SEIF, EnKF, EKF SLAM

Pieter Abbeel
UC Berkeley EECS
Information Filter

- From an analytical point of view == Kalman filter

- Difference: keep track of the inverse covariance rather than the covariance matrix [matter of some linear algebra manipulations to get into this form]

- Why interesting?
  - Inverse covariance matrix = 0 is easier to work with than covariance matrix = infinity (case of complete uncertainty)
  - Inverse covariance matrix is often sparser than the covariance matrix --- for the “insiders”: inverse covariance matrix entry (i,j) = 0 if \( x_i \) is conditionally independent of \( x_j \) given some set \( \{x_k, x_l, \ldots\} \)
  - Downside: when extended to non-linear setting, need to solve a linear system to find the mean (around which one can then linearize)

- See Probabilistic Robotics pp. 78-79 for more in-depth pros/cons and Probabilistic Robotics Chapter 12 for its relevance to SLAM (then often referred to as the “sparse extended information filter (SEIF)”)
Represent the Gaussian distribution by samples

- Empirically: even 40 samples can track the atmospheric state with high accuracy with enKF
- \( \leftarrow \rightarrow \) UKF: \( 2 \times n \) sigma-points, \( n = 10^6 \) + then still forms covariance matrices for updates

The intellectual innovation:

- Transforming the Kalman filter updates into updates which can be computed based upon samples and which produce samples while never explicitly representing the covariance matrix

Ensemble Kalman filter (enKF)
**KF**

Keep track of $\mu, \Sigma$

**Prediction:**

\[
\hat{\mu}_t = A_t \mu_{t-1} + B_t u_t \\
\hat{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t
\]

**Correction:**

\[
K_t = \hat{\Sigma}_t C_t^T (C_t \hat{\Sigma}_t C_t^T + \bar{Q}_t)^{-1} \\
\mu_t = \mu_{t-1} + K_t (z_t - C_t \mu_{t-1}) \\
\Sigma_t = (I - K_t C_t) \hat{\Sigma}_t
\]

**Return** $\mu_t, \Sigma_t$

---

**enKF**

Keep track of ensemble $[x_1, \ldots, x_N]$

Can update the ensemble by simply propagating through the dynamics model + adding sampled noise

?
enKF correction step

- **KF:**
  \[
  K_t = \Sigma_t C_t^T (C_t \Sigma_t C_t^T + Q_t)^{-1}
  \]
  \[
  \mu_t = \mu_t + K_t (z_t - C_t \mu_t)
  \]
  \[
  \Sigma_t = (I - K_tC_t)\Sigma_t
  \]

- Current ensemble \(X = [x_1, \ldots, x_N]\)

- Build observations matrix \(Z = [z_t + v_1 \ldots z_t + v_N]\) where \(v_i\) are sampled according to the observation noise model

- Then the columns of
  \[
  X + K_t(Z - C_t X)
  \]
  form a set of random samples from the posterior

Note: when computing \(K_t\), leave \(\Sigma_t\) in the format
\[
\Sigma_t = [x_1-\mu_t \ldots x_N-\mu_t] [x_1-\mu_t \ldots x_N-\mu_t]^T
\]
How about C?

- Indeed, would be expensive to build up C.
- However: careful inspection shows that C only appears as in:
  - $C X$
  - $C \sum C^T = C X X^T C^T$

- → can simply compute $h(x)$ for all columns $x$ of $X$ and compute the empirical covariance matrices required

- [details left as exercise]
Are the columns of \( X + K^t (Z - C^t X) \) really sampled from \( N(\mu^t, \Sigma_t) \)?

One column:
\[
y^{(i)} = x^{(i)} + k^t (z^{(i)} + u^{(i)} - C_t x^{(i)})
\]
where
\[
x^{(i)} \sim N(\mu^t, \Sigma_t) \quad u^{(i)} \sim N(0, \Theta_t)
\]

1. \[
E[y^{(i)}] = \mu^t + k^t (z^t + o - C_t \mu^t)
\]
\[
= \mu^t + k^t (z^t - C_t \mu^t)
\]
\[
= \mu^t
\]

