

# Recall: The two update steps in robot localization

1. ACTION (or prediction) update: the robot moves and estimates its position through its proprioceptive sensors. During this step, the robot uncertainty grows. This step uses the **Total Probability formula** to update belief

$$\overline{bel}(x_t) = p(x_t | u_t, x_{t-1}) * bel(x_{t-1})$$

$$p(x) = \int_y p(x|y)p(y)dy$$

2. PERCEPTION (or measurement) update: the robot makes an observation using its exteroceptive sensors and correct its position by opportunely combining its belief before the observation with the probability of making exactly that observation. During this step, the robot uncertainty shrinks. This step uses the **Bayes Rule** to update belief.

$$bel(x_t) = \eta \cdot p(z_t | x_t) \cdot \overline{bel}(x_t)$$

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)}$$



# Recall: Markov versus Kalman localization

---

- Two approaches exist to represent the probability distribution and to compute the Total Probability and Bayes Rule during the Action and Perception phases
- Markov approach:
  - The configuration space is divided into many cells. Each cell contains the probability of the robot to be in that cell.
  - The probability distribution of the sensors model is also discrete.
  - During Action and Perception, all the cells are updated. Therefore, the computational cost is very high.
- Kalman approach:
  - The probability distribution of both the robot configuration and the sensor model is assumed to be continuous and Gaussian!
  - Need only to update mean value  $\mu$  and covariance  $\Sigma$ . Therefore the computational cost is very low!

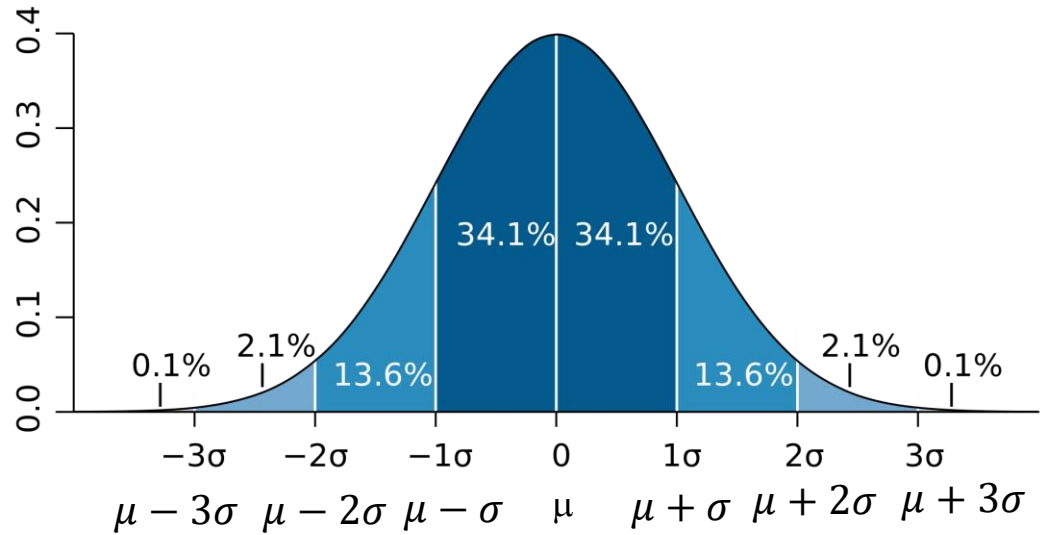


# Gaussian (Normal) Distribution: Univariate

## Univariate

$X \sim N(\mu, \sigma^2)$ :

