深蓝学院
shenlanxueyuan.com

Autonomous Valet Parking（AVP） Theory and Practice
自主代客泊车理论与实践

Lecture 2：State Estimation (EKF)

Lecturer Tong QIN

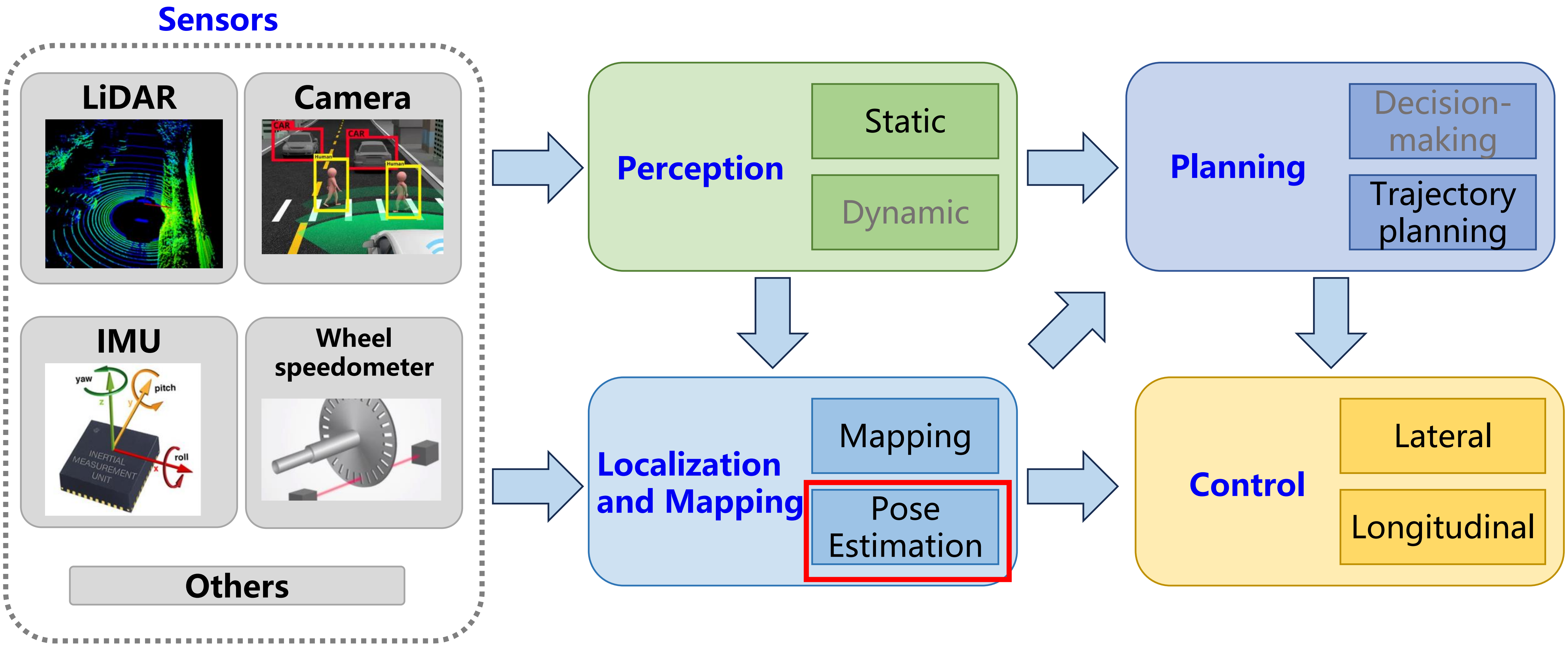
Associate Professor
Shanghai Jiao Tong University
Global Institute of Future Technology
qintong@sjtu.edu.cn

Courtesy: Shaojie SHEN's ELEC 5660, HKUST

Content

State Estimation

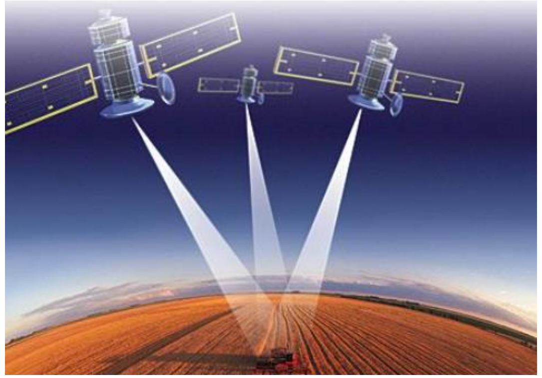
AVP Architecture



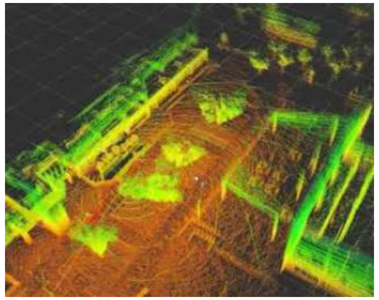
State Estimation

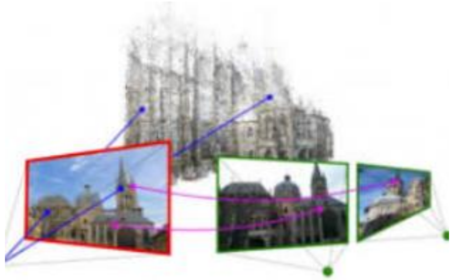
- What sensors can we used for localization?

  - GPS (GNSS: BDS/GPS/Glonass/Galileo)
    - meter level absolution localization / Centi-meter Level (RTK)

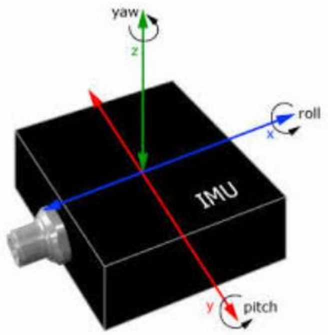
  

  - Lidar / Camera
    - Accurate localization within known maps
    - Relative odometry without maps

