6.869
Computer Vision and Applications

Prof. Bill Freeman

Tracking
- Density propagation
- Linear Dynamic models / Kalman filter
- Data association
- Multiple models

Readings: F&P Ch 17

Schedule
- Thursday, April 28:
  - Kalman filter, PS4 due.
- Tuesday, May 3:
  - Tracking articulated objects, Exam 2 out
- Thursday, May 5:
  - How to write papers & give talks, Exam 2 due
- Tuesday, May 10:
  - Motion microscopy, separating shading and paint (“fun things my group is doing”)
- Thursday, May 12:
  - 5-10 min. student project presentations, projects due.

Tracking Applications
- Motion capture
- Recognition from motion
- Surveillance
- Targeting
Things to consider in tracking

What are the
• Real world dynamics
• Approximate / assumed model
• Observation / measurement process

Density propagation

• Tracking == Inference over time
• Much simplification is possible with linear dynamics and Gaussian probability models

Outline

• Recursive filters
• State abstraction
• Density propagation
• Linear Dynamic models / Kalman filter
• Data association
• Multiple models

Tracking and Recursive estimation

• Real-time / interactive imperative.
• Task: At each time point, re-compute estimate of position or pose.
  – At time n, fit model to data using time 0…n
  – At time n+1, fit model to data using time 0…n+1
• Repeat batch fit every time!

Recursive estimation

• Decompose estimation problem
  – part that depends on new observation
  – part that can be computed from previous history
• E.g., running average:
  \[ a_t = \alpha a_{t-1} + (1-\alpha) y_t \]
• Linear Gaussian models: Kalman Filter
• First, general framework…

Tracking

• Very general model:
  – We assume there are moving objects, which have an underlying state \( X \)
  – There are measurements \( Y \), some of which are functions of this state
  – There is a clock
    • at each tick, the state changes
    • at each tick, we get a new observation
• Examples
  – object is ball, state is 3D position+velocity, measurements are stereo pairs
  – object is person, state is body configuration, measurements are frames, clock is in camera (30 fps)
Three main issues in tracking

- **Prediction**: we have seen $y_1, \ldots, y_{i-1}$ — what state does this set of measurements predict for the $i$th frame? to solve this problem, we need to obtain a representation of $P(X_i|Y_1 = y_1, \ldots, Y_{i-1} = y_{i-1})$.
- **Data association**: Some of the measurements obtained from the $i$th frame may tell us about the object’s state. Typically, we use $P(X_i|Y_1 = y_1, \ldots, Y_{i-1} = y_{i-1})$ to identify these measurements.
- **Correction**: now that we have $y_i$ — the relevant measurements — we need to compute a representation of $P(X_i|Y_1 = y_1, \ldots, Y_i = y_i)$.

Simplifying Assumptions

- **Only the immediate past matters**: formally, we require
  \[ P(X_i|X_1, \ldots, X_{i-1}) = P(X_i|X_{i-1}) \]
  This assumption hugely simplifies the design of algorithms, as we shall see; furthermore, it isn’t terribly restrictive if we’re clever about interpreting $X_i$ as we shall show in the next section.
- **Measurements depend only on the current state**: we assume that $Y_i$ is conditionally independent of all other measurements given $X_i$. This means that
  \[ P(Y_1, Y_2, \ldots, Y_i|X_i) = P(Y_i|X_i)P(Y_1, \ldots, Y_{i-1}|X_i) \]
  Again, this isn’t a particularly restrictive or controversial assumption, but it yields important simplifications.

