Video: Tracking and Action Recognition

EECS 442 – David Fouhey
Fall 2019, University of Michigan

http://web.eecs.umich.edu/~fouhey/teaching/EECS442_F19/
Today: Tracking Objects

• Goal: Locating a moving object/part across video frames

• This class:
  • Examples
  • Probabilistic Tracking
  • Kalman filter
  • Particle filter

Slide credit: D. Hoiem
Tracking Examples
Best Tracking

Slide credit: B. Babenko
Difficulties

- Erratic movements, rapid motion
- Occlusion
- Surrounding similar objects
Tracking by Detection

Tracking by detection:
• Works if object is detectable
• Need some way to link up detections
Tracking With Dynamics

Based on motion, predict object location
• Restrict search for object
• Measurement noise is reduced by smoothness
• Robustness to missing or weak observations
Strategies For Tracking

- Tracking with motion prediction:
  - Predict object’s state in next frame.
  - Fuse with observation.
General Tracking Model

**State X**: actual state of object that we want to estimate. Could be: Pose, viewpoint, velocity, acceleration.

**Observation Y**: our “measurement” of state X. Can be noisy. At each time step \( t \), state changes to \( X_t \), get \( Y_t \).
Steps of Tracking

**Prediction**: What’s the next state of the object given past measurements

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

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

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

Only immediate past matters (Markovian)

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

Measurement depends only on current state (Independence)

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

Have models for:

(1) P(next state) given current state / Transition

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

(2) P(observation) given state / Observation

\[ P(Y_t|X_t) \]

Want to recover, for each timestep t

\[ P(X_t|y_0, \ldots, y_t) \]
Probabilistic tracking

- Base case:
  - Start with initial *prediction/prior*: $P(X_0)$
  - For the first frame, *correct* this given the first measurement: $Y_0 = y_0$
Probabilistic tracking

• Base case:
  • Start with initial \textit{prediction}/prior: \( P(X_0) \)
  • For the first frame, \textit{correct} this given the first measurement: \( Y_0 = y_0 \)

• Each subsequent step:
  • \textit{Predict} \( X_t \) given past evidence
  • Observe \( y_t \): \textit{correct} \( X_t \) given current evidence
Prediction

Given \( P(X_{t-1}|y_0,\ldots,y_{t-1}) \) want \( P(X_t|y_0,\ldots,y_{t-1}) \)

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

**Total probability**

**Condition on \( X_{t-1} \)**

**Markovian**

**dynamics model**

**corrected estimate from previous step**

Slide credit: D. Hoiem
Correction

Given \( P(X_t|y_0,\ldots,y_{t-1}) \) want \( P(X_t|y_0,\ldots,y_{t-1},y_t) \)

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

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

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

Bayes Rule

Independence Assumption

Condition on \( X_t \)

Slide credit: D. Hoiem
Correction

Given $P(X_t|y_0,\ldots,y_{t-1})$ want $P(X_t|y_0,\ldots,y_{t-1},y_t)$

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

Bayes Rule

Independence Assumption

Condition on $X_t$

Normalization Factor

Slide credit: D. Hoiem
Summarize

Transition

Observation

Prediction:

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

Correction:

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

Nasty integrals! Also these are probability distributions
Solution 1 – Kalman Filter

- What’s the product of two Gaussians?
  - Gaussian
- What do you need to keep track of for a multivariate Gaussian?
  - Mean, Covariance

Kalman filter: assume everything’s Gaussian
“The Apollo computer used 2k of magnetic core RAM and 36k wire rope [...]. The CPU was built from ICs [...]. Clock speed was under 100 kHz”

