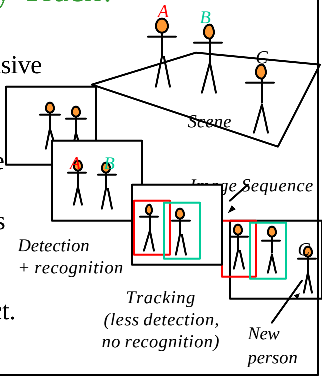
Tracking

Definition of Tracking

- Tracking:
  - Generate some conclusions about the motion of the scene, objects, or the camera, given a sequence of images.
  - Knowing this motion, predict where things are going to project in the next image, so that we don't have so much work looking for them.

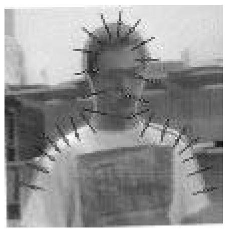
Why Track?

- Detection and recognition are expensive
- If we get an idea of where an object is in the image because we have an idea of the motion from previous images, we need less work detecting or recognizing the object.



Tracking a Silhouette by Measuring Edge Positions

- Observations are positions of edges along normals to tracked contour



Why not Wait and Process the Set of Images as a Batch?

- In a car system, detecting and tracking pedestrians in real time is important.
- Recursive methods require less computing

Implicit Assumptions of Tracking

- Physical cameras do not move instantly from a viewpoint to another
- Object do not teleport between places around the scene
- Relative position between camera and scene changes incrementally
- We can model motion

Related Fields

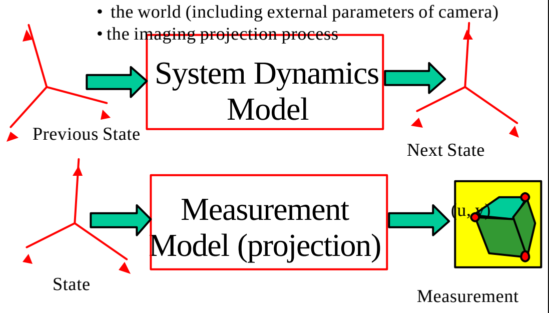
- Signal Detection and Estimation
- Radar technology

The Problem: Signal Estimation

- We have a system with parameters
  - Scene structure, camera motion, automatic zoom
  - System state is unknown (“hidden”)
- We have measurements
  - Components of stable “feature points” in the images.
  - “Observations”, projections of the state.
- We want to recover the state components from the observations

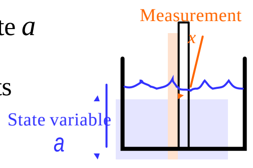
Necessary Models

- We use models to describe a priori knowledge about
  - the world (including external parameters of camera)
  - the imaging projection process



A Simple Example of Estimation by Least Square Method

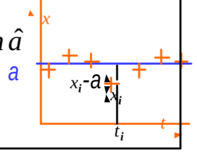
- Goal: Find estimate $\hat{a}$ of state a such that the least square error between measurements and the state is minimum



$$C = \frac{1}{2}\sum_{i=1}^{n}(x_i - a)^2$$

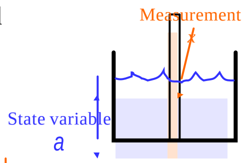
$$\frac{\partial C}{\partial a} = 0 = \sum_{i=1}^{n}(x_i - \hat{a}) = \sum_{i=1}^{n} x_i - n\hat{a}$$

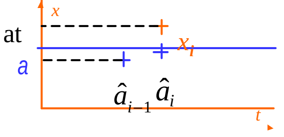
$$\hat{a} = \frac{1}{n}\sum_{i=1}^{n} x_i$$



Recursive Least Square Estimation

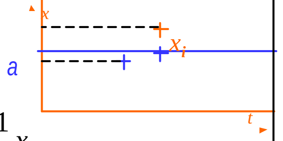
- We don’t want to wait until all data have been collected to get an estimate $\hat{a}$ of the depth
- We don’t want to reprocess old data when we make a new measurement
- Recursive method: data at step i are obtained from data at step i - 1





Recursive Least Square Estimation 2

- Recursive method: data at step i are obtained from data at step i - 1



$$\hat{a}_i = \frac{1}{i}\sum_{k=1}^{i} x_k = \frac{1}{i}\sum_{k=1}^{i-1} x_k + \frac{1}{i}x_i \qquad \hat{a}_{i-1} = \frac{1}{i-1}\sum_{k=1}^{i-1} x_k$$

$$\hat{a}_i = \frac{i-1}{i}\hat{a}_{i-1} + \frac{1}{i}x_i$$

$$\boxed{\hat{a}_i = \hat{a}_{i-1} + \frac{1}{i}(x_i - \hat{a}_{i-1})}$$

Recursive Least Square Estimation 3

Gain

Actual measure

Predicted measure

$$\hat{a}_i = \hat{a}_{i-1} + \frac{1}{i}\underbrace{(x_i - \hat{a}_{i-1})}_{\text{Innovation}}$$