Kalman filter graphical model

- **Tracking as induction**
  - Assume data association is done
    - we’ll talk about this later, a dangerous assumption
  - Do correction for the 0’th frame
  - Assume we have corrected estimate for $i$’th frame
    - show we can do prediction for $i+1$, correction for $i+1$

Base case

Firstly, we assume that we have $P(X_0)$

\[
P(X_0|Y_0 = y_0) = \frac{P(y_0|X_0)P(X_0)}{P(y_0)} \propto P(y_0|X_0)P(X_0)
\]

Induction step

**Prediction**

Prediction involves representing

\[ P(X_i|Y_0, \ldots, Y_{i-1}) \]

given

\[ P(X_{i-1}|Y_0, \ldots, Y_{i-1}) \]

Our independence assumptions make it possible to write

\[
P(X_i|Y_0, \ldots, Y_{i-1}) = \int P(X_i, X_{i-1}|y_0, \ldots, y_{i-1})dX_{i-1}
\]

\[
= \int P(X_i|X_{i-1}, y_0, \ldots, y_{i-1})P(X_{i-1}|y_0, \ldots, y_{i-1})dX_{i-1}
\]

\[
= \int P(X_i|X_{i-1})P(X_{i-1}|y_0, \ldots, y_{i-1})dX_{i-1}
\]
### Linear dynamic models

- A linear dynamic model has the form
  \[
  x_i = N(\mathbf{D}_i X_i; \Sigma_x) \\
  y_i = N(\mathbf{M}_i X_i; \Sigma_y)
  \]

- This is much, much more general than it looks, and extremely powerful

### Constant velocity

- We have
  \[
  u_i = u_{i-1} + \Delta v_{i-1} + \epsilon_i \\
  v_i = v_{i-1} + \zeta_i
  \]

  - (the Greek letters denote noise terms)

- Stack \((u, v)\) into a single state vector
  \[
  \begin{bmatrix}
  u_i \\
  v_i
  \end{bmatrix} =
  \begin{bmatrix}
  1 & \Delta t \\
  0 & 1
  \end{bmatrix}
  \begin{bmatrix}
  u_{i-1} \\
  v_{i-1}
  \end{bmatrix} + \text{noise}
  \]

  - which is the form we had above

### Constant acceleration

- We have
  \[
  u_i = u_{i-1} + \Delta v_{i-1} + \epsilon_i \\
  v_i = v_{i-1} + \Delta a_{i-1} + \zeta_i \\
  a_i = a_{i-1} + \xi_i
  \]

  - (the Greek letters denote noise terms)

- Stack \((u, v, a)\) into a single state vector
  \[
  \begin{bmatrix}
  u_i \\
  v_i \\
  a_i
  \end{bmatrix} =
  \begin{bmatrix}
  1 & \Delta t & 0 \\
  0 & 1 & \Delta t \\
  0 & 0 & 1
  \end{bmatrix}
  \begin{bmatrix}
  u_{i-1} \\
  v_{i-1} \\
  a_{i-1}
  \end{bmatrix} + \text{noise}
  \]

  - which is the form we had above
Assume we have a point, moving on a line with a periodic movement defined with a differential eq:
\[ \frac{d^2 p}{dt^2} = -p \]
can be defined as
\[ \frac{du}{dt} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} u = \mathbf{S} u \]
with state defined as stacked position and velocity \( u = (p, v) \)

Periodic motion
\[ x_i = N(d_{i-1}, x_{i-1}; \Sigma_x) \]
\[ y_i = N(M x_i; \Sigma_y) \]

Take discrete approximation…(e.g., forward Euler integration with \( \Delta t \) stepsize.)
\[ u_i = u_{i-1} + \Delta t \frac{du}{dt} \]
\[ = u_{i-1} + \Delta t \mathbf{S} u_{i-1} \]
\[ = \begin{pmatrix} 1 & \Delta t \\ -\Delta t & 1 \end{pmatrix} u_{i-1} \]

Independence assumption
- Velocity and/or acceleration augmented position
- Constant velocity model equivalent to
\[ P(p_0|p_1, \ldots, p_{n-1}) = N(p_0, \ldots, p_{n-1}; \Sigma_p) \]
- \( \text{velocity} = p_{n-1} - p_{n-2} \)
- \( \text{acceleration} = (p_{n-1} - p_{n-2}) - (p_{n-2} - p_{n-3}) \)
- could also use \( p_{n-1} \) etc.