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1(x-\mu)^2}{2\sigma^2}}$$



- $\mu$ : Mean
- $\sigma^2$ : Variance
- $\sigma$ : Standard Deviation

$$\mu = E(X)$$

$$\sigma^2 = E((X - \mu)^2)$$

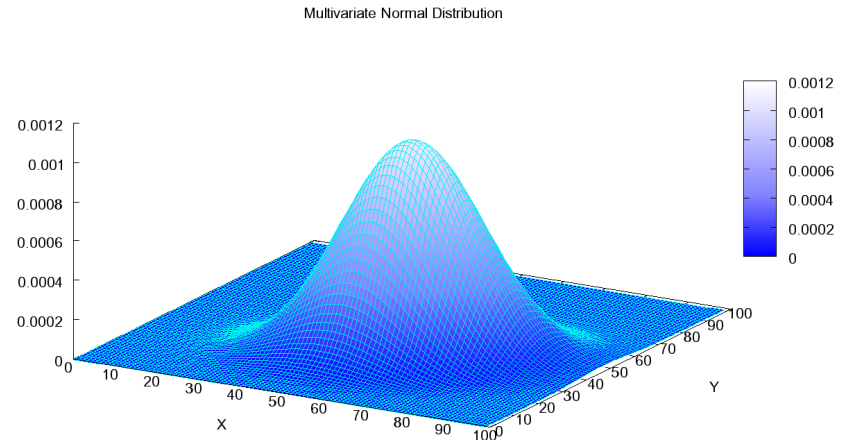


# Gaussian (Normal) Distribution: Multivariate

## Multivariate

$$\mathbf{X} = (X_1, \dots, X_k)^T \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma}):$$

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{1/2} |\boldsymbol{\Sigma}|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})}$$



- $\boldsymbol{\mu}$ : Mean vector

$$\boldsymbol{\mu} = (\mu_1, \dots, \mu_k)^T = (E(X_1), \dots, E(X_k))^T$$

- $\Sigma_{i,j}$ : Covariance ( $i \neq j$ )

$$\Sigma_{i,j} = E((X_i - \mu_i)(X_j - \mu_j))$$

- $\Sigma_{i,i} = \sigma_i^2$ : Variance

$$\Sigma_{i,i} = \sigma_i^2 = E((X_i - \mu_i)^2)$$

- $\boldsymbol{\Sigma}$ : Covariance matrix

$$\boldsymbol{\Sigma} = (\Sigma_{i,j})_{k \times k}$$



# Properties of Gaussians

$$\left. \begin{array}{l} X \sim N(\mu, \sigma^2) \\ Y = aX + b \end{array} \right\} \Rightarrow Y \sim N(a\mu + b, a^2\sigma^2)$$

← Univariate  
↓

$$\left. \begin{array}{l} X_1 \sim N(\mu_1, \sigma_1^2) \\ X_2 \sim N(\mu_2, \sigma_2^2) \end{array} \right\} \Rightarrow P(X_1)P(X_2) \sim N\left(\frac{\sigma_2^2}{\sigma_1^2 + \sigma_2^2}\mu_1 + \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2}\mu_2, \frac{1}{\sigma_1^{-2} + \sigma_2^{-2}}\right)$$

$$\left. \begin{array}{l} X \sim N(\mu, \Sigma) \\ Y = AX + B \end{array} \right\} \Rightarrow Y \sim N(A\mu + B, A\Sigma A^T)$$

← Multivariate Gaussians  
↓

$$\left. \begin{array}{l} X_1 \sim N(\mu_1, \Sigma_1) \\ X_2 \sim N(\mu_2, \Sigma_2) \end{array} \right\} \Rightarrow P(X_1)P(X_2) \sim N\left(\frac{\Sigma_2}{\Sigma_1 + \Sigma_2}\mu_1 + \frac{\Sigma_1}{\Sigma_1 + \Sigma_2}\mu_2, \frac{1}{\Sigma_1^{-1} + \Sigma_2^{-1}}\right)$$

- We stay in the “Gaussian world” as long as we start with Gaussians and perform only linear transformations.

