CS231A

Computer Vision: From 3D Reconstruction to Recognition

Optimal Estimation Cont’
Recap

• Recursive Filter
• Kalman Filter
• Extended Kalman Filter
Nonparametric filters

• No fixed functional form of the posterior – can capture multimodality
• Instead: finite numbers of values

• Histogram filter: State = finitely many regions
• Particle filter: Distribution represented by samples
Particle Filter

\[ p(x_t | z_{t:1}, u_{t:1}, x_0) \rightarrow X_t = \{ x_t^0, \ldots, x_t^N \} \]
The Particle filter algorithm

Algorithm Particle\_filter(\mathcal{X}_{t-1}, u_t, z_t):

1. \( \bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset \)
2. for \( m = 1 \) to \( M \) do
3. sample \( x_t^{[m]} \sim p(x_t \mid u_t, x_{t-1}^{[m]} \)
4. \( w_t^{[m]} = p(z_t \mid x_t^{[m]} \)
5. \( \bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \langle x_t^{[m]}, w_t^{[m]} \rangle \)
6. endfor
7. for \( m = 1 \) to \( M \) do
8. draw \( i \) with probability \( \propto w_t^{[i]} \)
9. add \( x_t^{[i]} \) to \( \mathcal{X}_t \)
10. endfor
11. return \( \mathcal{X}_t \)
12. Process Model
13. Measurement Model
14. Importance Sampling

Before resampling: \( \overline{\text{bel}}(x_t) = \int p(x_t \mid u_t, x_{t-1}) \overline{\text{bel}}(x_{t-1}) \, dx \)

After resampling: \( \overline{\text{bel}}(x_t) = \eta \int \frac{p(z_t \mid x_t)}{\overline{\text{bel}}(x_t)} \overline{\text{bel}}(x_t) \, dx \)
When to Use Each?

Bayes Filter
- General Framework
- No implementation!

Kalman Filter
- Linear Models
- Gaussian Distributions

Extended Kalman Filter
- Non-Linear Models (linearizable)
- Gaussian Distributions

Particle Filter
- Any Model
- Any Distribution
- Low Dimensional State Space
Graphical Model of System to Estimate

\[ z(t) = h(x(t)) \]

1: **Algorithm Bayes filter**\( \text{bel}(x_{t-1}, u_t, z_t) \):
2: \quad for all \( x_t \) do
3: \quad \quad \text{bel}(x_t) = \int p(x_t \mid u_t, x_{t-1}) \text{bel}(x_{t-1}) \, dx
4: \quad \quad \text{bel}(x_t) = \eta p(z_t \mid x_t) \text{bel}(x_t)
5: \quad endfor
6: \quad return \text{bel}(x_t)
Example Observation model for 3D object

Algorithm Particle filter(\(X_{t-1}, u_t, z_t\)):

1. \(X_t = X_t = \emptyset\)
2. for \(m = 1\) to \(M\) do
   1. sample \(x_t^{[m]} \sim p(x_t | u_t, x_t^{[m-1]}\))
   2. \(w_t^{[m]} = p(z_t | x_t^{[m]}\))
   3. \(X_t = X_t + \{x_t^{[m]}, w_t^{[m]}\}\)
3. endfor
4. for \(m = 1\) to \(M\) do
   1. draw \(i\) with probability \(w_t^{[i]}\)
   2. add \(x_t^{[i]}\) to \(X_t\)
5. endfor
6. return \(X_t\)

Tracking by Detection

Control Input:
- $u(t - 1)$
- $u(t)$
- $u(t + 1)$

State:
- $x(t - 1)$
- $x(t)$
- $x(t + 1)$

Observation Model:
- $z(t) = x(t)$

Observation:
- $z(t - 1)$
- $z(t)$
- $z(t + 1)$

Detections:
- $g(I(t - 1))$
- $g(I(t))$
- $g(I(t + 1))$

Object Detector: Input Sensory Data
Problem Statement: Input

Probabilistic 3d multi-object tracking for autonomous driving. H Chiu, A Prioretti, J Li, J Bohg

- Object detections at each frame in a sequence
- Each detection bounding box is represented by:
  - center position \((x, y, z)\), rotation angle along the z-axis \((a)\), and the scale \((l, w, h)\)
  - category label (car, pedestrian, ...), confidence score \((c)\)
Problem Statement: Output

- Tracking object bounding boxes at each frame in a sequence
- Each tracking bounding box is represented by:
  - center position \((x, y, z)\), rotation angle along the z-axis \((a)\), and the scale \((l, w, h)\)
  - category label (car, pedestrian, ...), confidence score \((c)\)
  - tracking id: one unique tracking id for each object instance across frames
Why Tracking?

- Filter out the out-liners from the detection results
- Continue estimating object states even if occluded
- Forecast the future based on past trajectories and motion patterns
- Make autonomous driving decisions
Our Proposed Method

Lecture 14
Kalman Filter for Tracking

Define the object **state** using a vector of random variables including the position, the rotation, the scale, linear velocity, and the angular velocity.

\[ s_t = (x, y, z, a, l, w, h, d_x, d_y, d_z, d_a)^T \]

