CS 1699: Intro to Computer Vision

Tracking

Prof. Adriana Kovashka
University of Pittsburgh
December 3, 2015
Today: Tracking

• Tracking how an object moves
• Examples of tracking applications
• Probabilistic approach to tracking
  – Kalman filters
• Tracking challenges

• TA will have HW4 graded on Tuesday
• HW5 due next Thursday
  – Use a random sample of images to train the attribute classifiers
Motion: Why is it useful?
Motion: Why is it useful?

- Even “impoverished” motion data can evoke a strong percept

Tracking Examples

Traffic: https://www.youtube.com/watch?v=DiZHQ4peqjg

Soccer: http://www.youtube.com/watch?v=ZqQIlItFAnxg

Face: http://www.youtube.com/watch?v=i_bZNVmhJ2o

Body: https://www.youtube.com/watch?v=_Ahy0Gh69-M

Eye: http://www.youtube.com/watch?v=NCTydmUEMtg

Gaze: http://www.youtube.com/watch?v=-G6Rw5cU-1c
How would you do it?
Tracking: some applications

Body pose tracking, activity recognition

Censusing a bat population

Video-based interfaces

Medical apps

Surveillance
Feature tracking

• Extract visual features (corners, textured areas) and “track” them over multiple frames
• Many problems (e.g. structure from motion) require matching points
• Many challenges, e.g. points may change appearance over time, points may appear or disappear
Feature tracking

- Given two subsequent frames, estimate the point translation
- The *Lucas-Kanade optical flow* method assumes:
  - **Brightness constancy**: projection of the same point looks the same in every frame
  - **Small motion**: points do not move very far
  - **Spatial coherence**: points move like their neighbors

Adapted from Derek Hoiem
Using optical flow: recognizing facial expressions

Recognizing Human Facial Expression (1994)
by Yaser Yacoob, Larry S. Davis

Kristen Grauman
Using optical flow: action recognition at a distance

- Features = optical flow within a region of interest
- Classifier = nearest neighbors

[Efros, Berg, Mori, & Malik 2003]
http://graphics.cs.cmu.edu/people/efros/research/action/
Search window is centered at the point where we last saw the feature, in image I1.

Best match = position where we have the highest normalized cross-correlation value.
Example: A Camera Mouse

• Video interface: use feature tracking as mouse replacement
  
  • User clicks on the feature to be tracked
  • Take the 15x15 pixel square of the feature
  • In the next image do a search to find the 15x15 region with the highest correlation
  • Move the mouse pointer accordingly
  • Repeat in the background every 1/30th of a second

James Gips and Margrit Betke
http://www.bc.edu/schools/csom/eagleeyes/
Example: A Camera Mouse

• Specialized software for communication, games

James Gips and Margrit Betke
http://www.bc.edu/schools/csom/eagleeyes/
Feature-based matching for motion

• For a discrete matching search, what are the tradeoffs of the chosen search window size?

• Which patches to track?
  • Select interest points – e.g. corners

• Where should the search window be placed?
  • Near match at previous frame
  • More generally, taking into account the expected dynamics of the object
Things that make visual tracking difficult

• Small, few visual features
• Erratic movements, moving very quickly
• Occlusions, leaving and coming back
• Surrounding similar-looking objects
Strategies for tracking

• Tracking by repeated detection
  – Works well if object is easily detectable (e.g., face or colored glove) and there is only one
  – Need some way to link up detections
  – Best you can do, if you can’t predict motion
Tracking with dynamics

• **Key idea:** Based on a model of expected motion, predict where objects will occur in next frame, before even seeing the image
  – Restrict search for the object
  – Measurement noise is reduced by trajectory smoothness
  – Robustness to missing or weak observations
  – **Assumptions:** camera is not moving instantly to new viewpoint, objects do not disappear and reappear in different places in the scene
Detection vs. tracking

t=1  t=2  

...  t=20  t=21
Detection vs. tracking

Detection: We detect the object independently in each frame and can record its position over time, e.g., based on blob’s centroid or detection window coordinates.
Detection vs. tracking

Tracking with *dynamics*: We use image measurements to estimate position of object, but also incorporate position predicted by dynamics, i.e., our expectation of object’s motion pattern.
Tracking: prediction + correction

Belief

Measurement

Corrected prediction
Tracking: prediction + correction

Belief: prediction

Corrected prediction

old belief

measurement

Time t

Time t+1
General model for tracking

• **state** $X$: The actual state of the moving object that we want to estimate but cannot observe
  – State could be any combination of position, pose, viewpoint, velocity, acceleration, etc.