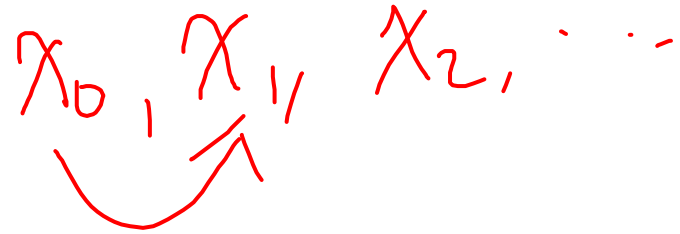


# Discrete Kalman Filter

- The state transition probability  $p(x_t | x_{t-1}, u_t)$  must be a linear function in its arguments with added Gaussian noise.

$$x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t$$

- $x_t$  and  $x_{t-1}$  are state vectors, and
- $u_t$  is the control vector at time

$$x_0, x_1, x_2, \dots$$


- The measurement probability  $p(z_t | x_t)$  must be a linear function in its arguments with added Gaussian noise

$$z_t = C_t x_t + \delta_t$$



# Components of a Kalman Filter

$$x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t$$

$$z_t = C_t x_t + \delta_t$$

$A_t$

Matrix ( $n \times n$ ) that describes how the state evolves from  $t-1$  to  $t$  without controls or noise.

$B_t$

Matrix ( $n \times m$ ) that describes how the control  $u_t$  changes the state from  $t-1$  to  $t$ .

$C_t$

Matrix ( $k \times n$ ) that describes how to map the state  $x_t$  to an observation  $z_t$ .

$\varepsilon_t$

Random variables representing the process and measurement noise that are assumed to be independent and normally distributed with covariance  $R_t$  and  $Q_t$  respectively.

$\delta_t$



# Linear Gaussian Systems: Initialization

---

- Initial belief is normally distributed:

$$bel(x_0) = N(x_0; \mu_0, \Sigma_0)$$

$\mu$

State mean matrix ( $n \times 1$ ).

$\Sigma$

State covariance matrix ( $n \times n$ )





# Linear Gaussian Systems: Dynamics

- Dynamics are linear function of state and control plus additive noise:

$$x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t$$

Total Probability formula

$$\overline{bel}(x_t) = p(x_t | u_t, x_{t-1}) * bel(x_{t-1})$$

ACTION (or prediction) update

$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = A_t \mu_{t-1} + B_t u_t \\ \bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t \end{cases}$$



# Linear Gaussian Systems: Observations

- Observations are linear function of state plus additive noise:

$$z_t = C_t x_t + \delta_t$$

$$C_t = (1, 0) \begin{pmatrix} x_t \\ \dot{x}_t \end{pmatrix} = x_t$$

Bayes Rule

PERCEPTION (or correction) update

$$bel(x_t) = \eta \cdot p(z_t | x_t) \cdot \overline{bel}(x_t)$$

$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - C_t \bar{\mu}_t) \\ \Sigma_t = (I - K_t C_t) \bar{\Sigma}_t \end{cases}$$

Where  $K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$  called Kalman Gain



# Kalman Filter Algorithm

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

$$x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t$$

$$z_t = C_t x_t + \delta_t$$

2. Prediction:

3. 
$$\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t$$

4. 
$$\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t$$

If  $R_t \rightarrow 0$ , Then the correction updates are mostly ignored.

5. Correction:

6. 
$$K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$$

7. 
$$\mu_t = \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t)$$

8. 
$$\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t$$

If  $Q_t \rightarrow 0$ , Then  $K_t \rightarrow 1$ .  
Adjust primarily with the corrections

9. Return  $\mu_t$ ,  $\Sigma_t$

If  $Q_t \rightarrow large$ , Then  $K_t \rightarrow 0$ .  
Adjust primarily with the prediction



