Kalman Filter의 이해

http://nerve.tistory.com

칼만필터는 노이즈가 포함되어 있는 선형 역학 시스템의 상태를 추적하는 필터로 루돌프 칼만이라는 사람에 의해 개발되었다. 칼만필터는 신호처리, 로봇 공학 등의 여러 분야에 사용되며, 매우 효율적인 성능으로 널리 쓰이고 있다.

여기에서는 칼만 필터의 수식 유도 과정에 대해 Gaussian Probability Density Function 관점에서 설명한다.

간단히 예를 들기 위해서 일차원에서 움직이는 물체에 대한 운동방정식을 모델링 해보기로 하자. 시스템의 state vector는 위치와 속도로 표현하면 아래와 같다.

$$X_t = \begin{bmatrix} x_t \\ \dot{x}_t \end{bmatrix}$$

물체에 가해지는 제어 입력에 대한 정의는 아래와 같이 할 수 있다.

$$u_t = \frac{f_t}{m}$$

물체의 모멘트(혹은 질량) m에 미치는 힘 f_t의 관계로 설명할 수 있다. 이것으로부터 시간 t에 해당하는 물체의 위치와 속도에 대한 관계식은 아래와 같이 정의할 수 있다.

$$x_t = x_{t-1} + \left(\dot{x}_{t-1} \times \Delta t\right) + \frac{f_1 (\Delta t)^2}{2m}$$

$$\dot{x}_t = \dot{x}_{t-1} + \frac{f_t \Delta t}{m}$$

위의 식을 시스템의 상태 방정식 형태(state space variable from)로 정리하면 아래와 같다.

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_{t-1} \\ \dot{x}_{t-1} \end{bmatrix} + \begin{bmatrix} \frac{\Delta t^2}{2} \\ \Delta t \end{bmatrix} u_t$$

➔ $X_t = A \cdot X_{t-1} + B \cdot u_t$

여기서 x_t 는 우리가 관심을 가지는 시스템의 state vector이고, u_t 는 제어 입력이다. A 는 시스템의 state transition matrix이며, B 는 control input matrix이다.

그런데 우리가 x_t 에 대한 정확한 정보를 알 수 있다면 좋겠지만 대부분의 경우 정확한 정보를 알 수가 없기 때문에 센서 등의 계측기로부터 x_t 에 대한 추정(estimation)을 해야 한다. x_t 에 대한 estimation value $\hat{x}_t$ 는 x_t 와 마찬가지로 시스템 상태 방정식을 가진다.

$$\hat{X}_t = A \cdot \hat{X}_{t-1} + B \cdot u_t$$

예측 정보인 $\hat{x}_t$ 와 x_t 사이에는 아무래도 오차가 있을 수 밖에 없는데 이 오차의 공분산을 다음과 같이 정의한다.

$$P_t^- = E\left[\left(x_t - \hat{x}_t^-\right)\left(x_t - \hat{x}_t^-\right)^T\right]$$

P_t^- 는 시간 t 에서의 오차 공분산에 대한 기대값 매트릭스이다. $\hat{x}_t^-$ 는 시간 t 에 추정 오차에 대한 보상을 하기 전 값이다. P_t^- 에 대한 식을 정리하기 전에 x_t 에 관해 정의를 다시 하도록 하자.

앞에서 x_t 를 설명하는 X_t 에 대한 식은 아래와 같이 정의했었다.

$$X_t = A \cdot X_{t-1} + B \cdot u_t$$

그런데, x_t 에 대한 위의 모델이 정말 완벽하게 정확할까? 모델링이 완벽하게 되었다면 아무런 문제가 없겠지만, 실제 시스템에 대한 모델링을 100% 완벽하게 하기는 어려운 문제이다. 따라서 위의 state space equation matrix는 어느 정도의 model error를 가지고 있다고 가정하고 이에 대한 term이 추가되어야 한다. 따라서 위의 식을 다시 정의하면,

$$X_t = A \cdot X_{t-1} + B \cdot u_t + w_t$$

로 다시 쓸 수 있다. 여기서 w_t 는 시스템 모델에 대한 error이며, process noise라고 한다. 이제 P_t^- 에 대한 식을 정리해 보기로 하자.