• **observations** $Y$: Our actual measurement or observation of state $X$, which can be very noisy

• At each time $t$, the state changes to $X_t$ and we get a new observation $Y_t$
General model for tracking

• **Our goal:** recover most likely state $X_t$ given
  - All observations seen so far, i.e. $y_1, y_2, ..., y_t$
  - Knowledge about dynamics of state transitions
Steps of tracking

- **Prediction:** What is the next state of the object given past measurements?

\[
P(X_t | Y_0 = y_0, \ldots, Y_{t-1} = y_{t-1})
\]
Steps of tracking

• **Prediction:** What is the next state of the object given past measurements?

\[
P(X_t | Y_0 = y_0, \ldots, Y_{t-1} = y_{t-1})
\]

• **Correction:** Compute an updated estimate of the state from prediction and measurements

\[
P(X_t | Y_0 = y_0, \ldots, Y_{t-1} = y_{t-1}, Y_t = y_t)
\]
Simplifying assumptions

• Only the immediate past matters

\[ P(X_t | X_0, \ldots, X_{t-1}) \]

Dynamics model
Simplifying assumptions

• Only the immediate past matters

\[ P(X_t | X_0, \ldots, X_{t-1}) = P(X_t | X_{t-1}) \]

dynamics model

• Measurements depend only on the current state

\[ P(Y_t | X_0, Y_0 \ldots, X_{t-1}, Y_{t-1}, X_t) \]
Simplifying assumptions

• Only the immediate past matters

\[ P(X_t | X_0, \ldots, X_{t-1}) = P(X_t | X_{t-1}) \]

dynamics model

• Measurements depend only on the current state

\[ P(Y_t | X_0, Y_0 \ldots, X_{t-1}, Y_{t-1}, X_t) = P(Y_t | X_t) \]

observation model
Problem statement

• We have models for

  Likelihood of next state given current state (dynamics model):  
  \[ P(X_t | X_{t-1}) \]

  Likelihood of observation given the state (observation or measurement model):  
  \[ P(Y_t | X_t) \]

• We want to recover, for each t:  
  \[ P(X_t | y_0, \ldots, y_t) \]
Notation reminder

\( \mathbf{x} \sim N(\mathbf{\mu}, \Sigma) \)

- Random variable with Gaussian probability distribution that has the mean vector \( \mathbf{\mu} \) and covariance matrix \( \Sigma \).
- \( \mathbf{x} \) and \( \mathbf{\mu} \) are \( d \)-dimensional, \( \Sigma \) is \( d \times d \).

If \( \mathbf{x} \) is 1-d, we just have one \( \Sigma \) parameter \( \rightarrow \) the variance: \( \sigma^2 \).
Dynamics and observation models

- **Dynamics** model (represents evolution of state over time)

\[ P(X_t | X_{t-1}) \]

- **Observation or measurement** model (at every time step we get a noisy measurement of the state)

\[ P(y_t | X_t) \]

This is how these models are defined in the *Kalman filter.*

Adapted from Kristen Grauman, Simon Prince
The Kalman filter

• Linear dynamics model: state undergoes linear transformation plus Gaussian noise

• Observation model: measurement is linearly transformed state plus Gaussian noise

• The predicted/corrected state distributions are Gaussian
  – You only need to maintain the mean and covariance
  – The calculations are easy
Example: Constant velocity (1D points)

1 d position

measurements

states

time

Kristen Grauman
Example: Constant velocity (1D points)

- State vector: position $p$ and velocity $v$

$$
\begin{align*}
  x_t &= \begin{bmatrix} p_t \\ v_t \end{bmatrix} \\
  p_t &= \\

  x_t &= D_t x_{t-1} + \text{noise} = \begin{bmatrix} 1 & \Delta t \\
  0 & 1 \end{bmatrix} \begin{bmatrix} p_{t-1} \\ v_{t-1} \end{bmatrix} + \text{noise} \\

- Measurement is position only

$$
\begin{align*}
  y_t &= M x_t + \text{noise} = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} p_t \\ v_t \end{bmatrix} + \text{noise}
\end{align*}$$
Probabilistic tracking