Higher order models

The Kalman Filter
- Key ideas:
  - Linear models interact uniquely well with Gaussian noise - make the prior Gaussian, everything else Gaussian and the calculations are easy
  - Gaussians are really easy to represent — once you know the mean and covariance, you’re done

Recall the three main issues in tracking
- Prediction: we have seen \( y_0, \ldots, y_{i-1} \) — what state does this set of measurements predict for the \( i \)th frame? to solve this problem, we need to obtain a representation of \( P(X_i|Y_0 = y_0, \ldots, Y_{i-1} = y_{i-1}) \).
- Data association: Some of the measurements obtained from the \( i \)th frame may tell us about the object’s state. Typically, we use \( P(X_i|Y_0 = y_0, \ldots, Y_{i-1} = y_{i-1}) \) to identify these measurements.
- Correction: now that we have \( y_i \) — the relevant measurements — we need to compute a representation of \( P(X_i|Y_0 = y_0, \ldots, Y_i = y_i) \).

(Ignore data association for now)
The Kalman Filter

The Kalman Filter in 1D

- Dynamic Model
  \[ x_t \sim N(d(x_{t-1}), \sigma^2_{x_t}) \]
  \[ y_t \sim N(m(x_t), \sigma^2_{y_t}) \]
  mean of \( P(X_{t+1} | y_0, \ldots, y_t) \) as \( \overline{X}_t \) ← Predicted mean
  mean of \( P(X_{t+1} | y_0, \ldots, y_t) \) as \( \overline{X}_t \) ← Corrected mean
  the standard deviation of \( P(X_{t+1} | y_0, \ldots, y_t) \) as \( \sigma_t \)
  of \( P(X_{t+1} | y_0, \ldots, y_t) \) as \( \sigma_t \)

Prediction for 1D Kalman filter

- The new state is obtained by
  \[ x_t \sim N(d(x_{t-1}), \sigma^2_{x_t}) \]
  \[ y_t \sim N(m(x_t), \sigma^2_{y_t}) \]
  - multiplying old state by known constant
  - adding zero-mean noise

- Therefore, predicted mean for new state is
  \[ \overline{X}_t = d_t \overline{X}_{t-1} \]
  \[ (\sigma_t)^2 = \sigma^2_{x_t} + (d_t \sigma^2_{y_t})^2 \]

- Old variance is normal random variable
  - variance is multiplied by square of constant
  - and variance of noise is added.
Correction for 1D Kalman filter

\[ x^*_t = \left( \frac{x_t \sigma_{e_t}^2 + m_t \sigma_{e_t}^2}{\sigma_{e_t}^2 + m_t \sigma_{e_t}^2} \right) \]

\[ \sigma^*_t = \sqrt{\frac{\sigma_{e_t}^2 + m_t \sigma_{e_t}^2}{\sigma_{e_t}^2 + m_t \sigma_{e_t}^2}} \]

Notice:
- if measurement noise is small, we rely mainly on the measurement,
- if it’s large, mainly on the prediction
- \( \sigma \) does not depend on \( y \)

---

Constant Velocity Model

Dynamical Model

\[ z_t \sim N(x_{t-1}, \sigma_z) \]
\[ \eta_t \sim N(0, \sigma_\eta) \]

Start Assumptions: \( \sigma_z \) and \( \sigma_\eta \) are known.