2. \[
E\left[ \begin{pmatrix} y^{(i)} - E[y^{(i)}] \\ E[y^{(i)}] - E[y^{(i)}]^T \end{pmatrix} \right]
\]
\[
= E\left[ \begin{pmatrix} \left( x^{(i)} + k^t (z^{(i)} + u^{(i)} - C_t x^{(i)}) - (\mu^t + k^t (z^t - C_t \mu^t)) \right) \\ \left( \frac{1}{2} \right) \end{pmatrix} \right]
\]
\[
= E\left[ \begin{pmatrix} (I - K_t C_t)(z^{(i)} - \mu^t) + k^t u^{(i)} \end{pmatrix} \right] \left( \begin{pmatrix} (I - K_t C_t)^T \end{pmatrix} \right)
\]

\( u^{(i)} \) and \( x^{(i)} \) independent

\[
= E\left[ (I - K_t C_t)(z^{(i)} - \mu^t)(z^{(i)} - \mu^t)^T (I - K_t C_t)^T \right] + E\left[ k^t u^{(i)}(u^{(i)})^T k^T \right]
\]
\[
= (I - K_t C_t) \Sigma_t (I - K_t C_t)^T + K_t Q_t K^T
\]
\[
= \Sigma_t + K_t C_t \Sigma_t K_t^T - K_t C_t \Sigma_t K_t^T + K_t Q_t K^T
\]
\[
K_t = \Sigma_t C_t (C_t \Sigma_t C_t + Q_t)^{-1}
\]
\[
= \Sigma_t + \Sigma_t C_t K_t^T - K_t C_t \Sigma_t - \Sigma_t C_t K_t^T
\]
\[
= \Sigma_t - K_t C_t \Sigma_t = \Sigma_t \quad Q.E.D.
\]
References for enKF

- Mandel, 2007 “A brief tutorial on the Ensemble Kalman Filter”
- Evensen, 2009, “The ensemble Kalman filter for combined state and parameter estimation”
KF Summary

- Kalman filter exact under linear Gaussian assumptions
- Extension to non-linear setting:
  - Extended Kalman filter
  - Unscented Kalman filter
- Extension to extremely large scale settings:
  - Ensemble Kalman filter
  - Sparse Information filter
- Main limitation: restricted to unimodal / Gaussian looking distributions
- Can alleviate by running multiple XKFs + keeping track of the likelihood; but this is still limited in terms of representational power unless we allow a very large number of them
**EKF/UKF SLAM**


- Now map = location of landmarks (vs. gridmaps)

Transition model:
- Robot motion model; Landmarks stay in place
Simultaneous Localization and Mapping (SLAM)

- In practice: robot is not aware of all landmarks from the beginning

- Moreover: no use in keeping track of landmarks the robot has not received any measurements about

  → Incrementally grow the state when new landmarks get encountered.
Simultaneous Localization and Mapping (SLAM)

- Landmark measurement model: robot measures \([x_k; y_k]\), the position of landmark \(k\) expressed in coordinate frame attached to the robot:
  
  \[
  h(n_R, e_R, \theta_R, n_k, e_k) = [x_k; y_k] = R(\theta) ( [n_k; e_k] - [n_R; e_R] )
  \]

- Often also some odometry measurements
  
  - E.g., wheel encoders
  
  - As they measure the control input being applied, they are often incorporated directly as control inputs (why?)
Victoria Park Data Set

[courtesy by E. Nebot]
Victoria Park Data Set Vehicle

[courtesy by E. Nebot]
Data Acquisition

[courtesy by E. Nebot]
Estimated Trajectory

[courtesy by E. Nebot]
EKF SLAM Application

[courtesy by J. Leonard]
EKF SLAM Application

odometry

estimated trajectory

[courtesy by John Leonard]
Landmark-based Localization
EKF-SLAM: practical challenges

- Defining landmarks
  - Laser range finder: Distinct geometric features (e.g. use RANSAC to find lines, then use corners as features)
  - Camera: “interest point detectors”, textures, color, …

- Often need to track multiple hypotheses
  - Data association/Correspondence problem: when seeing features that constitute a landmark --- Which landmark is it?
  - Closing the loop problem: how to know you are closing a loop?
    - Can split off multiple EKFs whenever there is ambiguity;
    - Keep track of the likelihood score of each EKF and discard the ones with low likelihood score

- Computational complexity with large numbers of landmarks.