$$\begin{aligned} x_t - \hat{x}_t^- &= \left(A \cdot x_{t-1} + B \cdot u_t + w_t\right) - \left(A \cdot \hat{x}_{t-1} + B \cdot u_t\right) \\ &= A \cdot \left(x_{t-1} - \hat{x}_{t-1}\right) + w_t \end{aligned}$$

이므로,

$$\begin{aligned} P_t^- &= E\left[\left(x_t - \hat{x}_{t-1}^-\right)\left(x_t - \hat{x}_{t-1}^-\right)^T\right] \\ &= E\left[(A \cdot (x_t - \hat{x}_{t-1}) + w_t) \times (A \cdot (x_t - \hat{x}_{t-1}) + w_t)^T\right] \\ &= A \cdot E\left[(x_t - \hat{x}_{t-1}) \times (x_t - \hat{x}_{t-1})^T\right] \cdot A^T + A \cdot E\left[(x_{t-1} - \hat{x}_{t-1}) \cdot w_t^T\right] \\ &+ E\left[w_t \cdot (x_{t-1} - \hat{x}_{t-1})^T\right] \cdot A^T + E\left[w_t \cdot w_t^T\right] \end{aligned}$$

여기서 $x_t - \hat{x}_t^-$ 와 process noise인 w_t 은 uncorrelated 하므로

$$A \cdot E\left[(x_{t-1} - \hat{x}_{t-1}) \cdot w_t^T\right] = E\left[w_t \cdot (x_{t-1} - \hat{x}_{t-1})\right] \cdot A^T = 0$$

이다. 따라서

$$P_t^- = A \cdot E\left[(x_t - \hat{x}_{t-1}) \times (x_t - \hat{x}_{t-1})^T\right] \cdot A^T + E\left[w_t \cdot w_t^T\right]$$

즉, 위의 식을 정리하여 오차 공분산 매트릭스를 구할 수 있다.

$$P_t^- = AP_{t-1}A^T + Q_t$$

시스템에 대한 추정으로부터 오차 매트릭스를 정리해 보았다. 그런데, 실제로 오차는 어떻게 표현될 수 있을까? x_t 의 추정에 대한 불확실성은 표현할 수 있는 방법은 여러 가지가 있을 수 있다. 가장 간단하면서 강력한 방법 중에 하나는 Gaussian function으로 표현하는 것이다. $\hat{x}_t$ 는 x_t 의 추정에 대한 mean 값과 variance 값을 가지는 Gaussian 확률 밀도 함수(Probability Density Function, PDF)로 정의할 수 있다.

추정(estimation)에 대한 문제에 관해 좀 더 생각해 보자.