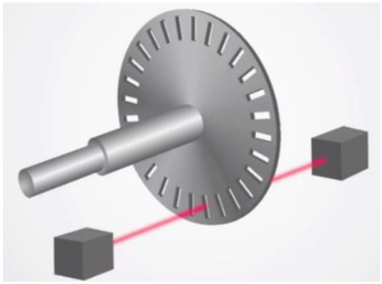
  

  - IMU
    - Acceleration and Angular velocity

  - Wheel odometry
    - Speed

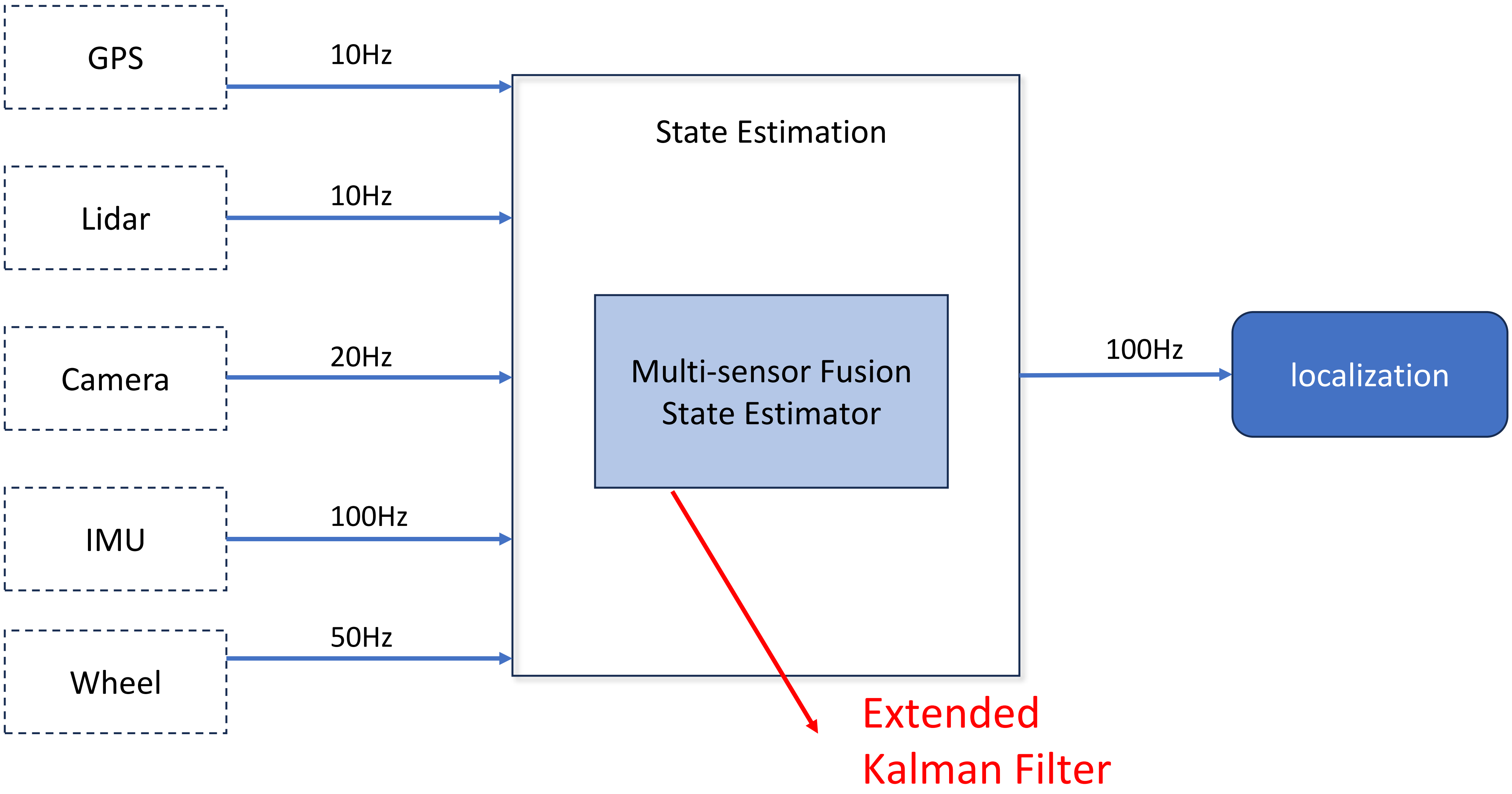
  

State Estimation

- Localization is important but difficult for AVP
  - GPS denied
  - Repeated texture
  - Limited illumination (dark)

- How to get accurate localization results?
  - Fuse multiple sensors

State Estimation

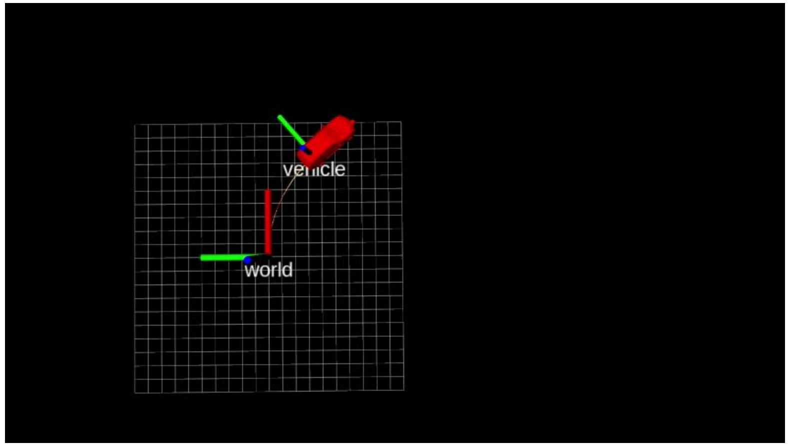


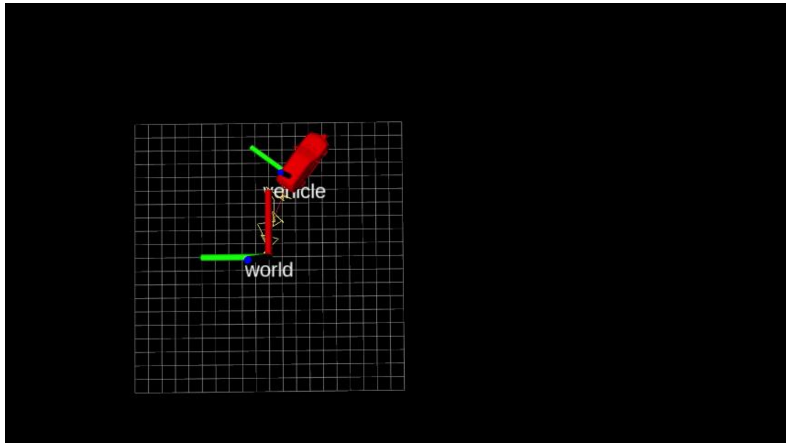
Example: EKF for Vehicle State Estimation