Define the **Process Model** for prediction based on the constant velocity motion:

\[
\begin{align*}
\hat{x}_{t+1} &= x_t + d_{x_t} + q_{x_t}, \\
\hat{y}_{t+1} &= y_t + d_{y_t} + q_{y_t}, \\
\hat{z}_{t+1} &= z_t + d_{z_t} + q_{z_t}, \\
\hat{a}_{t+1} &= a_t + d_{a_t} + q_{a_t}, \\
\end{align*}
\]

\[
\begin{align*}
\hat{d}_{x_{t+1}} &= d_{x_t} + q_{d_{x_t}}, \\
\hat{d}_{y_{t+1}} &= d_{y_t} + q_{d_{y_t}}, \\
\hat{d}_{z_{t+1}} &= d_{z_t} + q_{d_{z_t}}, \\
\hat{d}_{a_{t+1}} &= d_{a_t} + q_{d_{a_t}}, \\
\end{align*}
\]

\[
\begin{align*}
\hat{l}_{t+1} &= l_t, \\
\hat{w}_{t+1} &= w_t, \\
\hat{h}_{t+1} &= h_t
\end{align*}
\]
Our Proposed Method

Predictions

Matching By Mahalanobis Distance

KalmanFilter Predict

KalmanFilter Update

Tracking Result

Tracking ID 1
Tracking ID 2
Tracking ID 3
Tracking ID 4
Tracking ID 5
Tracking ID 6
Data Association

Mahalanobis Distance \( m = \sqrt{(z_t - C\mu_t)^T S_t^{-1} (z_t - C\mu_t)} \)

- \( S = \) Innovation Covariance
- \( z_t - C\mu_t = \) innovation
Kalman Filter

1: **Algorithm Kalman filter**$(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t)$:

2: \[ \bar{\mu}_t = A_t \mu_{t-1} + B_t u_t \]

3: \[ \bar{\Sigma}_t = A_t \Sigma_{t-1} A^T_t + R_t \]

4: \[ K_t = \bar{\Sigma}_t C^T_t \left( C_t \bar{\Sigma}_t C^T_t + Q_t \right)^{-1} = S_t^{-1} \]

5: \[ \mu_t = \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t) \]

6: \[ \Sigma_t = (I - K_t C_t) \bar{\Sigma}_t \]

7: return $\mu_t, \Sigma_t$
Data Association

Mahalanobis Distance \( m = \sqrt{(z_t - C\mu_t)^T S_t^{-1} (z_t - C\mu_t)} \)

If \( m > 3 \times \sigma \) then reject as outlier. 99.7% of values lie within 3*standard deviation.

Measuring the distance between the observation and the distribution of the predicted state.

Providing distance measurement when there is no overlap between the prediction and detection.

Taking the uncertainty information from the prediction into account.
Data Association - Greedy

Kalman Filter Predictions

Detections
Qualitative Results
Qualitative Results

**AB3DMOT baseline**

**Result**
Qualitative Results

w/o angular velocity

Result
Qualitative Results

Ground-truth

Result
Priors and Hyperparameters

A lot of hardcoded knowledge!

• State Representation
• Models
  • Forward Model
    • State to next state
    • Action to next state
  • Measurement Model
• Probabilistic Properties
  • Process Noise
  • Measurement Noise
Differentiable filters

Can we learn models and hyperparameters from data?

Approach: Embed algorithmic structure of Bayesian Filtering into a recurrent neural network.

- prevents overfitting through regularization
- Avoids manual tuning and modeling

- Differentiable version of the Kalman Filter
- Uses Images as observations; learns a sensors that outputs state directly

\[ g(I_t) = z_t \approx x_t \]

Example Sequence w/ few occlusions
Example Sequence w/ many occlusions

Track red disk position

\[
\begin{align*}
  t & \quad t + 1
\end{align*}
\]
Differentiable Kalman Filter - Structure

\[ g(I_t) = z_t \approx x_t \]

R is high if red disk is occluded
Differentiable Kalman Filter - Structure

\[ L' L^T = R \]
Differentiable Kalman Filter – Loss Function

\[ L(l_0...T, \mu_0...T, \Sigma_0...T, w) = \]

\[ \lambda_1 \sum_{t=0}^{T} \frac{1}{2} (l_t - \mu_t)^T \Sigma_t^{-1} (l_t - \mu_t) + \log(|\Sigma_t|) + \lambda_2 \sum_{t=0}^{T} \| l_t - \mu_t \|_2 + \lambda_3 \| w \|_2 \]

Ground truth state \quad Network weights

Negative log likelihood of ground truth given current belief \quad Mean-Squared Error \quad Regularization
# Differentiable Kalman Filter – Experiments and Baselines

## Table 1: Benchmark Results

<table>
<thead>
<tr>
<th>State Estimation Model</th>
<th># Parameters</th>
<th>RMS test error ±σ</th>
</tr>
</thead>
<tbody>
<tr>
<td>feedforward model</td>
<td>7394</td>
<td>0.2322 ± 0.1316</td>
</tr>
<tr>
<td>piecewise KF</td>
<td>7397</td>
<td>0.1160 ± 0.0330</td>
</tr>
<tr>
<td>LSTM model (64 units)</td>
<td>33506</td>
<td>0.1407 ± 0.1154</td>
</tr>
<tr>
<td>LSTM model (128 units)</td>
<td>92450</td>
<td>0.1423 ± 0.1352</td>
</tr>
<tr>
<td><strong>BKF (ours)</strong></td>
<td><strong>7493</strong></td>
<td><strong>0.0537 ± 0.1235</strong></td>
</tr>
</tbody>
</table>
Differentiable Kalman Filter – Experiments and Baselines

- Kitti – Visual Odometry Dataset
- 22 stereo sequences with LIDAR
  - 11 sequences with ground truth (GPS/IMU data)
  - 11 sequences without ground truth (for evaluation)
Differentiable Kalman Filter – Experiments and Baselines

Results reproduced by Claire Chen

Blue – Result of BackpropKF
Red – Ground truth
Green – w/ Ground truth velocities
CS231
Introduction to Computer Vision

Next lecture:
Neural Radiance Fields for Novel View Synthesis