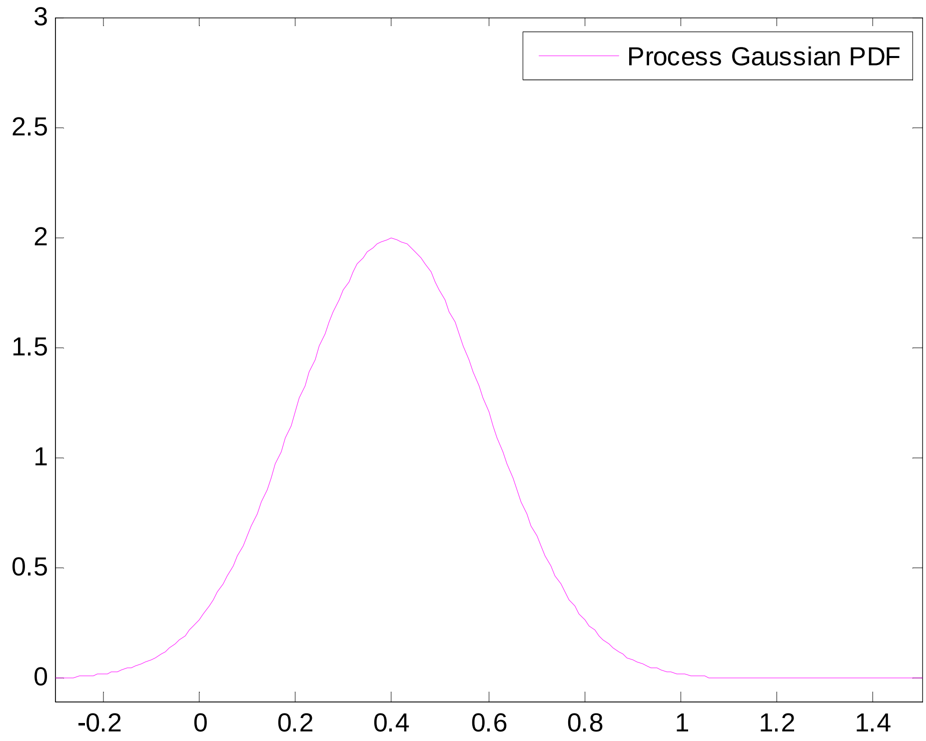


Figure 1 Gaussian PDF

앞에서 x_t 가 존재하는 위치에 대한 추정은 Gaussian PDF를 이용할 수 있다고 했다. 만약 Gaussian PDF로 추정을 한다면, x_t 가 r 에 위치할 확률은 아래와 같다.

$$y(r;\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(r-\mu)^2}{2\sigma^2}}$$

여기서 y 는 mean이 μ 이고 variance가 σ 일 때, x_t 가 r 에 위치할 확률로 생각할 수 있다. 우리는 이 Gaussian PDF를 이용하여 프로세스 오차 및 측정 오차를 표현하며 이를 역으로 이용하여 매 state 마다 x_t 를 추정하는데 이용한다. Gaussian PDF는 state를 추정할 때와 측정할 때 각각 따로 존재한다. 따라서, 우리가 사용할 x_t 에 대한 최종 추정 값은 이 두 Gaussian PDF의 합성(fusion)을 이용해야 한다.

여기서 Gaussian PDF의 중요한 속성을 이용한다. 서로 다른 두 개의 Gaussian PDF를 합성하면 또 다른 형태의 Gaussian PDF의 중요한 속성을 이용한다. 서로 다른 두 개의 Gaussian PDF를 합성하면 또 다른 형태의 Gaussian PDF가 만들어진다는 것이다. 우리는 이러한 특성을 이용해 복잡도를 증가시키지 않고 매 state에 대한 Gaussian PDF를 계산할 수 있다.

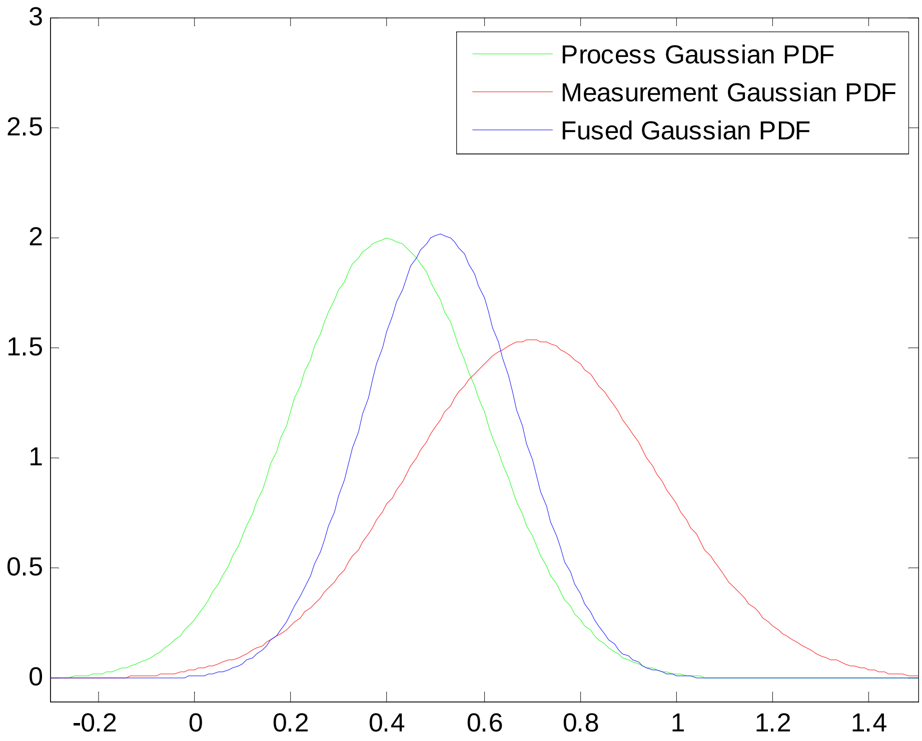


Figure 2 Gaussian PDF의 합성(Fusion)

