Lecture-14

Kalman Filter

Main Points

• Very useful tool.
• It produces an optimal estimate of the state vector based on the noisy measurements (observations).
• For the state vector it also provides confidence (certainty) measure in terms of a covariance matrix.
• It integrates estimate of state over time.
• It is a sequential state estimator.
State-Space Model

State-transition equation

\[ z(k) = \Phi(k, k-1)z(k-1) + w(k) \]

Measurement (observation) equation

\[ y(k) = H(k)z(k) + v(k) \]

Kalman Filter Equations

State Prediction

\[ \hat{z}_b(k) = \Phi(k, k-1)\hat{z}_a(k-1) \]

Covariance Prediction

\[ p_b(k) = \Phi(k, k-1)p_a(k-1)\Phi^T(k, k-1) + Q(k) \]

Kalman Gain

\[ K(k) = p_b(k)H^T(k)(H(k)p_b(k)H^T(k) + R(k))^{-1} \]

State-update

\[ \hat{z}_a(k) = \hat{z}_b(k) + K(k)[y(k) - H(k)\hat{z}_b(k)] \]

Covariance-update

\[ p_a(k) = p_b(k) - K(k)H(k)p_b(k) \]

Copyright Mubarak Shah 2003
Two Special Cases

- Steady State
  \[ \Phi(k, k - 1) = \Phi \]
  \[ Q(k) = Q \]
  \[ H(k) = H \]
  \[ R(k) = R \]

- Recursive least squares
  \[ \Phi(k, k - 1) = I \]
  \[ Q(k) = 0 \]

Comments

- In some cases, state transition equation and the observation equation both may be non-linear.
- We need to linearize these equation using Taylor series.
Extended Kalman Filter

\[ z(k) = f(z(k-1)) + w(k) \]

\[ y(k) = h(z(k)) + v(k) \]

\[ f(z(k-1)) \approx f(\hat{z}_a(k-1)) + \frac{\partial f(z(k-1))}{\partial z(k-1)} (z(k-1) - \hat{z}_a(k-1)) \]

Taylor series

\[ h(z(k)) \approx h(\hat{z}_b(k)) + \frac{\partial h(z(k))}{\partial z(k)} (z(k) - \hat{z}_b(k-1)) \]

Copyright Mubarak Shah 2003

---

Extended Kalman Filter

\[ z(k) = f(z(k-1)) + w(k) \]

\[ z(k) = f(\hat{z}_a(k-1)) + \frac{\partial f(z(k-1))}{\partial z(k-1)} (z(k-1) - \hat{z}_a(k-1)) + w(k) \]

\[ z(k) \approx \Phi(k,k-1)z(k-1) + u(k) + w(k) \]

\[ u(k) = f(\hat{z}_a(k-1)) - \Phi(k,k-1)\hat{z}_a(k-1) \]

\[ \Phi(k,k-1) = \frac{\partial f(z(k-1))}{\partial z(k-1)} \]

Copyright Mubarak Shah 2003
Extended Kalman Filter

\[ y(k) = h(z(k)) + v(k) \]
\[ y(k) = h(\hat{z}_b(k)) + \frac{\partial h(z(k))}{\partial z(k)}(z(k) - \hat{z}_b(k-1)) + v(k) \]
\[ \tilde{y}(k) \approx H(k)z(k) + v(k) \]
\[ \tilde{y}(k) = y(k) - h(\hat{z}_b(k)) + H(k)\hat{z}_b(k) \]
\[ H(k) = \frac{\partial h(z(k))}{\partial z(k)} \]