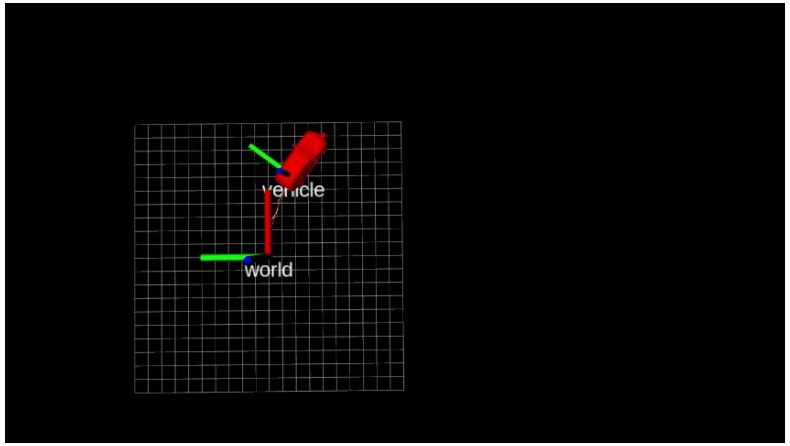
- Dead Reckoning :
  (IMU + Wheel)
  - Smooth
  - Drift

- Measurement Only:
  (GPS)
  - Noisy
  - No drift

- EKF Fusion:
  (IMU + Wheel + GPS)
  - Smooth
  - No drift







Content

Extended Kalman Filter Derivation

→ Bayes Filter → Kalman Filter → Extended Kalman Filter

Bayes Filter

- Markov Property

Kalman Filter

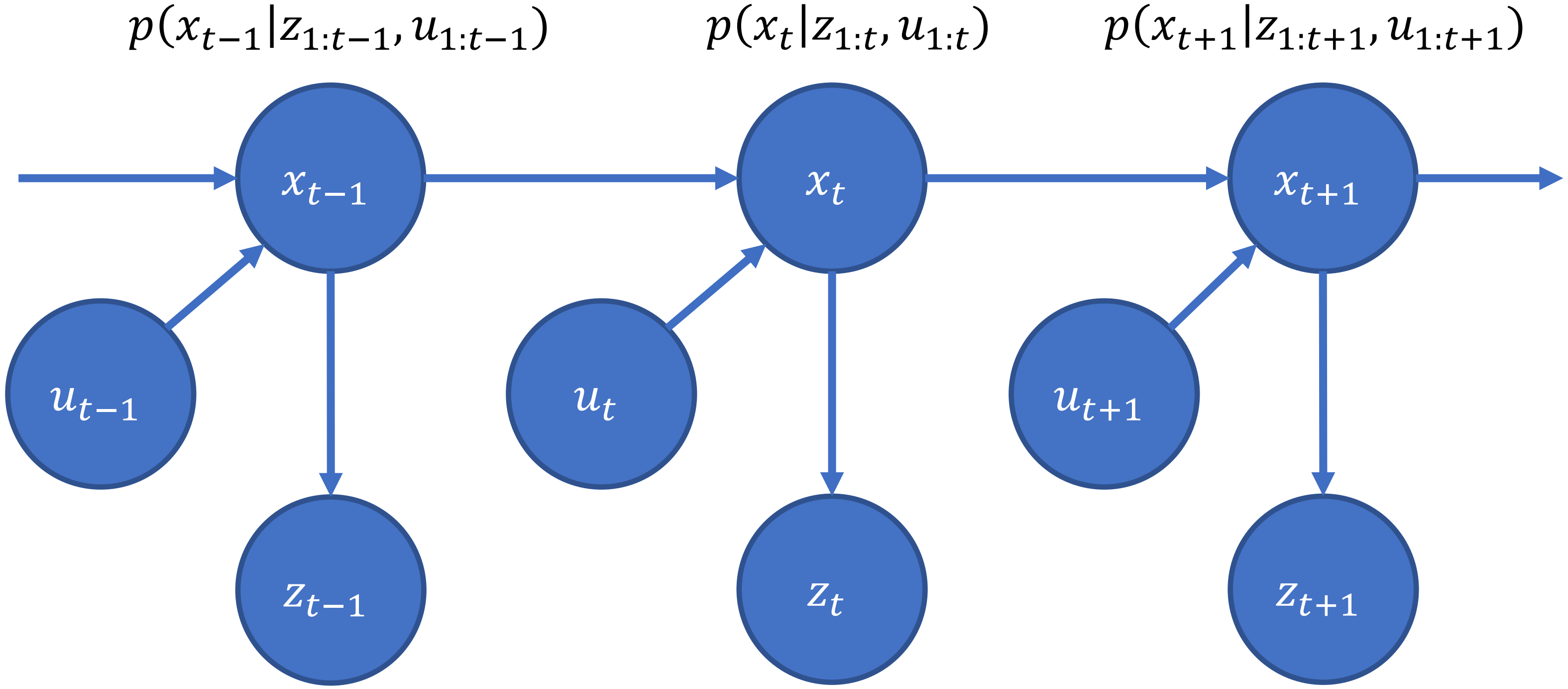
- Prior state is Gaussian distribution
- Motion model is linear with Gaussian noise
- Measurement model is linear with Gaussian noise

Extended Kalman Filter

- Linearize the motion model
- Linearize the measurement model

Bayes' Filter

Bayesian Filter



x: state
u: control signal
z: measurement

Bayes' Theorem

Likelihood Prior

- $$p(x \mid z) = \frac{p(z \mid x)\, p(x)}{p(z)} = \frac{p(z \mid x)\, p(x)}{\int p(z \mid x')\, p(x') dx'}$$

Posterior Evidence

- **Derivation:**
  - Conditional Probability
  - $p(x, z) = p(z \mid x)\, p(x) = p(x \mid z)\, p(z)$

Markov Property

- **Definition:** The future state of the system is conditionally independent of the past states given the current state
  - $p(x_{t+1} | x_{1:t}) = p(x_{t+1} | x_t)$

