Computer Vision 2 – Lecture 5

Tracking with Linear Dynamic Models(02.05.2016)

Prof. Dr. Bastian Leibe, Dr. Jörg Stückler
leibe@vision.rwth-aachen.de, stueckler@vision.rwth-aachen.de

RWTH Aachen University, Computer Vision Group
http://www.vision.rwth-aachen.de
Content of the Lecture

- Single-Object Tracking
- Bayesian Filtering
  - Kalman Filters, EKF
  - Particle Filters
- Multi-Object Tracking
- Visual Odometry
- Visual SLAM & 3D Reconstruction
Recap: Tracking-by-Detection

Main ideas

- Apply a generic object detector to find objects of a certain class
- Based on the detections, extract object appearance models
- Link detections into trajectories
Recap: Elements of Tracking

- **Detection**
  - *Where are candidate objects?*

- **Data association**
  - *Which detection corresponds to which object?*

- **Prediction**
  - *Where will the tracked object be in the next time step?*
Recap: Sliding-Window Object Detection

- Fleshing out this pipeline a bit more, we need to:
  1. Obtain training data
  2. Define features
  3. Define classifier

Slide credit: Kristen Grauman
Recap: Object Detector Design

• In practice, the classifier often determines the design.
  – Types of features
  – Speedup strategies

• We looked at 3 state-of-the-art detector designs
  – Based on SVMs
  – Based on Boosting
  – Based on CNNs
Recap: Histograms of Oriented Gradients (HOG)

- Holistic object representation
  - Localized gradient orientations

Object/Non-object

Linear SVM

Collect HOGs over detection window

Contrast normalize over overlapping spatial cells

Weighted vote in spatial & orientation cells

Compute gradients

Gamma compression

Image Window
Recap: Deformable Part-based Model (DPM)

• Multiscale model captures features at two resolutions

Score of object hypothesis is sum of filter scores minus deformation costs

Score of filter: dot product of filter with HOG features underneath it

[Felzenszwalb, McAllister, Ramanan, CVPR'08]
Recap: DPM Hypothesis Score

\[
score(p_0, \ldots, p_n) = \sum_{i=0}^{n} F_i \cdot \phi(H, p_i) - \sum_{i=1}^{n} d_i \cdot (dx_i^2, dy_i^2)\]

“data term”

“spatial prior”

filters

deformation parameters

displacements

score(\(z\)) = \beta \cdot \Psi(H, z)

concatenation filters and deformation parameters

concatenation of HOG features and part displacement features

[Felzenszwalb, McAllister, Ramanan, CVPR’08]
Recap: Integral Channel Features

- Generalization of Haar Wavelet idea from Viola-Jones
  - Instead of only considering intensities, also take into account other feature channels (gradient orientations, color, texture).
  - Still efficiently represented as integral images.

Recap: Integral Channel Features

- Generalize also block computation
  - 1\textsuperscript{st} order features:
    - Sum of pixels in rectangular region.
  - 2\textsuperscript{nd}-order features:
    - Haar-like difference of sum-over-blocks
  - Generalized Haar:
    - More complex combinations of weighted rectangles
  - Histograms
    - Computed by evaluating local sums on quantized images.
Recap: VeryFast Detector

- **Idea 1**: Invert the template scale vs. image scale relation

Recap: VeryFast Detector

• Idea 2: Reduce training time by feature interpolation

- Shown to be possible for Integral Channel features
Recap: VeryFast Classifier Construction

- Ensemble of short trees, learned by AdaBoost

\[ \text{score} = w_1 \cdot h_1 + w_2 \cdot h_2 + \ldots + w_N \cdot h_N \]
Recap: Elements of Tracking

- Detection
  - *Where are candidate objects?*

- Data association
  - *Which detection corresponds to which object?*

- Prediction
  - *Where will the tracked object be in the next time step?*
Today: Tracking with Linear Dynamic Models

Figure from Forsyth & Ponce
Topics of This Lecture

• Tracking with Dynamics
  – Detection vs. Tracking
  – Tracking as probabilistic inference
  – Prediction and Correction

• Linear Dynamic Models
  – Zero velocity model
  – Constant velocity model
  – Constant acceleration model

• The Kalman Filter
  – Kalman filter for 1D state
  – General Kalman filter
  – Limitations
Tracking with Dynamics

• **Key idea**
  – Given a model of expected motion, predict where objects will occur in next frame, even before seeing the image.

• **Goals**
  – Restrict search for the object
  – Improved estimates since measurement noise is reduced by trajectory smoothness.