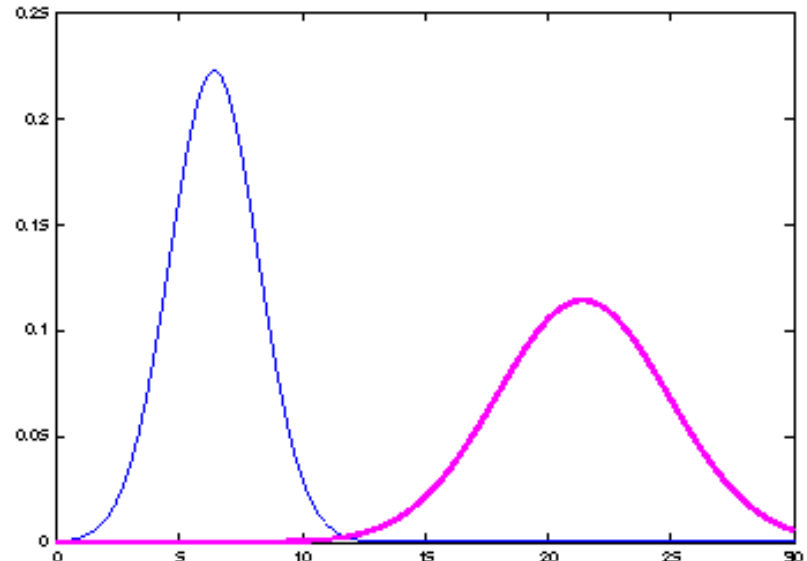
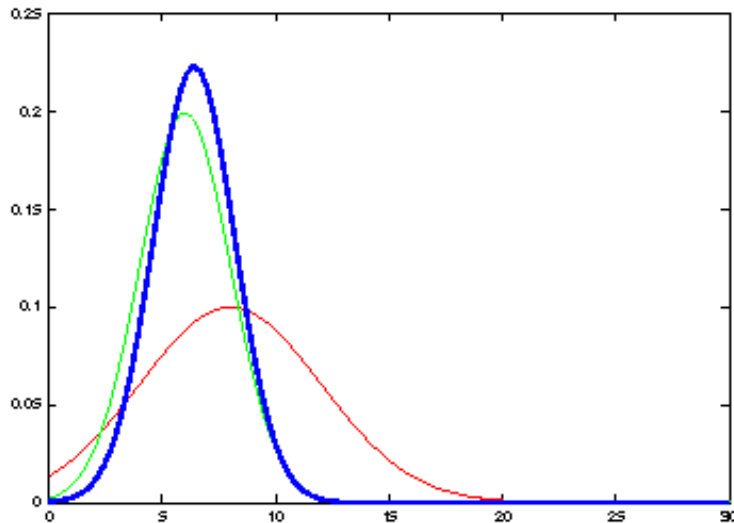
# Kalman Filter **Prediction** Update in 1D

$$x_t = a_t x_{t-1} + b_t u_t + \varepsilon_t$$

$$z_t = c_t x_t + \delta_t$$

$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = a_t \mu_{t-1} + b_t u_t \\ \bar{\sigma}_t^2 = a_t^2 \sigma_{t-1}^2 + \sigma_{act,t}^2 \end{cases}$$