Bayes' Filter Derivation

Bayes' Theorem

$$p(x \mid z) = \frac{p(z \mid x)\, p(x)}{p(z)}$$

$z_t, z_{1:t-1}, u_{1:t}$

- **Goal:** Want to update the probability distribution of the robot pose using the realizations of the control input and measurement
- $p(x_t \mid z_{1:t}, u_{1:t}) = \dfrac{p(z_t \mid x_t, z_{1:t-1}, u_{1:t})\, p(x_t \mid z_{1:t-1}, u_{1:t})}{p(z_t \mid z_{1:t-1}, u_{1:t})}$ (1)
- **Note:** The measurement is *conditionally independent* of the past measurements and control inputs given the current state of the robot
- $p(z_t \mid x_t, z_{1:t-1}, u_{1:t}) = p(z_t \mid x_t)$ (2)
- **Note:** The denominator can be found as a *marginal distribution* of the numerator
- $p(z_t \mid z_{1:t-1},\ u_{1:t}) = \int p(x_t, z_t \mid z_{1:t-1},\ u_{1:t})\, dx_t$

$$= \int p(z_t \mid x_t) p(x_t \mid z_{1:t-1},\ u_{1:t}) dx_t \quad (3)$$

Marginal Distribution

$$p(z) = \int p(x, z)\, dx$$

Conditional Probability

$$p(x, z) = p(z \mid x)\, p(x)$$

Motion Model

$$p(x_t \mid z_{1:t}, u_{1:t}) = \frac{p(z_t \mid x_t,\ z_{1:t-1}\ , u_{1:t})\, p(x_t \mid z_{1:t-1}, u_{1:t})}{p(z_t \mid z_{1:t-1},\ u_{1:t}\ )}$$

Marginal Distribution

$$p(z) = \int p(x, z)\, dx$$

Conditional Probability

$$p(x, z) = p(z \mid x)\, p(x)$$

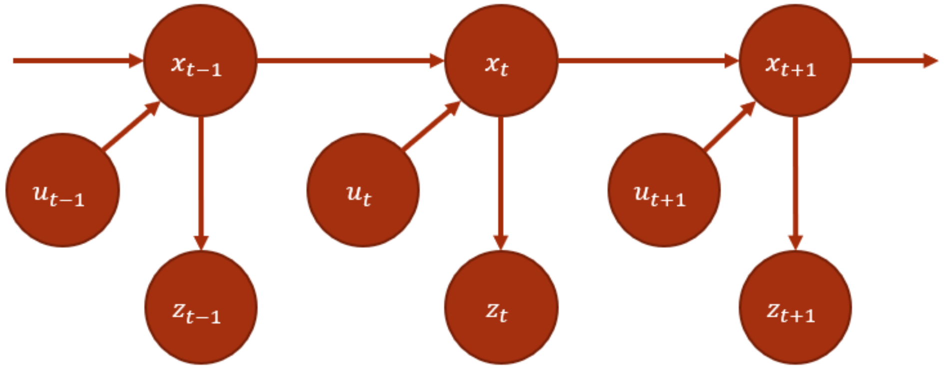
- $p(x_t \mid z_{1:t-1}, u_{1:t})$
- **Note:** Can find the current pose via marginalization
- $p(x_t \mid z_{1:t-1}, u_{1:t}) = \int p(x_t, x_{t-1} \mid z_{1:t-1}, u_{1:t})\, dx_{t-1}$
- $\quad = \int p(x_t \mid x_{t-1}, z_{1:t-1}, u_{1:t})\, p(x_{t-1} \mid z_{1:t-1}, u_{1:t})\, dx_{t-1}$
- **Note:** The future state is *conditionally independent* of the past measurements and control inputs given the current state and input
- $p(x_t \mid z_{1:t-1}, u_{1:t}) = \int p(x_t \mid x_{t-1}, u_t)\, p(x_{t-1} \mid z_{1:t-1}, u_{1:t-1})\, dx_{t-1}$ (4)

Prediction Motion model Prior

Bayesian Filter

$$p(x_t|z_{1:t},u_{1:t})$$

x: state
u: control signal
z: measurement



(1) (2)(3)

$$p(x_t \mid z_{1:t},u_{1:t}) = \frac{p(z_t \mid x_t,\ z_{1:t-1}\ , u_{1:t})\, p(x_t \mid z_{1:t-1}, u_{1:t})}{p(z_t \mid z_{1:t-1},\ u_{1:t}\ )} = \frac{p(z_t \mid x_t)\, p(x_t \mid z_{1:t-1}, u_{1:t})}{\int p(z_t \mid x_t) p(x_t \mid z_{1:t-1}, u_{1:t}) dx_t}$$

(4)

$$\underbrace{p(x_t \mid z_{1:t-1}, u_{1:t})}_{\text{Prediction}} = \int \underbrace{p(x_t \mid x_{t-1}, u_t)}_{\text{Motion model}}\, \underbrace{p(x_{t-1} \mid z_{1:t-1}, u_{1:t-1})}_{\text{Prior}}\, dx_{t-1}$$

Bayes' Filter

$$p(x_t \mid z_{1:t}, u_{1:t}) = \frac{p(z_t \mid x_t)\, p(x_t \mid z_{1:t-1}, u_{1:t})}{\int p(z_t \mid x_t) p(x_t \mid z_{1:t-1}, u_{1:t}) dx_t}$$

$$p(x_t \mid z_{1:t-1}, u_{1:t}) = \int p(x_t \mid x_{t-1}, u_t)\, p(x_{t-1} \mid z_{1:t-1}, u_{1:t-1})\, dx_{t-1}$$

- **Prior:** $p(x_0)$
- **Motion model:** $f(x_t \mid x_{t-1}, u_t)$
- **Measurement model:** $g(z_t \mid x_t)$

- **Prediction step:**
- $p(x_t \mid z_{1:t-1}, u_{1:t}) = \int f(x_t \mid x_{t-1}, u_t)\, p(x_{t-1} \mid z_{1:t-1}, u_{1:t-1})\, dx_{t-1}$
- **Update step:**
- $p(x_t \mid z_{1:t}, u_{1:t}) = \dfrac{g(z_t \mid x_t)\, p(x_t \mid z_{1:t-1}, u_{1:t})}{\int g(z_t \mid x_t')\, p(x_t' \mid z_{1:t-1}, u_{1:t})\, dx_t'}$

Difficult to calculate!