• Base case:
  – Start with initial prior that predicts state in absence of any evidence: $P(X_0)$
  – For the first frame, correct this given the first measurement: $Y_0 = y_0$
Probabilistic tracking

• Base case:
  – Start with initial prior that predicts state in absence of any evidence: \( P(X_0) \)
  – For the first frame, correct this given the first measurement: \( Y_0 = y_0 \)

\[
P(X_0 \mid Y_0 = y_0) = \frac{P(y_0 \mid X_0)P(X_0)}{P(y_0)}
\]
Probabilistic tracking

• Base case:
  – Start with initial prior that predicts state in absence of any evidence: \( P(X_0) \)
  – For the first frame, correct this given the first measurement: \( Y_0=y_0 \)

• Given corrected estimate for frame \( t-1 \):
  – Predict for frame \( t \) \( \Rightarrow \) \( P(X_t|y_0,\ldots,y_{t-1}) \)
  – Observe \( y_t \); Correct for frame \( t \) \( \Rightarrow \) \( P(X_t|y_0,\ldots,y_{t-1},y_t) \)
Prediction

- Prediction involves representing \( P(X_t | y_0, \ldots, y_{t-1}) \) given \( P(X_{t-1} | y_0, \ldots, y_{t-1}) \)

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

Law of total probability

\[
\Pr(A) = \sum_n \Pr(A \cap B_n)
\]
Prediction

• Prediction involves representing $P(X_t | y_0, \ldots, y_{t-1})$
given $P(X_{t-1} | y_0, \ldots, y_{t-1})$

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

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

Conditioning on $X_{t-1}$

\[ P(A, B) = P(A | B)P(B) \]
Prediction

- Prediction involves representing $P(X_t | y_0, \ldots, y_{t-1})$ given $P(X_{t-1} | y_0, \ldots, y_{t-1})$

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

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

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

Independence assumption (only immediate past state matters)

Adapted from Amin Sadeghi
Prediction

- Prediction involves representing $P(X_t | y_0, \ldots, y_{t-1})$
given $P(X_{t-1} | y_0, \ldots, y_{t-1})$

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

- Dynamics model
- Corrected estimate from previous step
Correction

• Correction involves computing \( P(X_t | y_0, \ldots, y_t) \) given predicted value \( P(X_t | y_0, \ldots, y_{t-1}) \)
Correction

- Correction involves computing \( P(X_t | y_0, \ldots, y_t) \)
given predicted value \( P(X_t | y_0, \ldots, y_{t-1}) \)
Correction

- Correction involves computing \( P(X_t | y_0, \ldots, y_t) \) given predicted value \( P(X_t | y_0, \ldots, y_{t-1}) \)

\[
P(X_t | y_0, \ldots, y_t) = \frac{P(y_t | X_t, y_0, \ldots, y_{t-1})}{P(y_t | y_0, \ldots, y_{t-1})} P(X_t | y_0, \ldots, y_{t-1})
\]

Bayes’ Rule

\[
P(A|B) = \frac{P(B|A)P(A)}{P(B)}.
\]
Correction

- Correction involves computing $P(X_t | y_0, \ldots, y_t)$ given predicted value $P(X_t | y_0, \ldots, y_{t-1})$

\[
P(X_t | y_0, \ldots, y_t) = \frac{P(y_t | X_t, y_0, \ldots, y_{t-1})}{P(y_t | y_0, \ldots, y_{t-1})} \frac{P(X_t | y_0, \ldots, y_{t-1})}{P(y_t | y_0, \ldots, y_{t-1})}
\]

Independence assumption (observation $y_t$ directly depends only on state $X_t$)
Correction

• Correction involves computing $P(X_t | y_0, \ldots, y_t)$ given predicted value $P(X_t | y_0, \ldots, y_{t-1})$

\[
P(X_t | y_0, \ldots, y_t) \]
\[
= \frac{P(y_t | X_t, y_0, \ldots, y_{t-1})}{P(y_t | y_0, \ldots, y_{t-1})} P(X_t | y_0, \ldots, y_{t-1})
\]
\[
= \frac{P(y_t | X_t) P(X_t | y_0, \ldots, y_{t-1})}{P(y_t | y_0, \ldots, y_{t-1})}
\]
\[
= \frac{P(y_t | X_t) P(X_t | y_0, \ldots, y_{t-1})}{\int P(y_t | X_t) P(X_t | y_0, \ldots, y_{t-1}) dX_t}
\]