Rudolf Kalman

Photo, Quote credit: Wikipedia
Comparison

Ground Truth

Observation

Correction

Slide credit: D. Hoiem
Example: Kalman Filter

Ground Truth

Observation

Prediction

Correction

Slide credit: D. Hoiem
Propagation of Gaussian densities

Current state

(a) Decent model if there is just one object, but localization is imprecise

(b) Expected change

(c) Stochastic diffusion

(d) Reactive effect of measurement

Uncertainty

Observation and Correction

Slide credit: D. Hoiem
Particle filtering

Represent the state distribution non-parametrically

- Prediction: Sample possible values $X_{t-1}$ for the previous state

- Correction: Compute likelihood of $X_t$ based on weighted samples and $P(y_t|X_t)$

M. Isard and A. Blake, CONDENSATION -- conditional density propagation for visual tracking, IJCV 29(1):5-28, 1998
Non-parametric densities

Good if there are multiple, confusible objects (or clutter) in the scene
Particle Filtering
Particle Filtering More Generally

• **Object tracking:**
  • State: object location
  • Observation: detect bounding box
  • Transition: assume constant velocity, etc.

• **Vehicle tracking:**
  • State: car location [x,y,theta] + velocity
  • Observation: register location in map
  • Transition: assume constant velocity, etc.
Particle Filtering More Generally

Lost! Leveraging the Crowd for Probabilistic Visual Self-Localization

Marcus A Brubaker, Andreas Geiger and Raquel Urtasun

Code and other videos at:
http://www.cs.toronto.edu/~mbrubake
In General

• If you have something intractable:
• Option 1: Pretend you’re dealing with Gaussians, everything is nice
• Option 2: Monte-carlo method, don’t have to do intractable math
MD-Net

- Offline: train to differentiate between target and bg for K different targets
- Online: fine-tune network in new sequence

Nam and Han, CVPR 2016, Learning Multi-Domain Convolutional Neural Networks For Visual Tracking
Learning Multi-Domain Convolutional Neural Networks for Visual Tracking

Hyeonseob Nam and Bohyung Han
Tracking Issues

- Initialization
  - Manual (click on stuff)
  - Detection
  - Background subtraction
Detour: Background Subtraction
Moving in Time

• Moving only in time, while not moving in space, has many advantages
  • No need to find correspondences
  • Can look at how each ray changes over time
  • In science, always good to change just one variable at a time

• This approach has always interested artists (e.g. Monet)
As can look at video data as a spatio-temporal volume
  - If camera is stationary, each line through time corresponds to a single ray in space
  - We can look at how each ray behaves
  - What are interesting things to ask?
Example

Slide credit: A. Efros
Examples

Average image

Median Image

Slide credit: A. Efros
Average/Median Image

Slide credit: A. Efros
Background Subtraction

Slide credit: A. Efros
Tracking Issues

• Initialization
• Getting observation and dynamics models
  • Observation model: match template or use trained detector
  • Dynamics Model: specify with domain knowledge
Tracking Issues

- Initialization
- Getting observation and dynamics models
- Combining prediction vs correction:
  - Dynamics too strong: ignores data
  - Observation too strong: tracking = detection

Too strong dynamics model

Too strong observation model

Slide credit: D. Hoiem
Tracking Issues

• Initialization
• Getting observation and dynamics models
• Combining prediction vs correction
• Data association:
  • Need to keep track of which object is which. Particle filters good for this
Tracking Issues – Data Association
Tracking Issues

- Initialization
- Getting observation and dynamics models
- Combining prediction vs correction
- Data association
- Drift
  - Errors can accumulate over time
Drift

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

Slide credit: D. Hoiem
Action Recognition

• Image recognition:
  • Input: HxWx3 image
  • Output: F-dimensional output

• Action recognition
  • Input: ?x?x? video
  • Output: F-dimensional output
Datasets – KTH

#Classes: 6, Videos: 2391, Source: Lab, Year: 2004
Recognizing Human Actions: A Local SVM Approach
C. Schuldt, I. Laptev, B. Caputo
Datasets – UCF 101

#Classes: 101, #Videos: 9,511, Source: YouTube, Year: 2012
K. Soomro, A. Zamir, M. Shah
Datasets – Kinetics

#Classes: 400, #Videos: 240K, Source: YouTube, Year: 2017
The Kinetics Human Action Video Dataset
Models for Action Recognition