Simplify…

Kalman Filter

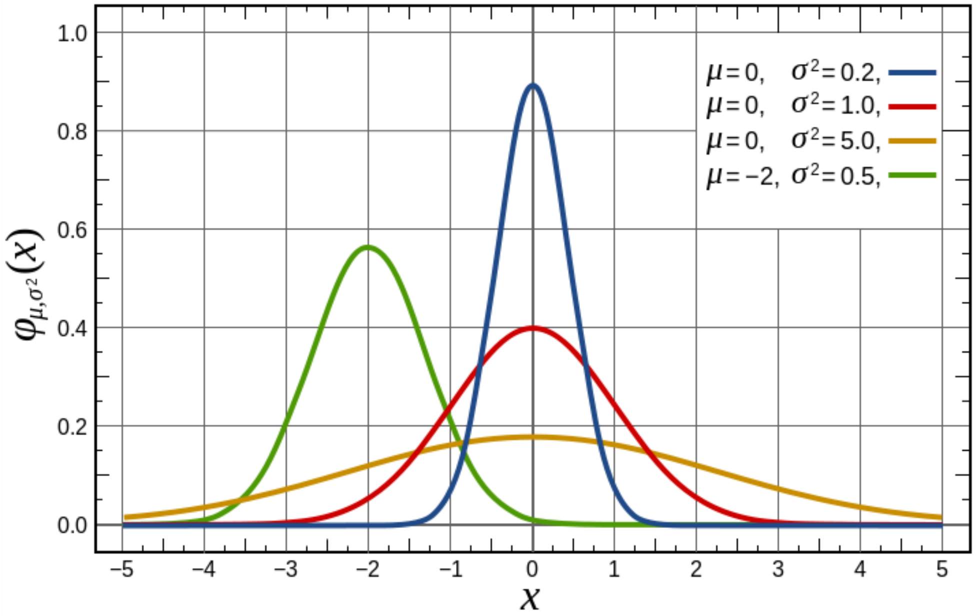
Gaussian Random Variables

Multivariate Normal (Gaussian) Distribution

- Let X be a vector of n random variables
- A multivariate normal distribution takes the form
- $f_X(x) = \frac{1}{(2\pi)^{n/2}\sqrt{\det(\Sigma)}} e^{\frac{-(x-\mu)^T\Sigma^{-1}(x-\mu)}{2}}$
- where $\mu \in \mathbf{R}^n$ and $\Sigma \in \mathbf{R}^{n \times n}$

Mean Covariance

- Fully parameterized by μ, Σ



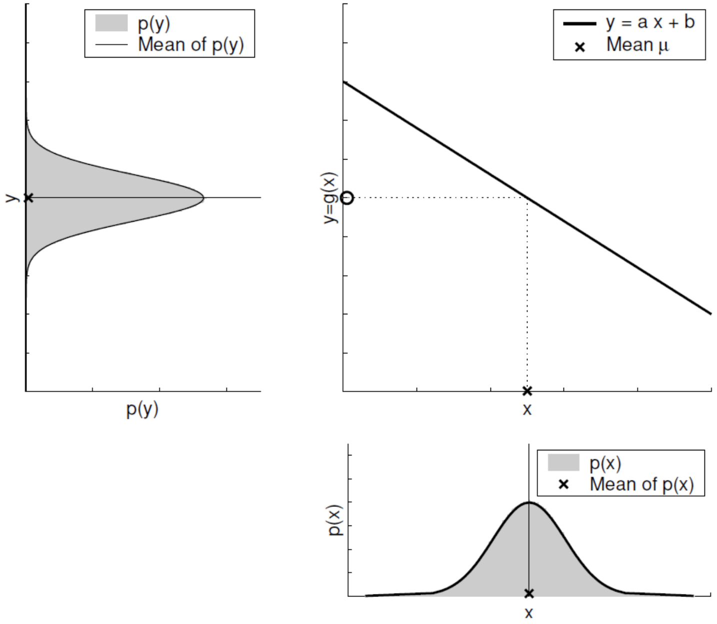
[http://en.wikipedia.org/wiki/Normal_distribution]

Affine Transformations

- Affine transformation of Gaussian distributions are Gaussian
- If $X \sim N(\mu_X, \Sigma_X)$ and $Y = AX + b$ then $Y \sim N(\mu_Y, \Sigma_Y)$ where
- $\mu_Y = A\,\mu_X + b$ and $\Sigma_Y = A\,\Sigma_X\,A^T$

$$
\begin{aligned}
\mu_Y &= E[Y] = E[AX + b] \\
&= A\,E[X] + b \\
&= A\,\mu_X + b
\end{aligned}
$$

$$
\begin{aligned}
\Sigma_Y &= E\left[(Y - \mu_Y)(Y - \mu_Y)^T\right] \\
&= E\left[(AX + b - A\mu_X - b)(AX + b - A\mu_X - b)^T\right] \\
&= E\left[(A(X - \mu_X))(A(X - \mu_X))^T\right] \\
&= A\,E\left[(X - \mu_X)(X - \mu_X)^T\right] A^T \\
&= A\,\Sigma_X\,A^T
\end{aligned}
$$



Example:

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

Bayes' Filter

- **Prior:** $p(x_0)$
- **Motion model:** $f(x_t \mid x_{t-1}, u_t)$
- **Measurement model:** $g(z_t \mid x_t)$

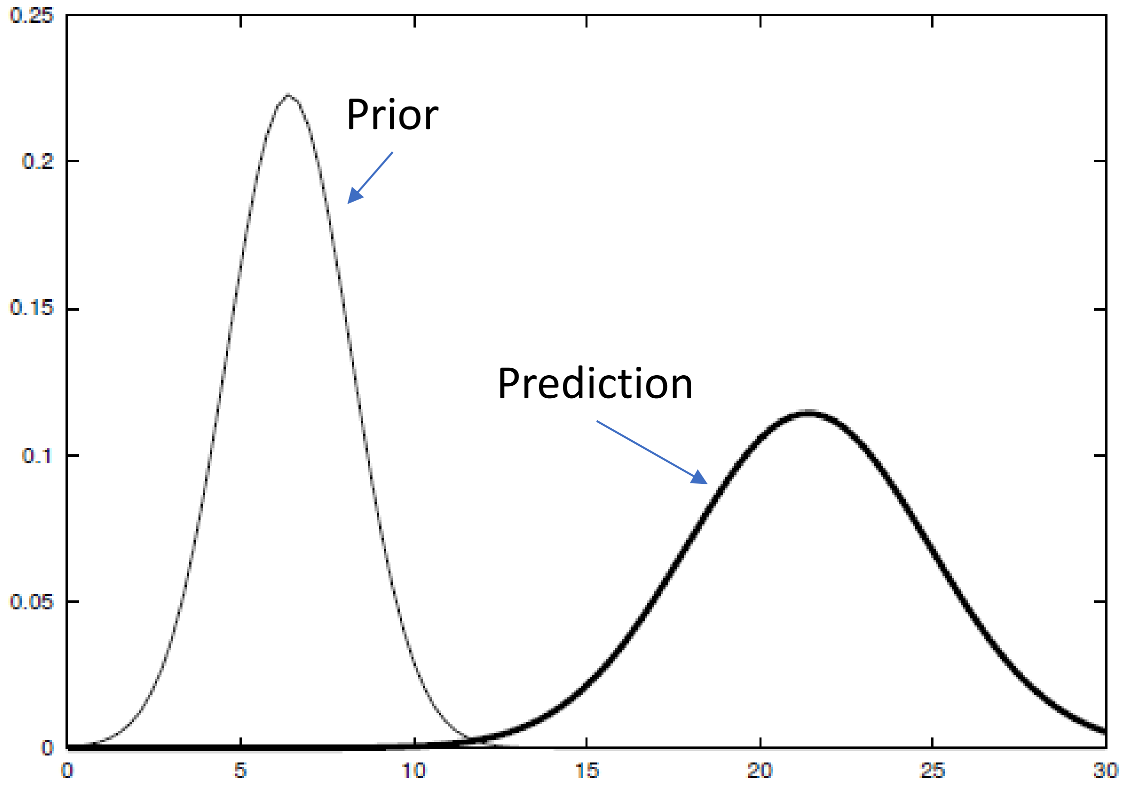
- **Prediction step:**
- $p(x_t \mid z_{1:t-1}, u_{1:t}) = \int f(x_t \mid x_{t-1}, u_t)\, p(x_{t-1} \mid z_{1:t-1}, u_{1:t-1})\, dx_{t-1}$
- **Update step:**
- $p(x_t \mid z_{1:t}, u_{1:t}) = \dfrac{g(z_t \mid x_t)\, p(x_t \mid z_{1:t-1}, u_{1:t})}{\int g(z_t \mid x_t')\, p(x_t' \mid z_{1:t-1}, u_{1:t})\, dx_t'}$