Update Equations: Prediction

\[ \hat{x}_t^* = \frac{x_{t-1} \sigma_{e_t}^2 + m_t \sigma_{e_t}^2}{\sigma_{e_t}^2 + m_t \sigma_{e_t}^2} \]

\[ \sigma_t^* = \sqrt{\frac{\sigma_{e_t}^2 + m_t \sigma_{e_t}^2}{\sigma_{e_t}^2 + m_t \sigma_{e_t}^2}} \]

Update Equations: Correction

The *s give \( \hat{x}_t^* \), +s give \( \hat{x}_t \). Vertical bars are 3 standard deviation bars.
Smoothing

- Idea
  - We don’t have the best estimate of state - what about the future?
  - Run two filters, one moving forward, the other backward in time.
  - Now combine state estimates
    - The crucial point here is that we can obtain a smoothed estimate by viewing the backward filter’s prediction as yet another measurement for the forward filter.

n-D

Generalization to n-D is straightforward but more complex.
n-D

Generalization to n-D is straightforward but more complex.

Prediction:

- Multiply estimate at prior time with forward model:
  \[ \bar{x}_t = D_t \bar{x}_{t-1} \]
- Propagate covariance through model and add new noise:
  \[ \Sigma_t = \Sigma_{a_t} + D_t \sigma_{n-t} D_t^T \]

n-D Correction

Generalization to n-D is straightforward but more complex.

Correction:

- Update a priori estimate with measurement to form a posteriori

\[ \bar{x}_t^+ = \bar{x}_t + K_t \left[ y_t - M_t \bar{x}_t \right] \]

\[ K_t = \Sigma_t^+ M_t^T \left[ M_t \Sigma_t^+ M_t^T + \Sigma_{m_t} \right]^{-1} \]

As measurement becomes more reliable, K weights residual more heavily,
\[ \lim_{\Sigma_m \to 0} K_t = M^{-1} \]

As prior covariance approaches 0, measurements are ignored:
\[ \lim_{\Sigma_a \to 0} K_t = 0 \]
Data Association

In real world, we have clutter as well as data…

E.g., match radar returns to set of aircraft trajectories.

Data Association

Approaches:

• Nearest neighbours
  – choose the measurement with highest probability given predicted state
  – popular, but can lead to catastrophe
• Probabilistic Data Association
  – combine measurements, weighting by probability given predicted state
  – gate using predicted state
Red: tracks of 10 drifting points. Blue, black: point being tracked

What if environment is sometimes unpredictable?
Do people move with constant velocity?
Test several models of assumed dynamics, use the best.

Multiple model filters
Test several models of assumed dynamics

MM estimate
Two models: Position (P), Position+Velocity (PV)

[figure from Welsh and Bishop 2001]
**P likelihood**

[figure from Welsh and Bishop 2001]

**No lag**

[figure from Welsh and Bishop 2001]

**Smooth when still**

[figure from Welsh and Bishop 2001]

**Resources**

- Kalman filter homepage
  http://www.cs.unc.edu/~welch/kalman/

- Kevin Murphy’s Matlab toolbox:
  http://www.ai.mit.edu/~murphyk/Software/Kalman/kalman.html

**Jepson, Fleet, and El-Maraghi tracker**


**Robust Online Appearance Models for Visual Tracking**

Allan D. Jepson*, David J. Fleet†, Thomas F. El-Maraghi‡

*Department of Computer Science, University of Toronto, Toronto, M5S 1A4
†Xerox Palo Alto Research Center, 3333 Coyote Hill Rd, Palo Alto, CA 94304
Jepson, Fleet, and El-Maraghi tracker

Figure 1: Each row shows, from left to right, the tracking region, the saliency component mixing probability on $p_h$ (orange, $2^1$), and the probability mixing on $p_h$ (red). The outer circle is drawn in blue, red, purple, and blue (left to right). See the model parameters.

Jepson, Fleet, and El-Maraghi tracker

Figure 2: Estimation using online EM: (top) The original data and the true-state (dashed blue) and the estimated mean of the visible process (thick black). The noise is a mixture of Gaussian and uniform densities, with mixing probabilities (0.9, 0.1), except for 15 frames at 300 which are pure mixtures. (bottom) Mixing probabilities for $\delta$ (black), $\nu$ (dashed blue), and the $\xi$ (light green).

Show videos