Multi-Frame Feature Tracking

Application of Kalman Filter

Copyright Mubarak Shah 2003
• Assume feature points have been detected in each frame.
• We want to track features in multiple frames.
• Kalman filter can estimate the position and uncertainty of feature in the next frame.
  – Where to look for a feature
  – how large a region should be searched

\[
p_k = \begin{bmatrix} x_k, y_k \end{bmatrix}^T \quad \text{Location}
\]

\[
v_k = \begin{bmatrix} u_k, v_k \end{bmatrix}^T \quad \text{Velocity}
\]

\[
Z = \begin{bmatrix} x_k, y_k, u_k, v_k \end{bmatrix}^T \quad \text{State Vector}
\]
System Model

\[ \mathbf{p}_k = \mathbf{p}_{k-1} + \mathbf{v}_{k-1} + \zeta_{k-1} \]
\[ \mathbf{v}_k = \mathbf{v}_{k-1} + \eta_{k-1} \]
\[ \mathbf{Z}_k = \Phi_{k-1} \mathbf{Z}_{k-1} + \mathbf{w}_{k-1} \]
\[ \Phi_{k-1} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \]
\[ \mathbf{w}_{k-1} = \begin{bmatrix} \zeta_{k-1} \\ \eta_{k-1} \end{bmatrix} \]

Measurement Model

\[ \mathbf{y}_k = \begin{bmatrix} 1 & 0 & 0 & 0 & \mathbf{p}_k \end{bmatrix} + \mu_k \]
\[ \mathbf{y}_k = \mathbf{H} \begin{bmatrix} \mathbf{p}_k \\ \mathbf{v}_k \end{bmatrix} + \mu_k \]

Copyright Mubarak Shah 2003
## Kalman Filter Equations

<table>
<thead>
<tr>
<th>Description</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>State Prediction</td>
<td>( \hat{\mathbf{z}}_b(k) = \Phi(k, k-1)\hat{\mathbf{z}}_a(k-1) )</td>
</tr>
<tr>
<td>Covariance Prediction</td>
<td>( \mathbf{P}_b(k) = \Phi(k, k-1)\mathbf{P}_a(k-1)\Phi^T(k, k-1) + \mathbf{Q}(k) )</td>
</tr>
<tr>
<td>Kalman Gain</td>
<td>( \mathbf{K}(k) = \mathbf{P}_b(k)\mathbf{H}^T(k)(\mathbf{H}(k)\mathbf{P}_b(k)\mathbf{H}^T(k) + \mathbf{R}(k))^{-1} )</td>
</tr>
<tr>
<td>State-update</td>
<td>( \hat{\mathbf{z}}_a(k) = \hat{\mathbf{z}}_b(k) + \mathbf{K}(k)\mathbf{y}(k) - \mathbf{H}(k)\hat{\mathbf{z}}_b(k) )</td>
</tr>
<tr>
<td>Covariance-update</td>
<td>( \mathbf{P}_a(k) = \mathbf{P}_b(k) - \mathbf{K}(k)\mathbf{H}(k)\mathbf{P}_b(k) )</td>
</tr>
</tbody>
</table>

Copyright  Mubarak Shah 2003

Copyright  Mubarak Shah 2003
Kalman Filter: Relation to Least Squares

\[ f_i(Z, y_i) = 0 \]

Taylor series

\[ f_i(Z, y_i) \approx f_i(\hat{Z}_{i-1}, \hat{y}_i) + \frac{\partial f_i}{\partial y} (y - \hat{y}_i) + \frac{\partial f_i}{\partial z} (z - \hat{z}_i) + w_i \]

\[ Y_i = H_i Z + w_i \]

\[ Y_i = -f_i(\hat{Z}_{i-1}, \hat{y}_i) + \frac{\partial f_i}{\partial z} \hat{z}_{i-1}, H_i = \frac{\partial f_i}{\partial z} \]

\[ w_i = \frac{\partial f_i}{\partial y} (y - \hat{y}_i) \]

Kalman Filter: Relation to Least Squares

Estimate state such that the following is minimized:
- first term: initial estimate weighted by corresponding covariance
- second term: other measurements weighted by corresponding covariances

\[ C = (\hat{Z}_0 - Z)^T P_0^{-1} (\hat{Z}_0 - Z) + \sum_{i=1}^{k} (Y_i - H_i Z)^T W_i^{-1} (Y_i - H_i Z) \]

minimize

\[ \hat{Z} = [P_0^{-1} + \sum_{i=1}^{k} H_i^T W_i^{-1} H_i]^{-1} [P_0^{-1} \hat{Z}_0 + \sum_{i=1}^{k} H_i^T W_i^{-1} Y_i] \]