그림 2의 추정에 대한 Gaussian PDF의 식이 아래와 같다고 하자.

$$y_1(r;\mu_1,\sigma_1) = \frac{1}{\sqrt{2\pi\sigma_1^2}} e^{-\frac{(r-\mu_1)^2}{2\sigma_1^2}}$$

또한 측정 Gaussian PDF가 아래와 같다고 하자.

$$y_2(r;\mu_2,\sigma_2) = \frac{1}{\sqrt{2\pi\sigma_2^2}} e^{-\frac{(r-\mu_2)^2}{2\sigma_2^2}}$$

위 두 개의 Gaussian PDF의 곱을 통해 PDF가 합성됨으로 인해 새로운 Gaussian PDF를 얻을 수 있다.

$$
\begin{aligned}
y_{fused}(r;\mu_1,\sigma_1,\mu_2,\sigma_2) &= \frac{1}{\sqrt{2\pi\sigma_1^2}} e^{-\frac{(r-\mu_1)^2}{2\sigma_1^2}} \times \frac{1}{\sqrt{2\pi\sigma_2^2}} e^{-\frac{(r-\mu_2)^2}{2\sigma_2^2}} \\
&= \frac{1}{2\pi\sqrt{\sigma_1^2\sigma_2^2}} e^{-\left(\frac{(r-\mu_1)^2}{2\sigma_1^2}+\frac{(r-\mu_2)^2}{2\sigma_2^2}\right)}
\end{aligned}
$$

위의 함수를 다시 정리하면,

$$
y_{fused}(r;\mu_{fused},\sigma_{fused}) = \frac{1}{\sqrt{2\pi\sigma_{fused}^2}} e^{-\frac{(r-\mu_{fused})^2}{2\sigma_{fused}^2}}
$$

여기서 μ_{fused} 와 σ_{fused}^2 는 다음과 같다.

$$
\begin{aligned}
\mu_{fused} &= \frac{\mu_1\sigma_2^2 + \mu_2\sigma_1^2}{\sigma_1^2 + \sigma_2^2} \\
&= \mu_1 + \frac{\sigma_1^2\left(\mu_2 - \mu_1\right)}{\sigma_1^2 + \sigma_2^2}
\end{aligned}
$$

$$
\begin{aligned}
\sigma_{fused} &= \frac{\sigma_1^2\sigma_2^2}{\sigma_1^2 + \sigma_2^2} \\
&= \sigma_1^2 - \frac{\sigma_1^4}{\sigma_1^2 + \sigma_2^2}
\end{aligned}
$$

위의 두 식은 칼만 필터 알고리즘에서 업데이트에 관한 수식이다. 프로세스 노이즈 및 측정 노이즈가 같은 도메인에 있다고 가정하고 전개하였다. 그러나 실제 x_t 가 위치에 대한 정보라면, 프로

세스 노이즈는 위치 도메인이고, 측정 노이즈는 시간 도메인이다. 한쪽의 Gaussian PDF를 다른 한쪽의 Gaussian PDF에 곱하기 위해서는 도메인을 맞추어 주어야 한다. 이를 위해서 프로세스 노이즈 쪽을 c 로 나누어 스케일링을 통해 측정 노이즈 도메인으로 맞추도록 한다.

$$y_1(s;\mu_1,\sigma_1,c) = \frac{1}{\sqrt{2\pi\left(\frac{\sigma_1}{c}\right)^2}} e^{-\frac{(s-\frac{\mu_1}{c})^2}{2\left(\frac{\sigma_1}{c}\right)^2}}$$

$$y_2(s;\mu_2,\sigma_2) = \frac{1}{\sqrt{2\pi\sigma_2^2}} e^{-\frac{(s-\mu_2)^2}{2\sigma_2^2}}$$