Assumptions

- Assumptions 1: The prior state of the robot is represented by a Gaussian distribution
  - $p(x_0) \sim N(\mu_0, \Sigma_0)$
- Assumptions 2: The motion model $f(x_t \mid x_{t-1}, u_t)$ is linear with additive Gaussian white noise
  - $x_t = A_t\, x_{t-1} + B_t\, u_t + n_t$
  - $n_t \sim N(0, Q_t)$
  - $x_t, n_t \in \mathbf{R}^n, u_t \in \mathbf{R}^m$, $A_t, Q_t \in \mathbf{R}^{n\times n}$, and $B_t \in \mathbf{R}^{n\times m}$
- Assumptions 3: The measurement model $g(z_t \mid x_t)$ is linear with additive Gaussian white noise
  - $z_t = C_t\, x_t + v_t$
  - $v_t \sim N(0, R_t)$
  - $z_t, v_t \in \mathbf{R}^p, C_t \in \mathbf{R}^{p\times n}$, and $R_t \in \mathbf{R}^{p\times p}$

Kalman Filter – Prediction

- Bayes:
  - $p(x_t \mid z_{1:t-1}, u_{1:t}) = \int f(x_t \mid x_{t-1}, u_t)\, p(x_{t-1} \mid z_{1:t-1}, u_{1:t-1})\, dx_{t-1}$
- $x_t = A_t\, x_{t-1} + B_t u_t + n_t$
- $n_t \sim N(0, Q_t)$
- Prior: $p(x_{t-1} \mid z_{1:t-1}, u_{1:t-1}) \sim N(\mu_{t-1}, \Sigma_{t-1})$
- Prediction:
  - $\bar{\mu}_t = A\, \mu_{t-1} + B\, u_t$
  - $\bar{\Sigma}_t = A\, \Sigma_{t-1}\, A^T + Q$



Bayes' Filter

State

Control input

- **Prior:** $p(x_0)$
- **Motion model:** $f(x_t \mid x_{t-1}, u_t)$
- **Measurement model:** $g(z_t \mid x_t)$

Measurement

- **Prediction step:**
- $p(x_t \mid z_{0:t-1}, u_{1:t}) = \int f(x_t \mid x_{t-1}, u_t)\, p(x_{t-1} \mid z_{1:t-1}, u_{1:t-1})\, dx_{t-1}$
- **Update step:**
- $p(x_t \mid z_{1:t}, u_{1:t}) = \dfrac{g(z_t \mid x_t)\, p(x_t \mid z_{1:t-1}, u_{1:t})}{\int g(z_t \mid x_t')\, p(x_t' \mid z_{1:t-1}, u_{1:t})\, dx_t'}$

Kalman Filter – Update

Update step:

$$p(x_t \mid z_{1:t}, u_{1:t}) = \frac{g(z_t \mid x_t)\, p(x_t \mid z_{1:t-1}, u_{1:t})}{\int g(z_t \mid x'_t)\, p\big(x'_t \,\big|\, z_{1:t-1}, u_{1:t}\big)\, dx'_t}$$

- The observation model is $z_t = C_t\, \bar{x}_t + v_t, \quad v_t \sim N(0, R_t)$
- The best estimation without a measurement is to set $x_t = \bar{x}_t$
- $\begin{bmatrix} x_t \\ z_t \end{bmatrix} = \begin{bmatrix} I & 0 \\ C & I \end{bmatrix} \begin{bmatrix} \bar{x}_t \\ v_t \end{bmatrix}$
- $\mu = \begin{bmatrix} \bar{\mu}_t \\ C\bar{\mu}_t \end{bmatrix}$
- $\Sigma = \begin{bmatrix} I & 0 \\ C & I \end{bmatrix} \begin{bmatrix} \bar{\Sigma}_t & 0 \\ 0 & R \end{bmatrix} \begin{bmatrix} I & C^T \\ 0 & I \end{bmatrix} = \begin{bmatrix} \bar{\Sigma}_t & \bar{\Sigma}_t C^T \\ C\bar{\Sigma}_t & C\bar{\Sigma}_t C^T + R \end{bmatrix}$
- $p(x_t \mid z_t)$

Conditional Distributions

- Let $X = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}$ be a multivariate Gaussian with mean $\mu = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}$ and covariance $\Sigma = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix}$
- Then the conditional density $f_{X_1|X_2}(x_1 | X_2 = x_2)$ is a multivariate normal distribution with
  - mean $\mu_{X_1|X_2} = \mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(x_2 - \mu_2)$
  - covariance $\Sigma_{X_1|X_2} = \Sigma_{11} - \Sigma_{12}\,\Sigma_{22}^{-1}\,\Sigma_{21}$