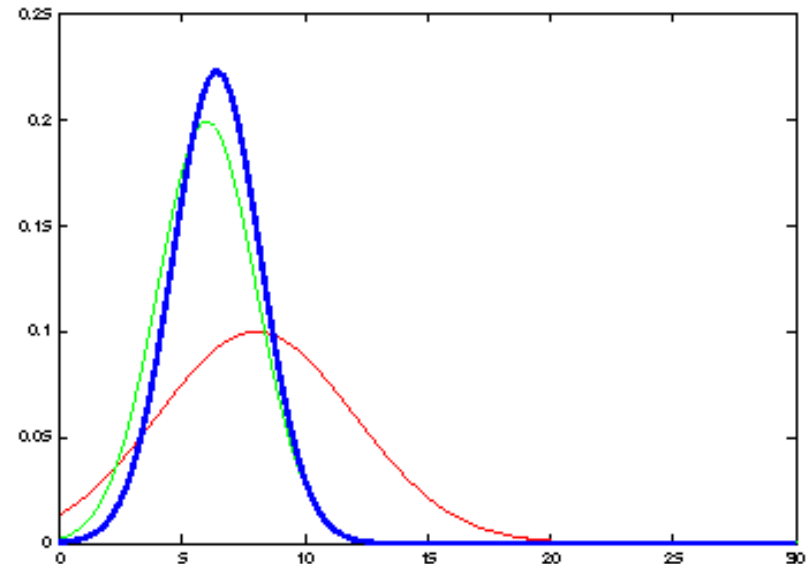
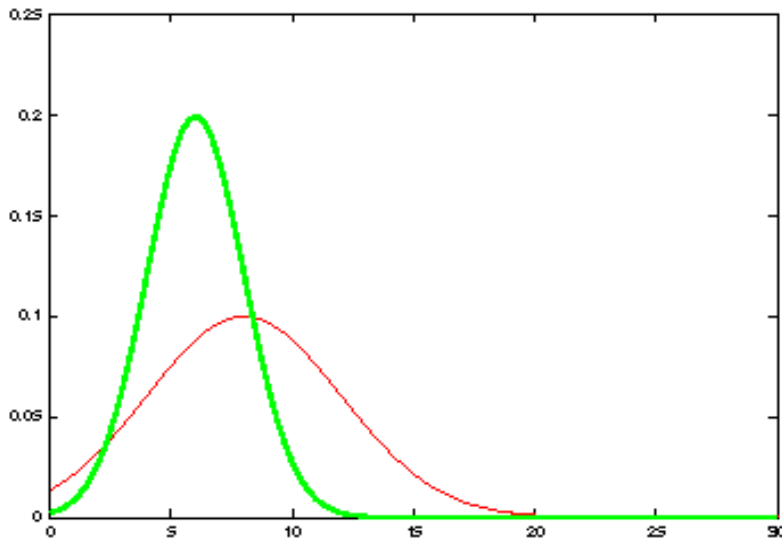
$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = A_t \mu_{t-1} + B_t u_t \\ \bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t \end{cases}$$



# Kalman Filter **Correction** Update in 1D

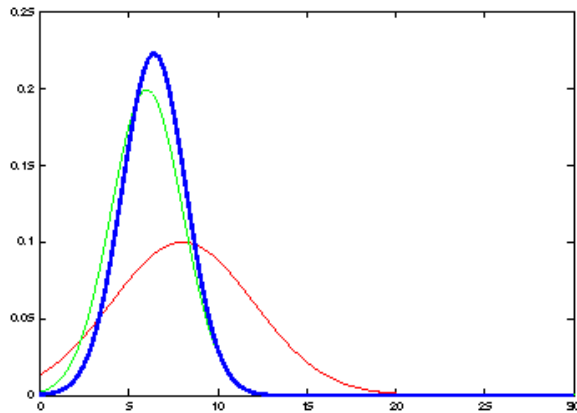
$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - \bar{\mu}_t) \\ \sigma_t^2 = (1 - K_t)\bar{\sigma}_t^2 \end{cases} \quad \text{with} \quad K_t = \frac{\bar{\sigma}_t^2}{\bar{\sigma}_t^2 + \bar{\sigma}_{obs,t}^2}$$

$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - C_t\bar{\mu}_t) \\ \Sigma_t = (I - K_tC_t)\bar{\Sigma}_t \end{cases} \quad \text{with} \quad K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$$



