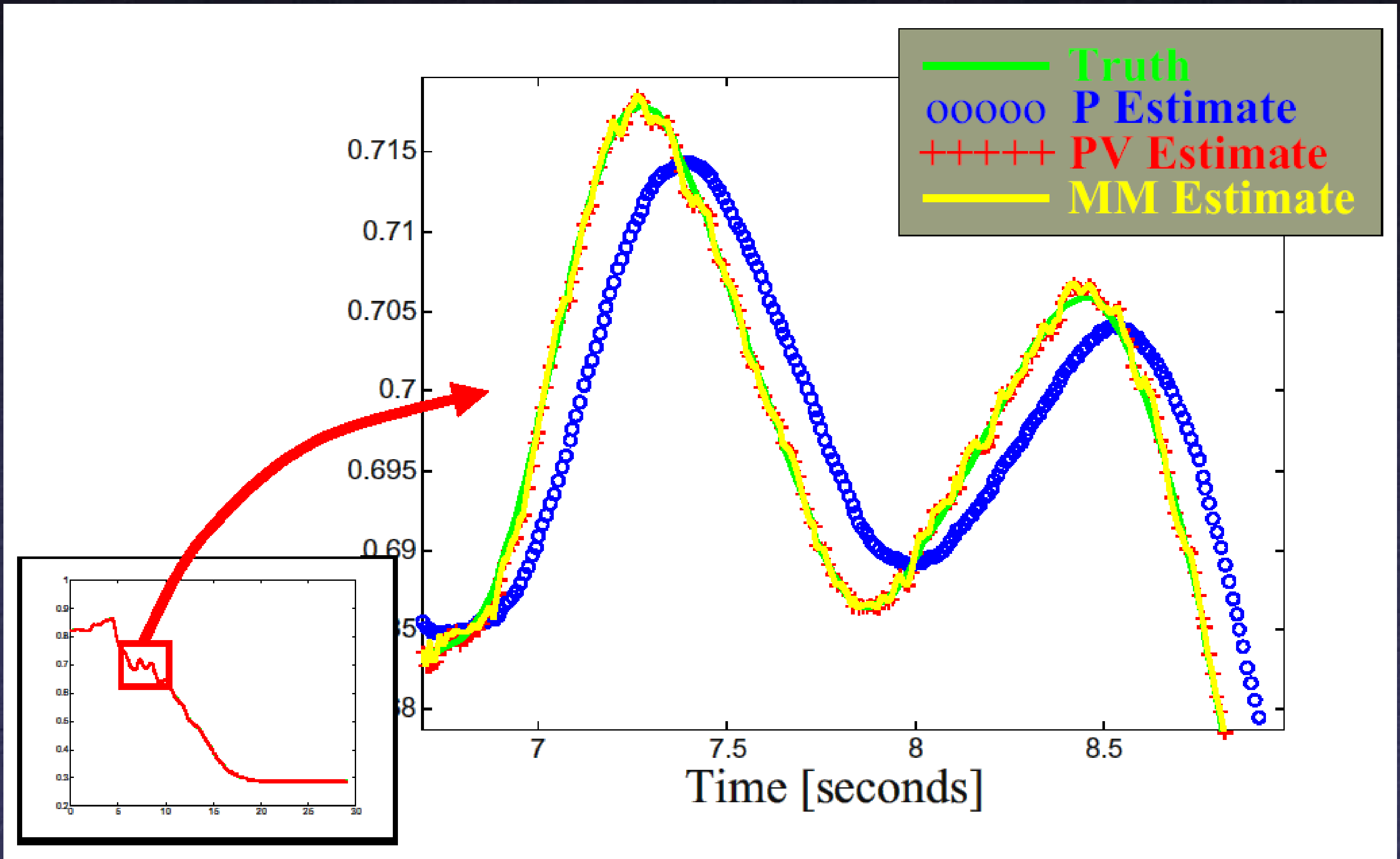
- Given some Kalman filter, the probability of a measurement z_k is just *n*-dimensional Gaussian

$$p = \frac{1}{\left(2\pi\,|\,C\,|\right)^{n/2}} e^{-\frac{1}{2}\left(z_k - H_k x'_k\right)^{\mathrm{T}} C^{-1} \left(z_k - H_k x'_k\right)^{\mathrm{T}}}$$