• **Assumption: continuous motion patterns**
  – Camera is not moving instantly to new viewpoint.
  – Objects do not disappear and reappear in different places.
  – Gradual change in pose between camera and scene.
General Model for Tracking

• Representation
  – The moving object of interest is characterized by an underlying \textit{state} $X$.  
  – State $X$ gives rise to \textit{measurements} or \textit{observations} $Y$.  
  – At each time $t$, the state changes to $X_t$ and we get a new observation $Y_t$. 

\[ \begin{align*} 
  X_1 & \rightarrow X_2 \rightarrow \ldots \rightarrow X_t \\
  Y_1 & \rightarrow Y_2 \rightarrow \ldots \rightarrow Y_t 
\end{align*} \]
State vs. Observation

- Hidden state: parameters of interest
- Measurement: what we get to directly observe
Tracking as Inference

• Inference problem
  – The hidden state consists of the true parameters we care about, denoted $X$.
  – The measurement is our noisy observation that results from the underlying state, denoted $Y$.
  – At each time step, state changes (from $X_{t-1}$ to $X_t$) and we get a new observation $Y_t$.

• Our goal: recover most likely state $X_t$ given
  – All observations seen so far.
  – Knowledge about dynamics of state transitions.
Steps of Tracking

• Prediction:
  – What is the next state of the object given past measurements?
  \[ P(X_t|Y_0 = y_0, \ldots, Y_{t-1} = y_{t-1}) \]

• Correction:
  – Compute an updated estimate of the state from prediction and measurements.
  \[ P(X_t|Y_0 = y_0, \ldots, Y_{t-1} = y_{t-1}, Y_t = y_t) \]

• Tracking
  – Can be seen as the process of propagating the posterior distribution of state given measurements across time.
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
Tracking as Induction

- **Base case:**
  - Assume we have initial prior that *predicts* state in absence of any evidence: \( P(X_0) \)
  - At the first frame, *correct* this given the value of \( Y_0 = y_0 \)

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

- **Posterior prob. of state given measurement**
- **Likelihood of measurement**
- **Prior of the state**
• Base case:
  – Assume we have initial prior that *predicts* state in absence of any evidence: $P(X_0)$
  – At the first frame, *correct* this given the value of $Y_0 = y_0$

• Given corrected estimate for frame $t$:
  – Predict for frame $t+1$
  – Correct for frame $t+1$
Induction Step: 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**
\[
P(A) = \int P(A, B) dB
\]
Induction Step: 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 $X_{t-1}$**

\[
P(A, B) = P(A | B) P(B)
\]
Induction Step: 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
Induction Step: Correction

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

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

Bayes rule

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$
Induction Step: Correction

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

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

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

Independence assumption
(observation $y_t$ depends only on state $X_t$)
Induction Step: Correction

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

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

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

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

Conditioning on \( X_t \)}
Summary: Prediction and Correction

- Prediction:

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

Dynamics model \hspace{1cm} Corrected estimate from previous step
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
Topics of This Lecture

• Tracking with Dynamics
  – Detection vs. Tracking
  – Tracking as probabilistic inference
  – Prediction and Correction

• Linear Dynamic Models
  – Zero velocity model
  – Constant velocity model
  – Constant acceleration model

• The Kalman Filter
  – Kalman filter for 1D state
  – General Kalman filter
  – Limitations
Notation Reminder

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

- 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 1D, we just have one \( \Sigma \) parameter: the variance \( \sigma^2 \).
Linear Dynamic Models

- Dynamics model
  - State undergoes linear transformation $D_t$ plus Gaussian noise

\[
x_t \sim N \left( D_t x_{t-1}, \Sigma_{d_t} \right)
\]

- Observation model
  - Measurement is linearly transformed state plus Gaussian noise

\[
y_t \sim N \left( M_t x_t, \Sigma_{m_t} \right)
\]
Example: Randomly Drifting Points

• Consider a stationary object, with state as position.
  – Position is constant, only motion due to random noise term.

\[ x_t = p_t \quad p_t = p_{t-1} + \varepsilon \]

⇒ State evolution is described by identity matrix \( D=I \)

\[ x_t = D_t x_{t-1} + \text{noise} = Ip_{t-1} + \text{noise} \]
Example: Constant Velocity (1D Points)
Example: Constant Velocity (1D Points)

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

$$x_t = \begin{bmatrix} p_t \\ v_t \end{bmatrix}$$

$$p_t =$$

$$x_t = D_t x_{t-1} + noise =$$

- Measurement is position only

$$y_t = M x_t + noise =$$

(greek letters denote noise terms)
Example: Constant Velocity (1D Points)

- State vector: position $\mathbf{p}$ and velocity $\mathbf{v}$
  
  $$
  x_t = \begin{bmatrix} p_t \\ v_t \end{bmatrix}
  \quad
  p_t = p_{t-1} + (\Delta t)v_{t-1} + \varepsilon
  \quad
  v_{t-1} + \xi
  $$

