Computer Vision 2 – Lecture 6

Beyond Kalman Filters (09.05.2016)

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
Today: Beyond Gaussian Error Models
Topics of This Lecture

• Recap: Kalman Filter
  – Basic ideas
  – Limitations
  – Extensions

• Particle Filters
  – Basic ideas
  – Propagation of general densities
  – Factored sampling

• Case study
  – Detector Confidence Particle Filter
  – Role of the different elements
Recap: 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.
Recap: 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 $Y_0 = y_o$

• Given corrected estimate for frame $t$:
  – Predict for frame $t+1$
  – Correct for frame $t+1$
Recap: Prediction and Correction

- **Prediction:**

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

- **Correction:**

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

  \[
  y_t \sim N\left(M_t x_t, \Sigma_{m_t}\right)
  \]
Recap: Constant Velocity (1D Points)

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

- Measurement is position only
  \[
  y_t = Mx_t + noise = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p_{t-1} \\ v_{t-1} \end{bmatrix} + noise
  \]
Recap: Constant Acceleration (1D Points)

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

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

  $\begin{align*}
  p_t &= p_{t-1} + (\Delta t)v_{t-1} + \frac{1}{2}(\Delta t)^2 a_{t-1} + \epsilon \\
  v_t &= v_{t-1} + (\Delta t)a_{t-1} + \zeta \\
  a_t &= a_{t-1} + \zeta
  \end{align*}$

  (greek letters denote noise terms)

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

- Measurement is position only

  $y_t = Mx_t + noise = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_t \\ v_t \\ a_t \end{bmatrix} + noise$
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
  \[ p_1 = p \quad p_2 = \frac{dp}{dt} \quad p_3 = \frac{d^2 p}{dt^2} \]

- Then we have
  \[
  x_t = \begin{bmatrix}
  p_{1,t} \\
  p_{2,t} \\
  p_{3,t}
  \end{bmatrix}
  \]
  \[
  p_{1,t} = p_{1,t-1} + (\Delta t)p_{2,t-1} + \frac{1}{2}(\Delta t)^2 p_{3,t-1} + \varepsilon
  \]
  \[
  p_{2,t} = p_{2,t-1} + (\Delta t)p_{3,t-1} + \xi
  \]
  \[
  p_{3,t} = -p_{1,t-1} + \zeta
  \]
  \[
  D_t = \begin{bmatrix}
  1 & \Delta t & \frac{1}{2}(\Delta t)^2 \\
  0 & 1 & \Delta t \\
  -1 & 0 & 0
  \end{bmatrix}
  \]
Recap: 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:

\[ \mu_t^+, \sigma_t^+ \]

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

Time advances: \( t++ \)
Recap: General Kalman Filter (>1dim)

**PREDICT**

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

**CORRECT**

\[ K_t = \Sigma_t^- M_t^T \left( M_t \Sigma_t^- M_t^T + \Sigma_{m_t} \right)^{-1} \]
\[ x_t^+ = x_t^- + K_t \left( y_t - M_t x_t^- \right) \]
\[ \Sigma_t^+ = (I - K_t M_t) \Sigma_t^- \]

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
Resources: Kalman Filter Web Site

http://www.cs.unc.edu/~welch/kalman

• Electronic and printed references
  – Book lists and recommendations
  – Research papers
  – Links to other sites
  – Some software

• News

• Java-Based KF Learning Tool
  – On-line 1D simulation
  – Linear and non-linear
  – Variable dynamics
Remarks

• Try it!
  – Not too hard to understand or program

• Start simple
  – Experiment in 1D
  – Make your own filter in Matlab, etc.

• Note: the Kalman filter “wants to work”
  – Debugging can be difficult
  – Errors can go un-noticed
Topics of This Lecture

• Recap: Kalman Filter
  – Basic ideas
  – Limitations
  – Extensions

• Particle Filters
  – Basic ideas
  – Propagation of general densities
  – Factored sampling

• Case study
  – Detector Confidence Particle Filter
  – Role of the different elements
Extension: Extended Kalman Filter (EKF)

• Basic idea
  – State transition and observation model don’t need to be linear functions of the state, but just need to be differentiable.
    
    \[ x_t = g(x_{t-1}, u_t) + \varepsilon \]

    \[ y_t = h(x_t) + \delta \]
  – The EKF essentially linearizes the nonlinearity around the current estimate by a Taylor expansion.