Estimate at step *i*

Gain specifies how much
do we pay attention
to the difference
between what we expected
and what we actually get

Least Square Estimation of the State Vector of a Static System

1. Batch method

$$x_i = \mathbf{H}_i \cdot \mathbf{a} \quad \Rightarrow \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{bmatrix} = \begin{bmatrix} \mathbf{H}_1 \\ \mathbf{H}_2 \\ \dots \\ \mathbf{H}_n \end{bmatrix} \mathbf{a}$$

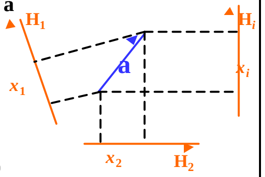
measurement equation

$$\Rightarrow \mathbf{x} = \mathbf{H\,a}$$

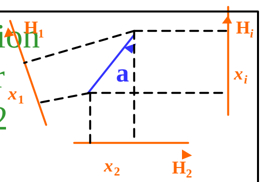
Find estimate $\hat{\mathbf{a}}$ that minimizes

$$C = \frac{1}{2}(\mathbf{X} - \mathbf{H\,a})^{\mathbf{T}}(\mathbf{X} - \mathbf{H\,a})$$

We find $\hat{\mathbf{a}} = (\mathbf{H}^{\mathbf{T}}\mathbf{H})^{-1}\mathbf{H}^{\mathbf{T}}\mathbf{X}$



Least Square Estimation of the State Vector of a Static System 2



2. Recursive method

Calculation is similar to calculation of running average

We had: $\hat{a}_i = \hat{a}_{i-1} + \frac{1}{i}(x_i - \hat{a}_{i-1})$

Predicted measure

Now we find: $\hat{\mathbf{a}}_\mathbf{i} = \hat{\mathbf{a}}_\mathbf{i-1} + \mathbf{K}_\mathbf{i}(\mathbf{X}_\mathbf{i} - \mathbf{H}_\mathbf{i}\hat{\mathbf{a}}_\mathbf{i-1})$

Gain matrix

Innovation

with $\mathbf{K}_\mathbf{i} = \mathbf{P}_\mathbf{i}\,\mathbf{H}_\mathbf{i}^{\mathbf{T}}$

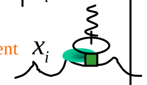
$$\mathbf{P}_\mathbf{i} = (\mathbf{H}_\mathbf{i}\,\mathbf{H}_\mathbf{i}^{\mathbf{T}})^{-1}$$

Dynamic System

$$X_i = X_{i-1} + V_{i-1}\,\Delta t$$
$$V_i = V_{i-1} + A_{i-1}\,\Delta t$$
$$A_i = A_{i-1} + w$$

State of rocket $\mathbf{a}_i \begin{vmatrix} X_i \\ V_i \\ A_i \end{vmatrix}$

Measurement x_i



$$\begin{bmatrix} X_i \\ V_i \\ A_i \end{bmatrix} = \begin{bmatrix} 1 & \Delta t & 0 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_{i-1} \\ V_{i-1} \\ A_{i-1} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ w \end{bmatrix} \Rightarrow \mathbf{a}_\mathbf{i} = \Phi\,\mathbf{a}_\mathbf{i-1} + \mathbf{w}$$

Tweak factor

State equation for rocket

$$x_i = [1 \quad 0 \quad 0]\begin{bmatrix} X_i \\ V_i \\ A_i \end{bmatrix} + V \quad \Rightarrow x_i = \mathbf{H\,a}_\mathbf{i} + V$$

Noise

Measurement equation

Recursive Least Square Estimation for a Dynamic System (Kalman Filter)

State equation

Tweak factor for model

$$\mathbf{a}_\mathbf{i} = \Phi_\mathbf{i}\;\mathbf{a}_\mathbf{i-1} + \mathbf{w}_\mathbf{i-1} \qquad \mathbf{w}_\mathbf{i} \sim N(0, \mathbf{Q}_\mathbf{i})$$

Measurement equation

Measurement noise

$$\mathbf{x}_\mathbf{i} = \mathbf{H}_\mathbf{i}\,\mathbf{a}_\mathbf{i} + \mathbf{v}_\mathbf{i} \qquad \mathbf{v}_\mathbf{i} \sim N(0, \mathbf{R}_\mathbf{i})$$