Conditioning on $X_t$
Correction

- Correction involves computing $P(X_t | y_0, \ldots, y_t)$ given predicted value $P(X_t | y_0, \ldots, y_{t-1})$

\[
P(X_t | y_0, \ldots, y_t) = \frac{P(y_t | X_t, y_0, \ldots, y_{t-1})}{P(y_t | y_0, \ldots, y_{t-1})} P(X_t | y_0, \ldots, y_{t-1})
\]

\[
= P(y_t | X_t) P(X_t | y_0, \ldots, y_{t-1}) \frac{P(y_t | y_0, \ldots, y_{t-1})}{P(y_t | y_0, \ldots, y_{t-1})}
\]

\[
= \int P(y_t | X_t) P(X_t | y_0, \ldots, y_{t-1}) dX_t
\]
Summary: Prediction and correction

• Prediction:
  Know corrected state from previous time step, and all measurements up to (excluding) the current one →
  Predict distribution over next state

\[
P(X_t \mid y_0, \ldots, y_{t-1})
\]

Time advances: \( t++ \)

• Correction:
  Know prediction of state, and next measurement →
  Update distribution over current state

\[
P(X_t \mid y_0, \ldots, y_t)
\]
Summary: Prediction and correction

Prediction:

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

dynamics model

corrected estimate from previous step

Correction:

\[
P(X_t | y_0, \ldots, y_t) = \frac{P(y_t | X_t)P(X_t | y_0, \ldots, y_{t-1})}{\int P(y_t | X_t)P(X_t | y_0, \ldots, y_{t-1})dX_t}
\]

observation model

predicted estimate
Kalman filter processing

- Time $t$
- Time $t+1$

**Constant velocity model**

- $\circ$ state
- $\times$ measurement
- $*$ predicted mean estimate
- $+$ corrected mean estimate
- Bars: variance estimates before and after measurements

Kristen Grauman
Constant velocity model

Kalman filter processing

- \(o\) state
- \(x\) measurement
- * predicted mean estimate
- + corrected mean estimate

bars: variance estimates before and after measurements

Kristen Grauman

Time t

Time t+1
Constant velocity model

Kalman filter processing

- Predicted mean estimate
- Corrected mean estimate

Bars: variance estimates before and after measurements

Kristen Grauman
Constant velocity model

Kalman filter processing

- o state
- x measurement
- * predicted mean estimate
- + corrected mean estimate
- bars: variance estimates before and after measurements
Questions

• How to *predict* if missing a detection in the current frame?
  – Prediction only depends on previous frames

• How to *correct* if missing detection in the current frame?
  – Don’t correct, just use prediction

• What if we have two detections in a frame?
  – Determine which of the two is more likely correct
Example: Kalman Filter

Ground Truth  Observation  Correction
A bat census

http://www.cs.bu.edu/~betke/research/bats/
Tracking issues

• Initialization
  – Manual
  – Background subtraction
  – Detection
Background subtraction: average/median image
Background subtraction
Tracking issues

• Initialization

• Obtaining observation and dynamics models
  – Observation model: match a template or use a trained detector
  – Dynamics model: usually specify using domain knowledge
Tracking issues

• Initialization
• Obtaining observation and dynamics models
• Uncertainty of prediction vs. correction
  – If the dynamics model is too strong, will end up ignoring the data
  – If the observation model is too strong, tracking is reduced to repeated detection

Too strong dynamics model

Too strong observation model
Tracking issues

• Initialization
• Getting observation and dynamics models
• Prediction vs. correction
• Data association
  – When tracking multiple objects, need to assign right objects to right tracks (particle filters good for this)
Tracking issues

• Initialization
• Getting observation and dynamics models
• Prediction vs. correction
• Data association
• Drift
  – Errors can accumulate over time

Things to remember

• Tracking objects = detection + prediction

• Probabilistic framework
  – Predict next state
  – Update current state based on observation

• Two simple but effective methods
  – Kalman filters: Gaussian distribution
  – Particle filters: Multimodal distribution (for multiple objects)

• Challenges