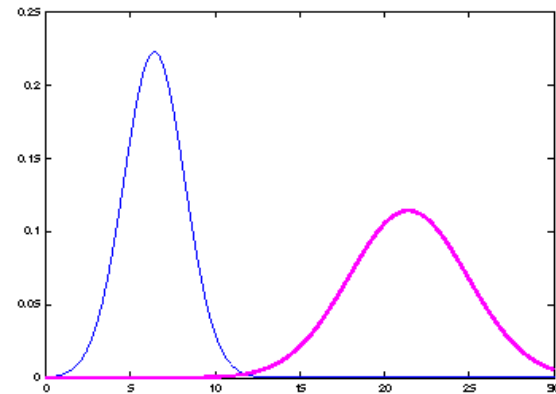
# The Prediction-Correction-Cycle

Prediction

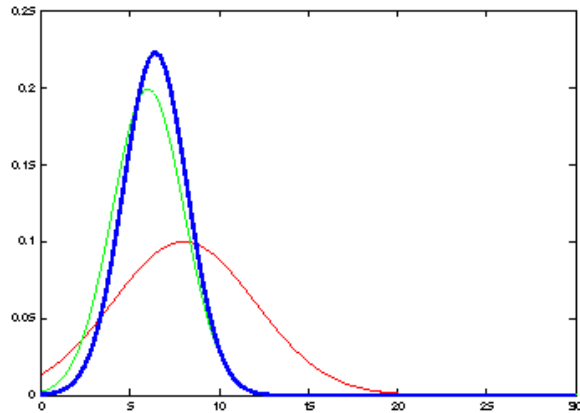


$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = a_t \mu_{t-1} + b_t u_t \\ \bar{\sigma}_t^2 = a_t^2 \sigma_{t-1}^2 + \sigma_{act,t}^2 \end{cases}$$

$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = A_t \mu_{t-1} + B_t u_t \\ \bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t \end{cases}$$

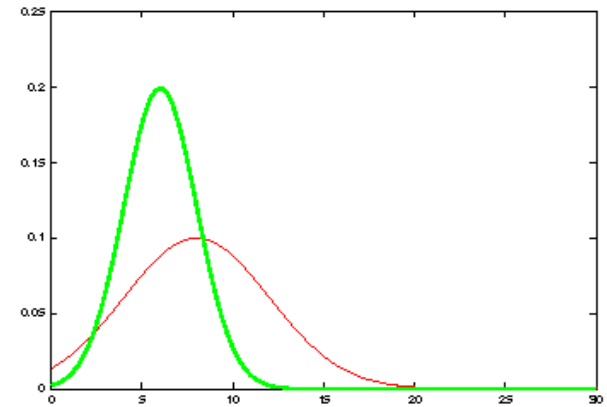


# The Prediction-Correction-Cycle



$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - \bar{\mu}_t) \\ \sigma_t^2 = (1 - K_t)\bar{\sigma}_t^2 \end{cases}, K_t = \frac{\bar{\sigma}_t^2}{\bar{\sigma}_t^2 + \bar{\sigma}_{obs,t}^2}$$

$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - C_t\bar{\mu}_t) \\ \Sigma_t = (I - K_tC_t)\bar{\Sigma}_t \end{cases}, K_t = \bar{\Sigma}_t C_t^T (C_t\bar{\Sigma}_t C_t^T + Q_t)^{-1}$$



Correction



# The Prediction-Correction-Cycle

Prediction

$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - \bar{\mu}_t) \\ \sigma_t^2 = (1 - K_t)\bar{\sigma}_t^2 \end{cases}, K_t = \frac{\bar{\sigma}_t^2}{\bar{\sigma}_t^2 + \bar{\sigma}_{obs,t}^2}$$

$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = a_t\mu_{t-1} + b_tu_t \\ \bar{\sigma}_t^2 = a_t^2\sigma_{t-1}^2 + \sigma_{act,t}^2 \end{cases}$$

$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - C_t\bar{\mu}_t) \\ \Sigma_t = (I - K_tC_t)\bar{\Sigma}_t \end{cases}, K_t = \bar{\Sigma}_t C_t^T (C_t\bar{\Sigma}_t C_t^T + Q_t)^{-1}$$

$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = A_t\mu_{t-1} + B_tu_t \\ \bar{\Sigma}_t = A_t\Sigma_{t-1}A_t^T + R_t \end{cases}$$

Correction





# Kalman Filter Updates in 1D: Example

$$x_t = x_{t-1} + u_t + \varepsilon_t$$

$$z_t = x_t + \delta_t$$

## Prediction

$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = \mu_{t-1} + u_t \\ \bar{\sigma}_t^2 = \sigma_{t-1}^2 + \sigma_{act,t}^2 \end{cases}$$