Prediction for a_i

$$\hat{\mathbf{a}}_\mathbf{i} = \Phi_\mathbf{i}\,\hat{\mathbf{a}}_\mathbf{i-1} + \mathbf{K}_\mathbf{i}(\mathbf{x}_\mathbf{i} - \mathbf{H}_\mathbf{i}\Phi_\mathbf{i}\hat{\mathbf{a}}_\mathbf{i-1})$$

Prediction for $\mathbf{x}_\mathbf{i}$

$$\mathbf{K}_\mathbf{i} = \mathbf{P'}_\mathbf{i}\,\mathbf{H}_\mathbf{i}^{\mathbf{T}}(\mathbf{H}_\mathbf{i}\,\mathbf{P'}_\mathbf{i}\,\mathbf{H}_\mathbf{i}^{\mathbf{T}} + \mathbf{R}_\mathbf{i})^{-1}$$

Gain

$$\mathbf{P'}_\mathbf{i} = \Phi_\mathbf{i}\mathbf{P}_\mathbf{i-1}\,\Phi_\mathbf{i}^{\mathbf{T}} + \mathbf{Q}_\mathbf{i-1}$$

Covariance matrix for prediction error

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

Covariance for estimation error

Estimation when System Model is Nonlinear (Extended Kalman Filter)

State equation

Measurement equation

$$\mathbf{a}_\mathbf{i} = \mathbf{f}(\mathbf{a}_\mathbf{i-1}) + \mathbf{w}_\mathbf{i-1} \qquad \mathbf{x}_\mathbf{i} = \mathbf{H}_\mathbf{i}\,\mathbf{a}_\mathbf{i} + \mathbf{v}_\mathbf{i}$$

$$\hat{\mathbf{a}}_\mathbf{i} = \mathbf{f}(\hat{\mathbf{a}}_\mathbf{i-1}) + \mathbf{K}_\mathbf{i}(\mathbf{x}_\mathbf{i} - \mathbf{H}_\mathbf{i}\,\mathbf{f}(\hat{\mathbf{a}}_\mathbf{i-1}))$$

$$\mathbf{K}_\mathbf{i} = \mathbf{P'}_\mathbf{i}\,\mathbf{H}_\mathbf{i}^{\mathbf{T}}(\mathbf{H}_\mathbf{i}\,\mathbf{P'}_\mathbf{i}\,\mathbf{H}_\mathbf{i}^{\mathbf{T}} + \mathbf{R}_\mathbf{i})^{-1}$$

$$\mathbf{P'}_\mathbf{i} = \frac{\partial \mathbf{f}}{\partial \mathbf{a}_\mathbf{i}}\mathbf{P}_\mathbf{i-1}\frac{\partial \mathbf{f}}{\partial \mathbf{a}_\mathbf{i}}^{\mathbf{T}} + \mathbf{Q}_\mathbf{i-1}$$

Jacobian Matrix

Differences compared to regular Kalman filter are circled in red

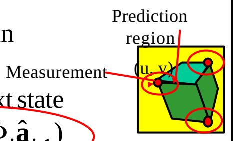
$$\mathbf{P}_\mathbf{i-1} = (\mathbf{I} - \mathbf{K}_\mathbf{i-1}\mathbf{H}_\mathbf{i-1})\mathbf{P'}_\mathbf{i-1}$$

Tracking Steps

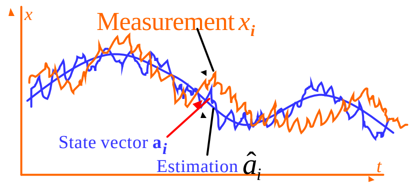
- Predict next state as $\Phi_i \hat{\mathbf{a}}_{i\text{-}1}$ using previous step and dynamic model
- Predict regions $N(\mathbf{H}_i \Phi_i \hat{\mathbf{a}}_{i\text{-}1}, \mathbf{P'}_i)$ of next measurements using measurement model and uncertainties
- Make new measurements $\mathbf{x}_i$ in predicted regions
- Compute best estimate of next state

$$\hat{\mathbf{a}}_i = \Phi_i \hat{\mathbf{a}}_{i\text{-}1} + \mathbf{K}_i(\mathbf{x}_i - \mathbf{H}_i \Phi_i \hat{\mathbf{a}}_{i\text{-}1})$$

"Correction" of predicted state



Recursive Least Square Estimation for a Dynamic System (Kalman Filter)