• Take learned sequence modeler (also used in language tasks, e.g., sentence -> sentiment)
• Feed in convnet activations as opposed to words
Models for Action Recognition

- One network (Image) takes HxWx3 image
- Other network (Flow) takes HxWx2*N image
- Add them together

Diagram credit: J. Carreira, A. Zisserman
Models for Action Recognition

Spatial stream ConvNet
- conv1: 7x7x96, stride 2, norm, pool 2x2
- conv2: 5x5x256, stride 2, norm, pool 2x2
- conv3: 3x3x512, stride 1
- conv4: 3x3x512, stride 1
- conv5: 3x3x512, stride 1, pool 2x2
- full6: 4096, dropout
- full7: 2048, dropout
- softmax

Temporal stream ConvNet
- conv1: 7x7x96, stride 2, norm, pool 2x2
- conv2: 5x5x256, stride 2, pool 2x2
- conv3: 3x3x512, stride 1
- conv4: 3x3x512, stride 1
- conv5: 3x3x512, stride 1, pool 2x2
- full6: 4096, dropout
- full7: 2048, dropout
- softmax

Diagram credit: J. Carreira, A. Zisserman
Models for Action Recognition

- Dump all the frames in as a HxWx3xN tensor
- Convolutions are 3D

Diagram credit: J. Carreira, A. Zisserman
Models for Action Recognition

- Filters pick up on spatial patterns and motion patterns
Models for Action Recognition

- RGB frames go in as $H \times W \times 3 \times N$ tensor
- Flow frames in as $H \times W \times 2 \times N$ tensor
- Convolutions are 3D

Diagram credit: J. Carreira, A. Zisserman
Comparisons

Take-homes:
- Flow + RGB does best
- 3D Convolutions does best

<table>
<thead>
<tr>
<th>Architecture</th>
<th>UCF-101</th>
<th>miniKinetics</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RGB</td>
<td>Flow</td>
</tr>
<tr>
<td>(a) LSTM</td>
<td>81.0</td>
<td>–</td>
</tr>
<tr>
<td>(b) 3D-ConvNet</td>
<td>51.6</td>
<td>–</td>
</tr>
<tr>
<td>(c) Two-Stream</td>
<td>83.6</td>
<td>85.6</td>
</tr>
<tr>
<td>(e) Two-Stream I3D</td>
<td>84.5</td>
<td>90.6</td>
</tr>
</tbody>
</table>
Hmm… #1

Just looking at independent frames does shockingly well.

<table>
<thead>
<tr>
<th>Architecture</th>
<th>UCF-101</th>
<th>miniKinetics</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RGB</td>
<td>Flow</td>
</tr>
<tr>
<td>(a) LSTM</td>
<td>81.0</td>
<td>–</td>
</tr>
<tr>
<td>(b) 3D-ConvNet</td>
<td>51.6</td>
<td>–</td>
</tr>
<tr>
<td>(c) Two-Stream</td>
<td>83.6</td>
<td>85.6</td>
</tr>
<tr>
<td>(e) Two-Stream I3D</td>
<td>84.5</td>
<td>90.6</td>
</tr>
</tbody>
</table>
Hmm… #2

Using optical flow as input improves things. If flow is so important, can’t it just learn this on its own?

<table>
<thead>
<tr>
<th>Architecture</th>
<th>UCF-101</th>
<th>miniKinetics</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RGB</td>
<td>Flow</td>
</tr>
<tr>
<td>(a) LSTM</td>
<td>81.0</td>
<td>–</td>
</tr>
<tr>
<td>(b) 3D-ConvNet</td>
<td>51.6</td>
<td>–</td>
</tr>
<tr>
<td>(c) Two-Stream</td>
<td>83.6</td>
<td>85.6</td>
</tr>
<tr>
<td>(e) Two-Stream I3D</td>
<td><strong>84.5</strong></td>
<td><strong>90.6</strong></td>
</tr>
</tbody>
</table>