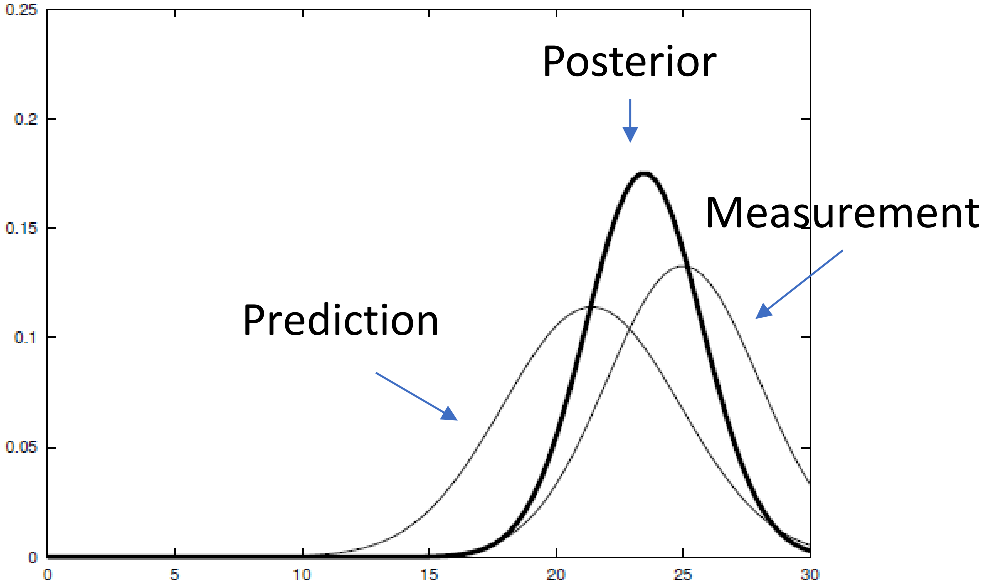
Kalman Filter – Update

- The distribution of x_t conditioned on z_t is thus normal with
- $\mu_{x_t|z_t} = \bar{\mu}_t + \bar{\Sigma}_t C^T \left(C\bar{\Sigma}_t C^T + R\right)^{-1} (z_t - C\bar{\mu}_t)$
- $\Sigma_{x_t|z_t} = \bar{\Sigma}_t - \bar{\Sigma}_t C^T \left(C\bar{\Sigma}_t C^T + R\right)^{-1} C\bar{\Sigma}_t$

Conditional Distributions

- Let $X = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}$ be a multivariate Gaussian with mean $\mu = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}$ and covariance $\Sigma = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix}$
- Then the conditional density $f_{X_1|X_2}(x_1|X_2 = x_2)$ is a multivariate normal distribution with
  - mean $\mu_{X_1|X_2} = \mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(x_2 - \mu_2)$
  - covariance $\Sigma_{X_1|X_2} = \Sigma_{11} - \Sigma_{12}\,\Sigma_{22}^{-1}\,\Sigma_{21}$

- Define the Kalman gain K_t
- $K_t = \bar{\Sigma}_t C^T \left(C\bar{\Sigma}_t C^T + R\right)^{-1}$
- $\mu_t = \bar{\mu}_t + K_t(z_t - C\bar{\mu}_t)$
- $\Sigma_t = \bar{\Sigma}_t - K_t C\bar{\Sigma}_t$



Kalman Gain

- $K_t = \bar{\Sigma}_t \, C^T \left(C \, \bar{\Sigma}_t \, C^T + R \right)^{-1}$
- **Intuition:** How much to trust the sensor vs. the prediction
- **Example:**
  - Perfect sensor $R = 0$
    - $K_t = \bar{\Sigma}_t \, C^T \left(C \, \bar{\Sigma}_t \, C^T + R \right)^{-1} = C^{-1}$
    - $\mu_t = \bar{\mu}_t + K_t(z_t - C \, \bar{\mu}_t) = C^{-1} z_t$
    - $\Sigma_t = \bar{\Sigma}_t - K_t \, C \, \bar{\Sigma}_t = 0$
  - Horrible sensor $R \to \infty$
    - $K_t = \bar{\Sigma}_t \, C^T \left(C \, \bar{\Sigma}_t \, C^T + R \right)^{-1} \to 0$
    - $\mu_t = \bar{\mu}_t + K_t(z_t - C \, \bar{\mu}_t) \to \bar{\mu}_t$
    - $\Sigma_t = \bar{\Sigma}_t - K_t \, C \, \bar{\Sigma}_t \to \bar{\Sigma}_t$

Kalman Filter

- Prior:
  - $p(x_0) \sim N(\mu_0, \Sigma_0)$
- Motion model:
  - $x_t = A_t \; x_{t-1} + B_t \; u_t + n_t$
  - $n_t \sim N(0, Q_t)$
- Measurement model:
  - $z_t = C_t \, x_t + v_t$
  - $v_t \sim N(0, R_t)$

- Prediction:
  - $\bar{\mu}_1 = A_1 \; \mu_0 + B_1 \; u_1$
  - $\bar{\Sigma}_1 = A_1 \, \Sigma_0 \, A_1^T + Q_1$
- Update:
  - $K_1 = \bar{\Sigma}_1 \, C_1^T \left(C_1 \, \bar{\Sigma}_1 \, C_1^T + R_1\right)^{-1}$
  - $\mu_1 = \bar{\mu}_1 + K_1 (z_1 - C_1 \, \bar{\mu}_1)$
  - $\Sigma_1 = \bar{\Sigma}_1 - K_1 \, C_1 \, \bar{\Sigma}_1$
- Prior:
  - μ_1, Σ_1

- Prior:
  - μ_{t-1}, Σ_{t-1}
- Prediction:
  - $\bar{\mu}_t = A_t \; \mu_{t-1} + B_t \, u_t$
  - $\bar{\Sigma}_t = A_t \, \Sigma_{t-1} \, A_t^T + Q_t$
- Update:
  - $K_t = \bar{\Sigma}_t \, C_t^T \left(C_t \, \bar{\Sigma}_t \, C_t^T + R_t\right)^{-1}$
  - $\mu_t = \bar{\mu}_t + K_t (z_t - C_t \, \bar{\mu}_t)$
  - $\Sigma_t = \bar{\Sigma}_t - K_t \, C_t \, \bar{\Sigma}_t$

Continuous Time Systems

Discrete vs. Continuous Time

Discrete Time

- Events occur at discrete points in time
- Time intervals often evenly spaced

- Example:
  - $x_t = A_t\ x_{t-1} + B_t\ u_t + n_t$

Continuous Time

- Events may occur close to each other in time

- Example:
  - $\dot{x} = f(x, u, n) = A\ x + B\ u + U\ n$

Continuous Dynamics

- $\dot{x} = f(x, u, n) = A\, x + B\, u + U\, n$

- One-step Euler integration (First order approximation)
  - $x_t = x_{t-1} + f(x_{t-1}, u_t, n_t)\, \delta t$
  - $x_t = (I + \delta t\, A)\, x_{t-1} + (\delta t\, B)\, u_t + (\delta t\, U)\, n_t$
  - $x_t = F\, x_{t-1} + G\, u_t + V\, n_t$
- Prediction:
  - $\bar{\mu}_t = F\, \mu_{t-1} + G\, u_t$
  - $\bar{\Sigma}_t = F\, \Sigma_{t-1} F^T + V\, Q\, V^T$

The derivative is fixed in a short period of time δt.