• Properties
  – Unlike the linear KF, the EKF is in general not an optimal estimator.
    ▪ If the initial estimate is wrong, the filter may quickly diverge.
  – Still, it’s the de-facto standard in many applications
    ▪ Including navigation systems and GPS
Recap: Kalman Filter – Detailed Algorithm

- **Algorithm summary**
  - Assumption: linear model
    \[ x_t = D_t x_{t-1} + \varepsilon_t \]
    \[ y_t = M_t x_t + \delta_t \]
  - Prediction step
    \[ x_t^- = D_t x_{t-1}^+ \]
    \[ \Sigma_t^- = D_t \Sigma_t^+ D_t^T + \Sigma_{d_t} \]
  - Correction step
    \[ K_t = \Sigma_t^- M_t^T \left( M_t \Sigma_t^- M_t^T + \Sigma_{m_t} \right)^{-1} \]
    \[ x_t^+ = x_t^- + K_t (y_t - M_t x_t^-) \]
    \[ \Sigma_t^+ = (I - K_t M_t) \Sigma_t^- \]
Extended Kalman Filter (EKF)

• Algorithm summary
  – Nonlinear model
    \[ \mathbf{x}_t = g(\mathbf{x}_{t-1}) + \mathbf{\varepsilon}_t \]
    \[ \mathbf{y}_t = h(\mathbf{x}_t) + \mathbf{\delta}_t \]
  – Prediction step
    \[ \mathbf{x}_t^- = g(\mathbf{x}_t^+) \]
    \[ \Sigma_t^- = \mathbf{G}_t \Sigma_{t-1}^+ \mathbf{G}_t^T + \Sigma_{d_t} \]
  – Correction step
    \[ \mathbf{K}_t = \Sigma_t^- \mathbf{H}_t^T (\mathbf{H}_t \Sigma_t^- \mathbf{H}_t^T + \Sigma_{m_t})^{-1} \]
    \[ \mathbf{x}_t^+ = \mathbf{x}_t^- + \mathbf{K}_t (\mathbf{y}_t - h(\mathbf{x}_t^-)) \]
    \[ \Sigma_t^+ = (\mathbf{I} - \mathbf{K}_t \mathbf{H}_t) \Sigma_t^- \]

with the Jacobians

\[ \mathbf{G}_t = \left. \frac{\partial g(\mathbf{x})}{\partial \mathbf{x}} \right|_{\mathbf{x}=\mathbf{x}_{t-1}^+} \]
\[ \mathbf{H}_t = \left. \frac{\partial h(\mathbf{x})}{\partial \mathbf{x}} \right|_{\mathbf{x}=\mathbf{x}_t^-} \]
Kalman Filter – Other Extensions

• Unscented Kalman Filter (UKF)
  – Used for models with highly nonlinear predict and update functions.
  – Here, the EKF can give very poor performance, since the covariance is propagated through linearization of the non-linear model.
  – Idea (UKF): Propagate just a few sample points ("sigma points") around the mean exactly, then recover the covariance from them.
  – More accurate results than the EKF’s Taylor expansion approximation.

• Ensemble Kalman Filter (EnKF)
  – Represents the distribution of the system state using a collection (an ensemble) of state vectors.
  – Replace covariance matrix by sample covariance from ensemble.
  – Still basic assumption that all prob. distributions involved are Gaussian.
  – EnKFs are especially suitable for problems with a large number of variables.
Even More Extensions

Switching linear dynamical system (SLDS):

\[ z_t \sim \pi_{z_{t-1}} \]
\[ x_t = A^{(z_t)} x_{t-1} + e_t(z_t) \]
\[ y_t = C' x_t + w_t \]
\[ e_t \sim \mathcal{N}(0, \Sigma^{(z_t)}) \quad w_t \sim \mathcal{N}(0, R) \]

- **Switching Linear Dynamic System (SLDS)**
  - Use a set of \( k \) dynamic models \( A^{(1)}, ..., A^{(k)} \), each of which describes a different dynamic behavior.
  - Hidden variable \( z_t \) determines which model is active at time \( t \).
  - A switching process can change \( z_t \) according to distribution \( \pi_{z_{t-1}} \).
Topics of This Lecture

• Recap: Kalman Filter
  – Basic ideas
  – Limitations
  – Extensions

• Particle Filters
  – Basic ideas
  – Propagation of general densities
  – Factored sampling

• Case study
  – Detector Confidence Particle Filter
  – Role of the different elements

Today: only main ideas
Formal introduction next lecture
When Is A Single Hypothesis Too Limiting?

Consider this example: say we are tracking the face on the right using a skin color blob to get our measurement.