## Correction

$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - \bar{\mu}_t) \\ \sigma_t^2 = (1 - K_t)\bar{\sigma}_t^2 \end{cases} \quad \text{with} \quad K_t = \frac{\bar{\sigma}_t^2}{\bar{\sigma}_t^2 + \bar{\sigma}_{obs,t}^2}$$

$$\mu_t = \frac{\bar{\mu}_t \bar{\sigma}_{obj,t}^2 + z_t \bar{\sigma}_t^2}{\bar{\sigma}_t^2 + \bar{\sigma}_{obj,t}^2}$$

$$\sigma_t^2 = 1 / \left( 1 / \bar{\sigma}_t^2 + 1 / \bar{\sigma}_{obs,t}^2 \right)$$



# Kalman Filter Update in 1D: Python Code

$$x_t = x_{t-1} + u_t + \varepsilon_t$$

$$z_t = x_t + \delta_t$$

## Prediction

$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = \mu_{t-1} + u_t \\ \bar{\sigma}_t^2 = \sigma_{t-1}^2 + \sigma_{act,t}^2 \end{cases}$$

```
def predict(mean1, var1, mean2, var2):  
    new_mean = mean1 + mean2  
    new_var = var1 + var2  
    return [new_mean, new_var]
```

## Correction

$$bel(x_t) = \mu_t = \frac{\bar{\mu}_t \bar{\sigma}_{obj,t}^2 + z_t \bar{\sigma}_t^2}{\bar{\sigma}_t^2 + \bar{\sigma}_{obj,t}^2}$$

$$\sigma_t^2 = 1 / \left( 1 / \bar{\sigma}_t^2 + 1 / \bar{\sigma}_{obs,t}^2 \right)$$

```
def correct(mean1, var1, mean2, var2):  
    new_mean = (var2 * mean1 + var1 * mean2) / (var1 + var2)  
    new_var = 1 / (1/var1 + 1/var2)  
    return [new_mean, new_var]
```



# Kalman Filter Update in 1D: Prediction

```
# Write a program that will predict your new mean and variance
# given the mean and variance of your prior belief
mu = 10.0
sig = 4.0

# and the mean and variance of your motion.
motion = 12.0
motion_sig = 4.0

def predict(mean1, var1, mean2, var2):
    new_mean = mean1 + mean2
    new_var = var1 + var2
    return [new_mean, new_var]

print (predict(mu, sig, motion, motion_sig))

[22.0, 8.0]
```



# Kalman Filter Update in 1D: Correction

```
# Write a program to correct your mean and variance
# when given the mean and variance of your current belief
mu = 10.0
sig = 8.0

# and the mean and variance of your measurement.
measurement = 13.0
measurement_sig = 2.0

def correct(mean1, var1, mean2, var2):
    new_mean = (mean1*var2 + mean2*var1)/(var1 + var2)
    new_var = 1/(1/var1 + 1/var2)
    return [new_mean, new_var]

print (correct(mu, sig, measurement, measurement_sig))

[12.4, 1.6]
```



# Programming Assignment 3 (PA2A)

- Write a program that will iteratively prediction and correction based on the location measurements and inferred motions shown below.

```
def predict(mean1, var1, mean2, var2):
    new_mean = mean1 + mean2
    new_var = var1 + var2
    return [new_mean, new_var]

def correct(mean1, var1, mean2, var2):
    new_mean = (var2 * mean1 + var1 * mean2) / (var1 + var2)
    new_var = 1/(1/var1 + 1/var2)
    return [new_mean, new_var]

measurements = [5., 6., 7., 9., 10.]
motion = [0., 1., 1., 2., 1.]
measurement_sig = 4.
motion_sig = 2.
mu = 4
sig = 10000

#Please print out ONLY the final values of the mean
#and the variance in a list [mu, sig].

# Insert code here
```