Batch Mode

Copyright Mubarak Shah 2003
Kalman Filter: Relation to Least Squares

\[ \hat{Z}_k = \left[ P_0^{-1} + \sum_{i=1}^{k} H_i^T W_i^{-1} H_i \right]^{-1} \left[ P_0^{-1} \hat{Z}_0 + \sum_{i=1}^{k} H_i^T W_i^{-1} Y_i \right] \]

Recursive Mode

Kalman Filter: Relation to Least Squares

\[ Z_k = Z_{k-1} + K_k (Y_k - H_k Z_{k-1}) \]

\[ K_k = P_{k-1} H_k^T \left( W_k + H_k P_{k-1} H_k^T \right)^{-1} \]

\[ P_k = (I - K_k H_k) P_{k-1} \]

\[ Y_k = -f^T (Z_{k-1}, Y_{k-1}) + \frac{\partial f}{\partial Z} Z_{k-1} \]

\[ \Phi(k, k-1) = I \]

\[ Q(k) = 0 \]

\[ H_k = \frac{\partial f}{\partial Z} \]

\[ W^k = \frac{\partial f}{\partial Y} A_k \frac{\partial f}{\partial Y}^T \]

Covariance matrix for measurement Vector y

Copyright Mubarak Shah 2003
Kalman Filter (Least Squares)

State Prediction
\[ \hat{z}_b(k) = \Phi(k,k-1)\hat{z}_a(k-1) \]
\[ \hat{z}_b(k) = \hat{z}_a(k-1) \]

Covariance Prediction
\[ P_b(k) = \Phi(k,k-1)P_a(k-1)\Phi^T(k,k-1) + Q(k) \]
\[ P_b(k) = P_a(k-1) \]

Kalman Gain
\[ K(k) = P_b(k)H^T(k)(H(k)P_b(k)H^T(k) + R(k))^{-1} \]
\[ K(k) = P_b(k)H^T(k)(H(k)P_b(k)H^T(k) + W(k))^{-1} \]

State-update
\[ \hat{z}_a(k) = \hat{z}_b(k) + K(k)[y(k) - H(k)\hat{z}_b(k)] \]
\[ \hat{z}(k) = \hat{z}(k-1) + K(k)[y(k) - H(k)\hat{z}(k-1)] \]

Covariance-update
\[ P_a(k) = P_b(k) - K(k)H(k)P_b(k) \]
\[ P(k) = P(k-1) - K(k)H(k)P(k-1) \]
Computing Motion Trajectories

Algorithm For Computing Motion Trajectories

- Compute tokens using Moravec’s interest operator (intensity constraint).
- Remove tokens which are not interesting with respect to motion (optical flow constraint).
  - Optical flow of a token should differ from the mean optical flow around a small neighborhood.
Algorithm For Computing Motion Trajectories

• Link optical flows of a token in different frames to obtain motion trajectories.
  – Use optical flow at a token to predict its location in the next frame.
  – Search in a small neighborhood around the predicted location in the next frame for a token.

• Smooth motion trajectories using Kalman filter.

Kalman Filter (Ballistic Model)

\[ x(t) = 0.5a_x t^2 + v_x t + x_0 \quad Z = (a_x, a_y, v_x, v_y) \]
\[ y(t) = 0.5a_y t^2 + v_y t + y_0 \quad y = (x(t), y(t)) \]

\[ f(Z, y) = (x(t) - 0.5a_x t^2 - v_x t - x_0, y(t) - 0.5a_y t^2 - v_y t - y_0) \]
Kalman Filter (Ballistic Model)

\[ Z(k) = Z(k-1) + K(k)(Y(k) - H(k)Z(k-1)) \]

\[ K(k) = P(k-1)H^T(k) \left( W(k) + H^T P(k-1)H^T(k) \right)^{-1} \]

\[ P(k) = (I - K(k)H(k))P(k-1) \]

\[ Y(k) = -f^T(Z(k-1), y) + \frac{\partial f}{\partial Z} Z(k-1) \]

\[ H(k) = \frac{\partial f}{\partial Z} \]

\[ W(k) = \frac{\partial f}{\partial y} A(k) \frac{\partial f}{\partial y}^T \]

Copyright Mubarak Shah 2003