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

- Measurement is position only
  
  $$
  y_t = M x_t + \text{noise} = \begin{bmatrix}
  1 & 0
  \end{bmatrix}
  \begin{bmatrix}
  p_t \\
  v_t
  \end{bmatrix}
  + \text{noise}
  $$

(greek letters denote noise terms)
Example: Constant Acceleration (1D Points)

![Graph showing constant acceleration in one-dimensional points.](image)

Position vs. time graph illustrating constant acceleration.
Example: Constant Acceleration (1D Points)

- State vector: position $p$, velocity $v$, and acceleration $a$.

\[
x_t = \begin{bmatrix} p_t \\ v_t \\ a_t \end{bmatrix}
\]

\[
p_t = p_{t-1} + (\Delta t)v_{t-1} + \frac{1}{2}(\Delta t)^2 a_{t-1} + \varepsilon
greek letters denote noise terms
\]

\[
x_t = D_t x_{t-1} + noise = \]

- Measurement is position only

\[
y_t = Mx_t + noise = \]
Example: Constant Acceleration (1D Points)

- State vector: position $p$, velocity $v$, and acceleration $a$.

\[
x_t = \begin{bmatrix} p_t \\ v_t \\ a_t \end{bmatrix} \quad \begin{aligned}
p_t &= p_{t-1} + (\Delta t)v_{t-1} + \frac{1}{2}(\Delta t)^2 a_{t-1} + \xi \\
v_t &= v_{t-1} + (\Delta t)a_{t-1} + \xi \\
a_t &= a_{t-1} + \xi
\end{aligned}
\]

\[
x_t = D_t x_{t-1} + \text{noise} = \begin{bmatrix} 1 & \Delta t & \frac{1}{2}(\Delta t)^2 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_{t-1} \\ v_{t-1} \\ a_{t-1} \end{bmatrix} + \text{noise}
\]

- Measurement is position only

\[
y_t = M x_t + \text{noise} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_t \\ v_t \\ a_t \end{bmatrix} + \text{noise}
\]
Recap: General Motion Models

- Assuming we have differential equations for the motion
  - E.g. for (undampened) periodic motion of a linear spring
    \[ \frac{d^2 p}{dt^2} = -p \]

- Substitute variables to transform this into linear system
  \[
  \begin{align*}
  p_1 &= p \\
  p_2 &= \frac{dp}{dt} \\
  p_3 &= \frac{d^2 p}{dt^2}
  \end{align*}
  \]

- Then we have
  \[
  x_t = \begin{bmatrix}
  p_{1,t} \\
  p_{2,t} \\
  p_{3,t}
  \end{bmatrix}
  \begin{align*}
  p_{1,t} &= p_{1,t-1} + (\Delta t) p_{2,t-1} + \frac{1}{2} (\Delta t)^2 p_{3,t-1} + \xi \\
  p_{2,t} &= p_{2,t-1} + (\Delta t) p_{3,t-1} + \xi \\
  p_{3,t} &= -p_{1,t-1} + \xi
  \end{align*}
  \]

\[
D_t = \begin{bmatrix}
1 & \Delta t & \frac{1}{2} (\Delta t)^2 \\
0 & 1 & \Delta t \\
-1 & 0 & 0
\end{bmatrix}
\]
Topics of This Lecture

• Tracking with Dynamics
  – Detection vs. Tracking
  – Tracking as probabilistic inference
  – Prediction and Correction

• Linear Dynamic Models
  – Zero velocity model
  – Constant velocity model
  – Constant acceleration model

• The Kalman Filter
  – Kalman filter for 1D state
  – General Kalman filter
  – Limitations
The Kalman Filter

• Kalman filter
  – Method for tracking linear dynamical models in Gaussian noise

• The predicted/corrected state distributions are Gaussian
  – You only need to maintain the mean and covariance.
  – The calculations are easy (all the integrals can be done in closed form).
The Kalman Filter

Know corrected state from previous time step, and all measurements up to the current one → Predict distribution over next state.

Receive measurement

Know prediction of state, and next measurement → Update distribution over current state.

Time update ("Predict")

| Mean and std. dev. of predicted state: |
| $\mu_t^-, \sigma_t^-$ |

Measurement update ("Correct")

| Mean and std. dev. of corrected state: |
| $\mu_t^+, \sigma_t^+$ |

Time advances: $t++$

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

$P(X_t|y_0, \ldots, y_t)$

Slide credit: Kristen Grauman
Kalman Filter for 1D State

- Want to represent and update

\[
P(x_t | y_0, \ldots, y_{t-1}) = N(\mu_t^-, (\sigma_t^-)^2)
\]

\[
P(x_t | y_0, \ldots, y_t) = N(\mu_t^+, (\sigma_t^+)^2)
\]
Propagation of Gaussian densities