where

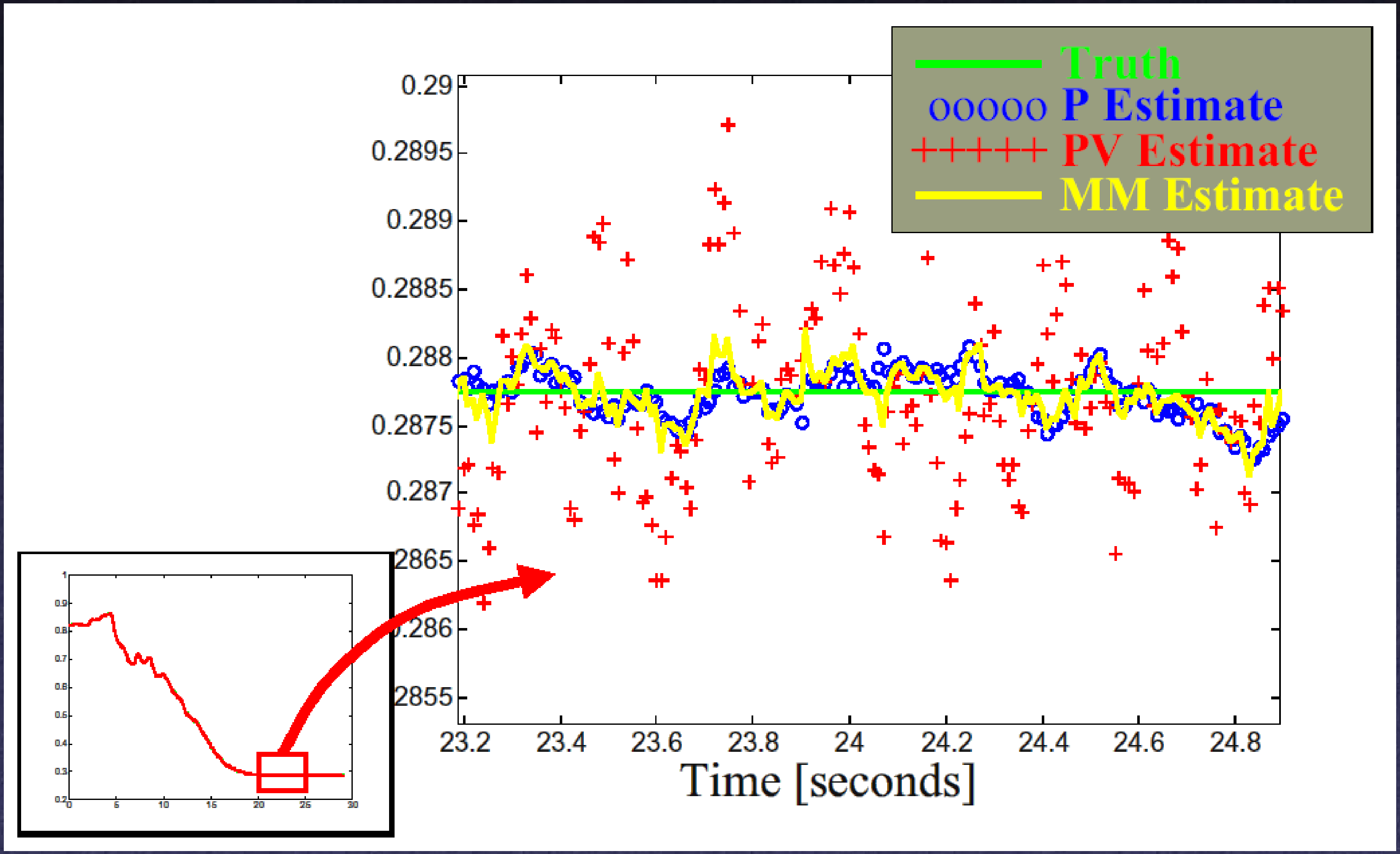
$$C = HPH^{\mathrm{T}} + R$$

Results: Multiple Models



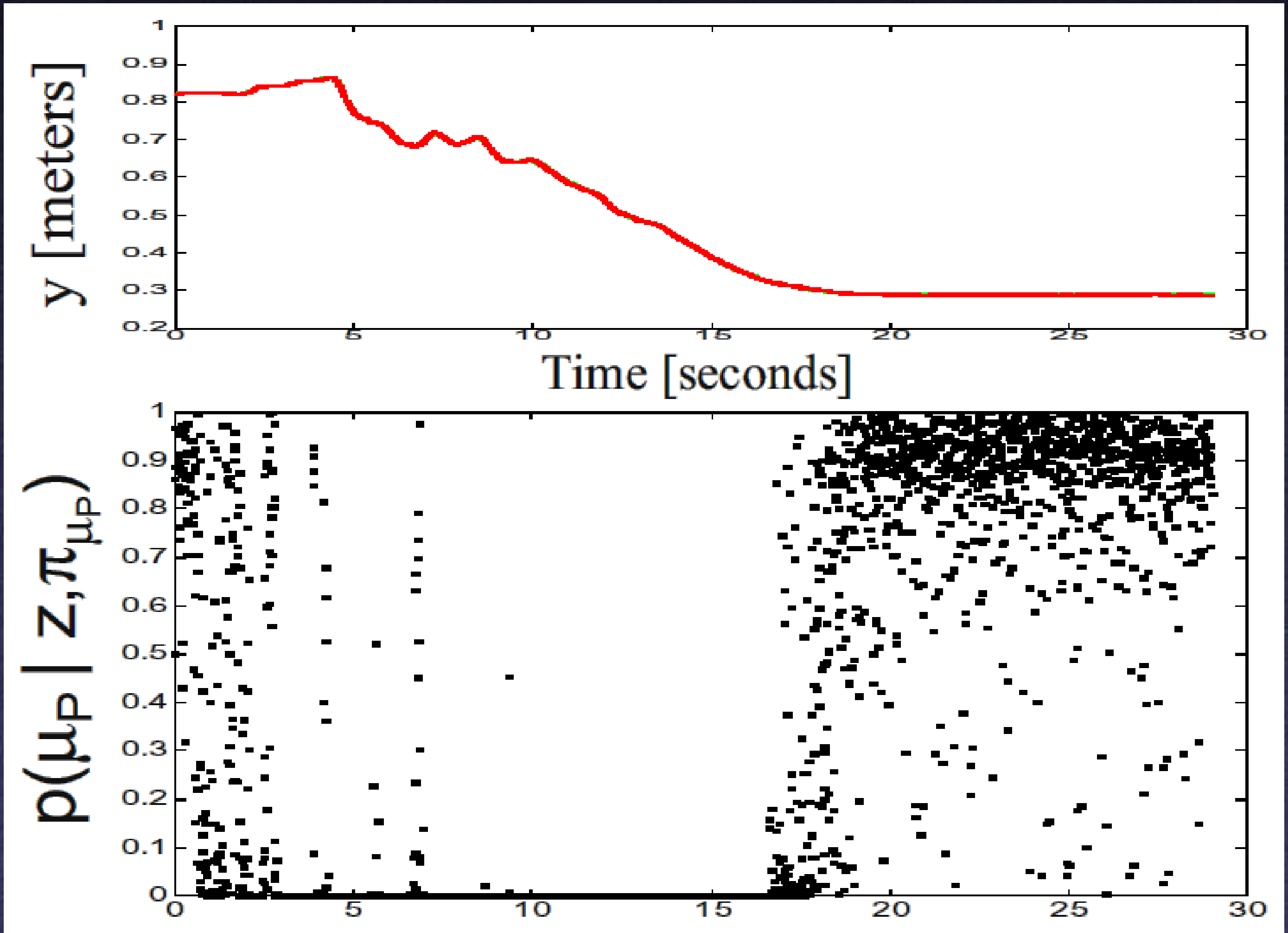
[Welch & Bishop]

Results: Multiple Models



[Welch & Bishop]

Results: Multiple Models

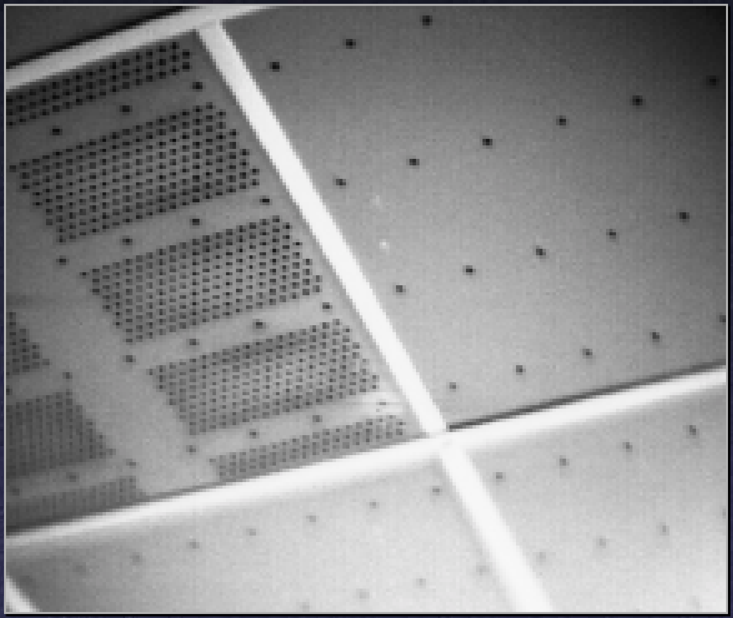


[Welch & Bishop]

Extension: SCAAT

- H can be different at different time steps
  - Different sensors, types of measurements
  - Sometimes measure only part of state
- Single Constraint At A Time (SCAAT)
  - Incorporate results from one sensor at once
  - Alternative: wait until you have measurements from enough sensors to know complete state (MCAAT)
  - MCAAT equations often more complex, but sometimes necessary for initialization

UNC HiBall



- 6 cameras, looking at LEDs on ceiling
- LEDs flash over time

[Welch & Bishop]

Extension: Nonlinearity (EKF)

- HiBall state model has nonlinear degrees of freedom (rotations)
- Extended Kalman Filter allows nonlinearities by:
  - Using general functions instead of matrices
  - Linearizing functions to project forward
  - Like 1st order Taylor series expansion
  - Only have to evaluate Jacobians (partial derivatives), not invert process/measurement functions

Other Extensions

- On-line noise estimation
- Using known system input (e.g. actuators)
- Using information from both past and future
- Non-Gaussian noise and particle filtering