Tracking as a Probabilistic Inference Problem

- Find distributions for state vector $\mathbf{a}_i$ and for measurement vector $\mathbf{x}_i$. Then we are able to compute the expectations $\hat{\mathbf{a}}_i$ and $\hat{\mathbf{x}}_i$
- Simplifying assumptions (same as for HMM)

$$P(\mathbf{a}_i \mid \mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_{i\text{-}1}) = P(\mathbf{a}_i \mid \mathbf{a}_{i\text{-}1})$$

(Only immediate past matters)

$$P(\mathbf{x}_i, \mathbf{x}_j, \ldots, \mid \mathbf{a}_i) = P(\mathbf{x}_i \mid \mathbf{a}_i) P(\mathbf{x}_j \mid \mathbf{a}_i) \ldots P(\mathbf{x}_k \mid \mathbf{a}_i)$$

(Conditional independence of measurements given a state)

Tracking as Inference

- Prediction

$$P(\mathbf{a}_i \mid \mathbf{x}_1, \cdots, \mathbf{x}_{i\text{-}1}) = \int P(\mathbf{a}_i \mid \mathbf{a}_{i\text{-}1}) P(\mathbf{a}_{i\text{-}1} \mid \mathbf{x}_1, \cdots, \mathbf{x}_{i\text{-}1}) d\mathbf{a}_{i\text{-}1}$$

- Correction

$$P(\mathbf{a}_i \mid \mathbf{x}_1, \cdots, \mathbf{x}_i) = \frac{P(\mathbf{x}_i \mid \mathbf{a}_i) P(\mathbf{a}_i \mid \mathbf{x}_1, \cdots, \mathbf{x}_{i\text{-}1})}{\int P(\mathbf{x}_i \mid \mathbf{a}_i) P(\mathbf{a}_i \mid \mathbf{x}_1, \cdots, \mathbf{x}_{i\text{-}1}) d\mathbf{a}_i}$$

- Produces same results as least square approach if distributions are Gaussians: Kalman filter
- See Forsyth and Ponce, Ch. 19

Kalman Filter for 1D Signals

State equation

$$a_i = f\, a_{i\text{-}1} + w_{i\text{-}1} \qquad w_i \sim N(0, q)$$

Tweak factor for model

Measurement equation

$$x_i = h\, a_i + v_i \qquad v_i \sim N(0, r)$$

Measurement noise

Prediction for $\mathrm{a_i}$ (*a priori* estimate)

$$\hat{a}_i = f\, \hat{a}_{i\text{-}1} + K_i(x_i - h\, f\, \hat{a}_{i\text{-}1})$$

Prediction for $\mathrm{x_i}$

$$K_i = p'_i\, h(h^2 p'_i + r)^{-1}$$

Gain

$$p'_i = f^2 p_{i-1} + q$$

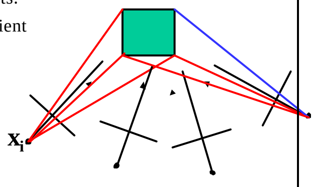
Standard deviation for prediction error

$$p_{i\text{-}1} = (1 - K_{i-1}\, h)\, p'_{i\text{-}1}$$

St.d. for estimation error

Applications: Structure from Motion

- Measurement vector components:
  - Coordinates of corners, "salient points"
- State vector components:
  - Camera motion parameters
  - Scene structure
- Is there enough equations?
  - N corners, 2N measurements
  - N unknown state components from structure (distances from first center of projection to 3D points)
  - 6 unknown state components from motion (translation and rotation)
  - More measurements than unknowns for every frame if N>6 (2N > N + 6)



- Batch methods
- Recursive methods (Kalman filter)

Problems with Tracking

- Initial detection
  - If it is too slow we will never catch up
  - If it is fast, why not do detection at every frame?
    - Even if raw detection can be done in real time, tracking saves processing cycles compared to raw detection. The CPU has other things to do.
- Detection is needed again if you lose tracking
- Most vision tracking prototypes use initial detection done by hand (see Forsyth and Ponce for discussion)

References

- Kalman, R.E., "A New Approach to Linear Prediction Problems", Transactions of the ASME--Journal of Basic Engineering, pp. 35-45, March 1960.
- Sorenson, H.W., "Least Squares Estimation: from Gauss to Kalman", IEEE Spectrum, vol. 7, pp. 63-68, July 1970.
- http://www.cs.unc.edu/~welch/kalmanLinks.html
- D. Forsyth and J. Ponce. Computer Vision: A Modern Approach, Chapter 19. http://www.cs.berkeley.edu/~daf/book3chaps.html
- O. Faugeras.. Three-Dimensional Computer Vision. MIT Press. Ch. 8, "Tracking Tokens over Time".