RECIPEs

- Introduction
- Mean and Median Filters
- Kalman Filter
- Particle Filter
This chapter illustrates the processing of sequential sensor data to reduce noise and infer context.

What is context?

Context is often gathered with sensors that repeatedly measure some aspect of the user’s state.

It includes a person’s location, activity, goals, resources, state of mind, and nearby people and things.

For example: Global Positioning System (GPS) sensor can repeatedly measure a person’s location at some interval in time.
Tracking Example

Actual Path and Measured Locations

\[ z_i = x_i + v_i \]
\[ v_i \sim N \left( 0, \begin{bmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{bmatrix} \right) \]
Mean and Median Filters

- This filters are simple to implement and works well to reduce noise.
- The primary disadvantage is that it introduces lag in the estimate.
- One way to reduce the lag is to use a weighted average, whose weights decrease in value for increasingly older measurements.
- Another potential problem with the mean filter is very sensitive.
Mean and Median Filters

- That’s why they are far from the actual path, and can move the mean to any location.
- By using this equation we can compute the mean.

$$\hat{x}_i = \frac{1}{n} \sum_{j=-n+1}^{i} z_j$$

- The median filter is more strong version than the mean filter.
- The median filter produces fewer large movements from the true path, because it is more strong to outliers.
Mean and Median Filters

- That’s why they are close to the actual path.
- By using this equation we can compute the median.

$$\hat{x}_i = \text{median}\{z_{i-n+1}, z_{i-n+2}, \ldots, z_{i-1}, z_i\}$$

- However, they do suffer from lag, and they do not built-in to estimate any higher level variables such as speed.
Mean and Median Filters
The Kalman filter is a big step-up in modern from the mean and median filters.
The mean and median filters can only estimate states that are directly measured.
The Kalman filter makes a distinction between what quantities are in the measurement vector and the unknown state vector.
The mean and median filters have no model of how the state variables change over time.
Kalman Filter

- Kalman filter has a dynamic model of the system to keep up with changes over time.
- The dynamic model can include parameters beyond the ones that are measured.
- The dynamic model counterbalances the measurement model to give a tunable tradeoff.
The Kalman filter has an advantage of knowing the exact noise covariance, from below Equation.

\[ R_i = \begin{bmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{bmatrix} \]

By using matrix \( H_i \) Kalman filter simply deletes the unmeasured velocity.

\[ H_i = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \]

The Kalman filter proceeds in two steps for each new measurement.
The first step is to extrapolate the state and the state error covariance from the previous estimates.

The second step is to update the extrapolations with the new measurement.

By Applying these equations:

\[ \hat{x}_i^{(-)} = \phi_{i-1} \hat{x}_i^{(+)} \]
\[ P_i^{(-)} = \phi_{i-1} P_i^{(+)} \phi_{i-1}^T + Q_{i-1} \]
\[ K_i = P_i^{(-)} H_i^T \left( H_i P_i^{(-)} H_i^T + R_i \right)^{-1} \]
\[ \hat{x}_i^{(+)} = \hat{x}_i^{(-)} + K_i (z_i - H_i \hat{x}_i^{(-)}) \]
\[ P_i^{(+)} = (I - K_i H_i) P_i^{(-)} \]
Kalman Filter
The Kalman filter result, in dark gray, tends to overshoot turns because its dynamic model assumes a single, straight line path.

The lighter gray line is a version of the Kalman filter tuned to be more sensitive to the data.

It follows turns better, but it is also more sensitive to noise.

The Kalman filter is also unsuitable for representing discrete state variables.
The particle filter is a more general version of the Kalman filter, with less restrictive assumptions.

The particle filter does not require a linear model for the process and it does not assume Gaussian noise.

One good example of a particle filter for a ubicomp application is the work of Patterson et al. (2003).

They created a rich model of a person’s location, velocity, transportation mode, GPS error, and the presence of a parking lot or bus stop.
An easy-to-understand introduction to particle filtering is the chapter by Doucet et al. (2001).

Hightower and Borriello (2005) give a useful case study of particle filters for location tracking in ubicomp.

To stay constant measurement, the particle filter will use the following equation.

\[ p(z_i|x_i) = N((x_i, y_i)^T, R_i) \]

\( P(Z_i/X_i) \) could express the fact that measurements in some regions are less accurate than measurements in other regions.
There are several variations of the particle filter, one of the most popular is called the bootstrap filter, introduced by Gordon (1994).

To represent the state vector, the particle filter works with a population of \( N \) particles.

The next step computes “importance weights” for each particle according to the measurement model.

\[
\tilde{w}_i^{(j)} = p(z_i | \tilde{x}_i^{(j)})
\]
Finally, the last step in the loop is the selection step.

This step tends to eliminate unlikely particles, and it is not unusual to pick the same more than once if its weight is relatively large.

This equation works for continuous variables.

\[ \hat{x}_i = \sum_{j=1}^{N} \tilde{w}_{i}^{(j)} \tilde{x}_i^{(j)} \]
The main problem with the particle filter is computation time, by using more N particles.

Fox (2003) gives a method for choosing N based on bounding, because particle filter allows adding more particles to account for the larger state space.

Rao-Blackwellized particle filter (Murphy and Russell, 2001) uses more conventional filter like, Kalman could cover location and velocity, and a particle filter could track the higher level states.
Particle Filter
THANK YOU FOR YOUR ATTENTION