위의 수식을 통해 다시

$$\frac{\mu_{fused}}{c} = \frac{\mu_1}{c} + \frac{\left(\frac{\sigma_1}{c}\right)^2\left(\mu_2 - \frac{\mu_1}{c}\right)}{\left(\frac{\sigma_1}{c}\right)^2 + \sigma_2^2}$$

$$\Rightarrow \mu_{fused} = \mu_1 + \left(\frac{\left(\frac{\sigma_1}{c}\right)^2}{\left(\frac{\sigma_1}{c}\right)^2 + \sigma_2^2}\right)\cdot\left(\mu_2 - \frac{\mu_1}{c}\right)$$

위의 식에서 $H = 1/c$ 그리고 $K = (H\sigma_1^2)/(H^2\sigma_1^2 + \sigma_2^2)$ 으로 치환하면 아래와 같은 식으로 다시 쓸 수 있다.

$$\mu_{fused} = \mu_1 + K\cdot(\mu_2 - H\mu_1)$$

마찬가지로 합성된 variance estimation에 대해서 정리하면,

$$\frac{\sigma_{fused}^2}{c} = \left(\frac{\sigma_1}{c}\right)^2 - \frac{\left(\frac{\sigma_1}{c}\right)^4}{\left(\frac{\sigma_1}{c}\right)^2 + \sigma_2^2}$$

$$\Rightarrow \ \sigma_{fused}^2 = \sigma_1^2 - \left(\frac{\frac{\sigma_1^2}{c}}{\left(\frac{\sigma_1}{c}\right)^2 + \sigma_2^2}\right) \cdot \frac{\sigma_1^2}{c}$$

$$\sigma_{fused}^2 = \sigma_1^2 - KH\sigma_1^2$$

자, 이제까지 칼만 필터에 필요한 모든 수식의 유도는 끝났다. 이를 기반으로 칼만 필터 알고리즘을 정리하면 아래와 같다.

$$\mu_{fused} \Rightarrow \hat{x}_t$$

$$\mu_1 \Rightarrow \hat{x}_{t-1}$$

$$\mu_2 \Rightarrow \hat{x}_t$$

$$\sigma_{fused}^2 \Rightarrow P_t$$

$$\sigma_1^2 \Rightarrow P_t^-$$

$$\sigma_1^2 \Rightarrow R_t$$

$$H \Rightarrow H_t$$

$$K = \frac{H\sigma_1^2}{H^2\sigma_1^2 + \sigma_2^2} \Rightarrow = P_t^- H^T (HP_t^- H^T + R_t)^{-1}$$

$$\mu_{fused} = \mu_1 + K \cdot \left(\mu_2 - H\mu_1\right) \Rightarrow \hat{x}_t = \hat{x}_t^- + K_t\left(z_t - H\hat{x}_t^-\right)$$

$$\sigma_{fused}^2 = \sigma_1^2 - KH\sigma_1^2 \Rightarrow P_t = P_t^- = K_t H_t P_t^-$$

칼만 필터 알고리즘에서 업데이트는 하기와 같이 이루어진다. 실제 칼만 필터를 구현 시에는

discrete domain에서 구현하기 때문에 수식 상에서 t ➔ k 로 대체한다.

Initial estimates for x_{k-1} and P_{k-1}

⇩

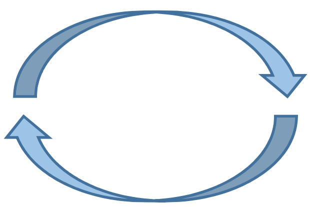
Time Update("Predict")

(1) Project the state ahead

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_k$$

(2) Project the error covariance ahead

$$P_k^- = AP_{k-1}A^T + Q$$



Measurement Update("Correct")
(1) Compute the Kalman gain $K_k = P_k^- H^T (H P_k^- H^T + R)^{-1}$ (2) Update estimate with measurement z_k $\hat{x}_k = \hat{x}_k^- + K_k (z_k - H \hat{x}_k^-)$ (3) Update the error covariance $P_k = (1 - K_k H_k) P_k^-$

Reference

"Understanding the Basis of the Kalman Filter Via a Simple and Intuitive Derivation", Ramsey Faragher, IEEE Signal Processing Magazine, SEPTEMBER 2012