Video from Jojic & Frey

Slide credit: Kristen Grauman

Figure from Thrun & Kosecka
Propagation of General Densities
Factored Sampling

- Idea: Represent state distribution non-parametrically
  - Prediction: Sample points from prior density for the state, \( P(X) \)
  - Correction: Weight the samples according to \( P(Y|X) \)

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

• (Also known as Sequential Monte Carlo Methods)

• Idea
  – We want to use sampling to propagate densities over time (i.e., across frames in a video sequence).
  – At each time step, represent posterior $P(X_t|Y_t)$ with weighted sample set.
  – Previous time step’s sample set $P(X_{t-1}|Y_{t-1})$ is passed to next time step as the effective prior.
Particle Filtering

• Many variations, one general concept:
  – Represent the posterior pdf by a set of randomly chosen weighted samples (particles)

  – Randomly Chosen = Monte Carlo (MC)
  – As the number of samples become very large – the characterization becomes an equivalent representation of the true pdf.
Start with weighted samples from previous time step
Sample and shift according to dynamics model
Spread due to randomness; this is predicted density $P(X_t | Y_{t-1})$
Weight the samples according to observation density
Arrive at corrected density estimate $P(X_t | Y_t)$

M. Isard and A. Blake, **CONDENSATION -- conditional density propagation for visual tracking**, IJCV 29(1):5-28, 1998
Particle Filtering – Visualization

Code and video available from
http://www.robots.ox.ac.uk/~misard/condensation.html
Particle Filtering Results

http://www.robots.ox.ac.uk/~misard/condensation.html
Particle Filtering Results

• Some more examples

http://www.robots.ox.ac.uk/~misard/condensation.html
Obtaining a State Estimate

- Note that there’s no explicit state estimate maintained, just a “cloud” of particles
- Can obtain an estimate at a particular time by querying the current particle set
- Some approaches
  - “Mean” particle
    - Weighted sum of particles
    - Confidence: inverse variance
  - Really want a mode finder—mean of tallest peak
Condensation: Estimating Target State

State samples
(thickness proportional to weight)

Mean of weighted state samples

From Isard & Blake, 1998

Figures from Isard & Blake
Summary: Particle Filtering

• **Pros:**
  – Able to represent arbitrary densities
  – Converging to true posterior even for non-Gaussian and nonlinear system
  – Efficient: particles tend to focus on regions with high probability
  – Works with many different state spaces
    ▪ E.g. articulated tracking in complicated joint angle spaces
  – Many extensions available
Summary: Particle Filtering

• Cons / Caveats:
  – #Particles is important performance factor
    ▪ Want as few particles as possible for efficiency.
    ▪ But need to cover state space sufficiently well.
  – Worst-case complexity grows exponentially in the dimensions
  – Multimodal densities possible, but still single object
    ▪ Interactions between multiple objects require special treatment.
    ▪ Not handled well in the particle filtering framework (state space explosion).
Topics of This Lecture

• Recap: Kalman Filter
  – Basic ideas
  – Limitations
  – Extensions

• Particle Filters
  – Basic ideas
  – Propagation of general densities
  – Factored sampling

• Case study
  – Detector Confidence Particle Filter
  – Role of the different elements
Challenge: Unreliable Object Detectors

• Example:
  – Low-res webcam footage (320×240), MPEG compressed

Detector input

Tracker output

How to get from here…

? …to here?
Tracking based on Detector Confidence

- Detector output is often not perfect
  - Missing detections and false positives
  - But continuous confidence still contains useful cues.

- Idea pursued here:
  - Use continuous detector confidence to track persons over time.
Main Ideas

• Detector confidence particle filter
  – Initialize particle cloud on strong object detections.
  – Propagate particles using continuous detector confidence as observation model.

• Disambiguate between different persons
  – Train a person-specific classifier with online boosting.
  – Use classifier output to distinguish between nearby persons.
Detector Confidence Particle Filter

- State:
  \[ x = \{x, y, u, v\} \]

- Motion model (constant velocity)
  \[
  (x, y)_t = (x, y)_{t-1} + (u, v)_{t-1} \cdot \Delta t + \varepsilon(x, y) \\
  (u, v)_t = (u, v)_{t-1} + \varepsilon(u, v)
  \]

- Observation model
  \[
  w_{tr,p} = p(y_t | x_t^{(i)}) = \\
  \beta \cdot I(tr) \cdot p_N(p - d^*) + \gamma \cdot d_c(p) \cdot p_o(tr) + \eta \cdot c_{tr}(p)
  \]

Discrete detections
Detector confidence
Classifier confidence
When Is Which Term Useful?

Discrete detections

Detector confidence

Classifier confidence

Lecture: Computer Vision 2 (SS 2016) – Beyond Kalman Filters
Prof. Dr. Bastian Leibe, Dr. Jörg Stückler
Each Observation Term Increases Robustness!