- Shifting the mean
- Increasing the variance
- Bayesian update
- Deterministic drift
- Stochastic diffusion
- Reactive effect of measurement
1D Kalman Filter: Prediction

• Have linear dynamic model defining predicted state evolution, with noise

\[ X_t \sim N(dx_{t-1}, \sigma_d^2) \]

• Want to estimate predicted distribution for next state

\[ P(X_t | y_0, \ldots, y_{t-1}) = N(\mu_t^-, (\sigma_t^-)^2) \]

• Update the mean:

\[ \mu_t^- = d\mu_{t-1}^+ \]

• Update the variance:

\[ (\sigma_t^-)^2 = \sigma_d^2 + (d\sigma_{t-1}^+)^2 \]

for derivations, see F&P Chapter 17.3

Lecture: Computer Vision 2 (SS 2016) – Template-based Tracking
Prof. Dr. Bastian Leibe, Dr. Jörg Stückler

Slide credit: Kristen Grauman
1D Kalman Filter: Correction

- Have linear model defining the mapping of state to measurements:
  \[ Y_t \sim N(mx_t, \sigma_m^2) \]

- Want to estimate corrected distribution given latest measurement:
  \[ P(X_t | y_0, \ldots, y_t) = N(\mu_t^+, (\sigma_t^+)^2) \]

- Update the mean:
  \[ \mu_t^+ = \frac{\mu_t^- \sigma_m^2 + m y_t (\sigma_t^-)^2}{\sigma_m^2 + m^2 (\sigma_t^-)^2} \]

- Update the variance:
  \[ (\sigma_t^+)^2 = \frac{\sigma_m^2 (\sigma_t^-)^2}{\sigma_m^2 + m^2 (\sigma_t^-)^2} \]

Derivations: F&P Chapter 17.3
Prediction vs. Correction

\[
\mu_t^+ = \frac{\mu_t^- \sigma_m^2 + m y_t (\sigma_t^-)^2}{\sigma_m^2 + m^2 (\sigma_t^-)^2} \quad (\sigma_t^+)^2 = \frac{\sigma_m^2 (\sigma_t^-)^2}{\sigma_m^2 + m^2 (\sigma_t^-)^2}
\]

- What if there is no prediction uncertainty \((\sigma_t^- = 0)\)?
  \[
  \mu_t^+ = \mu_t^- \quad (\sigma_t^+)^2 = 0
  \]
  The measurement is ignored!

- What if there is no measurement uncertainty \((\sigma_m = 0)\)?
  \[
  \mu_t^+ = \frac{y_t}{m} \quad (\sigma_t^+)^2 = 0
  \]
  The prediction is ignored!

The measurement is ignored!

The prediction is ignored!
Recall: Constant Velocity Example

State is 2D: position + velocity
Measurement is 1D: position

Slide credit: Kristen Grauman
Constant Velocity Model

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

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

Slide credit: Kristen Grauman

Figure from Forsyth & Ponce
Constant Velocity Model

\[ \text{o state} \]

\[ \text{x measurement} \]

* predicted mean estimate

+ corrected mean estimate

bars: variance estimates before and after measurements
Constant Velocity Model

- State \( \mathbf{x} \)
- Measurement \( \mathbf{z} \)
- * predicted mean estimate
- + corrected mean estimate
- Bars: variance estimates before and after measurements

Slide credit: Kristen Grauman

Figure from Forsyth & Ponce
Kalman Filter: General Case (>1dim)

\[
x_t^- = D_t x_{t-1}^+
\]
\[
\Sigma_t^- = D_t \Sigma_{t-1}^+ D_t^T + \Sigma_d
\]

\[
K_t = \Sigma_t^- M_t^T (M_t \Sigma_t^- M_t^T + \Sigma_m)^{-1}
\]
\[
x_t^+ = x_t^- + K_t (y_t - M_t x_t^-)
\]
\[
\Sigma_t^+ = (I - K_t M_t) \Sigma_t^-
\]

More weight on residual when measurement error covariance approaches 0.

Less weight on residual as a priori estimate error covariance approaches 0.

for derivations, see F&P Chapter 17.3
Summary: Kalman Filter

• Pros:
  – Gaussian densities everywhere
  – Simple updates, compact and efficient
  – Very established method, very well understood

• Cons:
  – Unimodal distribution, only single hypothesis
  – Restricted class of motions defined by linear model
Why Is This A Restriction?

- Many interesting cases don’t have linear dynamics
  - E.g. pedestrians walking
  - E.g. a ball bouncing
Ball Example: What Goes Wrong Here?

- Assuming constant acceleration model

- Prediction is too far from true position to compensate…

- Possible solution:
  - Keep multiple different motion models in parallel
  - I.e. would check for bouncing at each time step
A very good introduction to tracking with linear dynamic models and Kalman filters can be found in Chapter 17 of