<table>
<thead>
<tr>
<th>Observation Model Terms</th>
<th>MOTP</th>
<th>MOTA</th>
<th>FN</th>
<th>FP</th>
<th>ID Sw.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1: Det+DetConf+Class</td>
<td>70.0%</td>
<td>72.9%</td>
<td>26.8%</td>
<td>0.3%</td>
<td>0</td>
</tr>
<tr>
<td>2: Det+DetConf</td>
<td>64.0%</td>
<td>54.5%</td>
<td>28.2%</td>
<td>17.2%</td>
<td>5</td>
</tr>
<tr>
<td>3: Det+Class</td>
<td>65.0%</td>
<td>55.3%</td>
<td>31.3%</td>
<td>13.4%</td>
<td>0</td>
</tr>
<tr>
<td>4: Det</td>
<td>67.0%</td>
<td>40.9%</td>
<td>30.7%</td>
<td>28.0%</td>
<td>10</td>
</tr>
</tbody>
</table>

Detector only

CLEAR MOT scores
Each Observation Term Increases Robustness!

<table>
<thead>
<tr>
<th>Observation Model Terms</th>
<th>MOTP</th>
<th>MOTA</th>
<th>FN</th>
<th>FP</th>
<th>ID Sw.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1: Det+DetConf+Class</td>
<td>70.0%</td>
<td>72.9%</td>
<td>26.8%</td>
<td>0.3%</td>
<td>0</td>
</tr>
<tr>
<td>2: Det+DetConf</td>
<td>64.0%</td>
<td>54.5%</td>
<td>28.2%</td>
<td>17.2%</td>
<td>5</td>
</tr>
<tr>
<td>3: Det+Class</td>
<td>65.0%</td>
<td>55.3%</td>
<td>31.3%</td>
<td>13.4%</td>
<td>0</td>
</tr>
<tr>
<td>4: Det</td>
<td>67.0%</td>
<td>40.9%</td>
<td>30.7%</td>
<td>28.0%</td>
<td>10</td>
</tr>
</tbody>
</table>

Detector + Confidence

CLEAR MOT scores
**Each Observation Term Increases Robustness!**

<table>
<thead>
<tr>
<th>Observation Model Terms</th>
<th>MOTP</th>
<th>MOTA</th>
<th>FN</th>
<th>FP</th>
<th>ID Sw.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1: Det+DetConf+Class</td>
<td>70.0%</td>
<td>72.9%</td>
<td>26.8%</td>
<td>0.3%</td>
<td>0</td>
</tr>
<tr>
<td>2: Det+DetConf</td>
<td>64.0%</td>
<td>54.5%</td>
<td>28.2%</td>
<td>17.2%</td>
<td>5</td>
</tr>
<tr>
<td>3: Det+Class</td>
<td>65.0%</td>
<td>55.3%</td>
<td>31.3%</td>
<td>13.4%</td>
<td>0</td>
</tr>
<tr>
<td>4: Det</td>
<td>67.0%</td>
<td>40.9%</td>
<td>30.7%</td>
<td>28.0%</td>
<td>10</td>
</tr>
</tbody>
</table>

Detector + Classifier

CLEAR MOT scores
Each Observation Term Increases Robustness!

<table>
<thead>
<tr>
<th>Observation Model Terms</th>
<th>MOTP</th>
<th>MOTA</th>
<th>FN</th>
<th>FP</th>
<th>ID Sw.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1: Det+DetConf+Class</td>
<td>70.0%</td>
<td>72.9%</td>
<td>26.8%</td>
<td>0.3%</td>
<td>0</td>
</tr>
<tr>
<td>2: Det+DetConf</td>
<td>64.0%</td>
<td>54.5%</td>
<td>28.2%</td>
<td>17.2%</td>
<td>5</td>
</tr>
<tr>
<td>3: Det+Class</td>
<td>65.0%</td>
<td>55.3%</td>
<td>31.3%</td>
<td>13.4%</td>
<td>0</td>
</tr>
<tr>
<td>4: Det</td>
<td>67.0%</td>
<td>40.9%</td>
<td>30.7%</td>
<td>28.0%</td>
<td>10</td>
</tr>
</tbody>
</table>

Detector + Confidence + Classifier

False negatives, false positives, and ID switches decrease!

CLEAR MOT scores
Qualitative Results
Remaining Issues

• Some false positive initializations at wrong scales…
  – Due to limited scale range of the person detector.
  – Due to boundary effects of the person detector.
• A good tutorial on Particle Filters

• The CONDENSATION paper