Kalman Filter Discussion

- **Advantages:**
  - Simple
  - Purely matrix operations
    - Computationally efficient, even for high dimensional systems
- **Disadvantages:**
  - Assumes everything is linear and Gaussian

Extended (to handle nonlinear systems) Kalman Filter

Assumptions for EKF

- The prior state of the robot is represented by a Gaussian distribution
  - $p(x_0) \sim N(\mu_0, \Sigma_0)$
- The continuous time motion model is:
  - $\dot{x} = f(x, u, n)$
  - $n_t \sim N(0, Q_t)$ is Gaussian white noise
- The measurement model is:
  - $z = g(x, v)$
  - $v_t \sim N(0, R_t)$ is Gaussian white noise

Prediction – Linearization

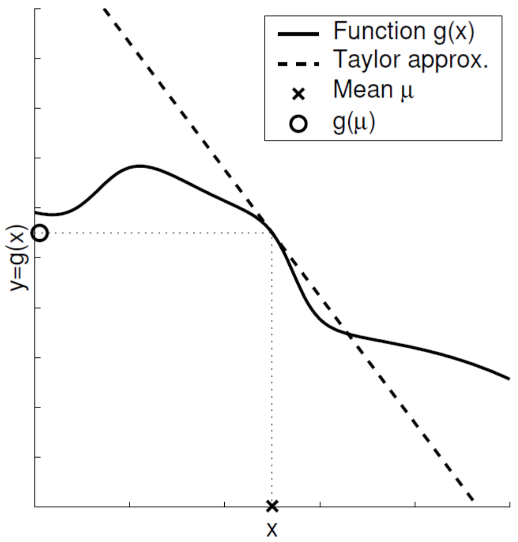
The continuous time motion model is:

$$\dot{x} = f(x, u, n)$$

- Linearize the dynamics with $x = \mu_{t-1},\ u = u_t,\ n = 0$

Taylor expansion

  - $\dot{x} \approx f(\mu_{t-1}, u_t, 0) + \left.\frac{\partial f}{\partial x}\right|_{\mu_{t-1}, u_t, 0} (x - \mu_{t-1}) + \left.\frac{\partial f}{\partial u}\right|_{\mu_{t-1}, u_t, 0} (u - u_t) + \left.\frac{\partial f}{\partial n}\right|_{\mu_{t-1}, u_t, 0} (n - 0)$



- Let:
  - $A_t = \left.\frac{\partial f}{\partial x}\right|_{\mu_{t-1}, u_t, 0}$
  - $B_t = \left.\frac{\partial f}{\partial u}\right|_{\mu_{t-1}, u_t, 0}$
  - $U_t = \left.\frac{\partial f}{\partial n}\right|_{\mu_{t-1}, u_t, 0}$
- Linear dynamics:
  - $\dot{x} \approx f(\mu_{t-1}, u_t, 0) + A_t (x - \mu_{t-1}) + B_t (u - u_t) + U_t (n - 0)$

Prediction – Discrete Time

- Linear dynamics:
  - $\dot{x} \approx f(\mu_{t-1}, u_t, 0) + A_t(x - \mu_{t-1}) + B_t(u - u_t) + U_t(n - 0)$

- One-step Euler integration
  - $\bar{x}_t \approx x_{t-1} + f(x_{t-1}, u_t, n_t)\,\delta t$
  - $\approx x_{t-1} + \delta t\, f(\mu_{t-1}, u_t, 0) + \delta t\, A_t\,(x_{t-1} - \mu_{t-1}) + \delta t\, B_t(u_t - u_t) + \delta t\, U_t(n_t - 0)$
  - $\approx x_{t-1} + \delta t\, f(\mu_{t-1}, u_t, 0) + \delta t\, A_t\,(x_{t-1} - \mu_{t-1}) + \delta t\, U_t(n_t - 0)$
  - $\approx (I + \delta t\, A_t)\, x_{t-1} + \delta t\, U_t\, n_t + \delta t(f(\mu_{t-1}, u_t, 0) - A_t\, \mu_{t-1})$
  - $\approx F_t\, x_{t-1} + V_t\, n_t + \delta t(f(\mu_{t-1}, u_t, 0) - A_t\, \mu_{t-1})$
- Prediction:
  - $\bar{\mu}_t = \mu_{t-1} + \delta t\, f(\mu_{t-1}, u_t, 0)$
  - $\bar{\Sigma}_t = F_t\, \Sigma_{t-1}\, F_t^T + V_t\, Q_t\, V_t^T$

Update – Linearization

The measurement model is:

$$z = g(x, v)$$

- Linearize the measurement model about $x = \bar{\mu}_t,\ n = 0$
  - $g(x, v) \approx g(\bar{\mu}_t, 0) + \left.\frac{\partial g}{\partial x}\right|_{\bar{\mu}_t, 0} (x - \bar{\mu}_t) + \left.\frac{\partial g}{\partial n}\right|_{\bar{\mu}_t, 0} (n - 0)$
- Let:
  - $C_t = \left.\frac{\partial g}{\partial x}\right|_{\bar{\mu}_t, 0}$
  - $W_t = \left.\frac{\partial g}{\partial n}\right|_{\bar{\mu}_t, 0}$
- Linear observation model:
  - $z_t = g(x_t, v_t) \approx g(\bar{\mu}_t, 0) + C_t\,(x_t - \bar{\mu}_t) + W_t\,n_t$

- Update:
  - $K_t = \bar{\Sigma}_t\,C_t^T\left(C_t\,\bar{\Sigma}_t\,C_t^T + W_t\,R\,W_t^T\right)^{-1}$
  - $\mu_t = \bar{\mu}_t + K_t\big(z_t - g(\bar{\mu}_t, 0)\big)$
  - $\Sigma_t = \bar{\Sigma}_t - K_t\,C_t\,\bar{\Sigma}_t$

Extended Kalman Filter

- Motion Model:
  - $\dot{x} = f(x, u, n)$
  - $n_t \sim N(0, Q_t)$

  $\Big\}$ Assumptions
  - $A_t = \left.\frac{\partial f}{\partial x}\right|_{\mu_{t-1}, u_t, 0}$
  - $U_t = \left.\frac{\partial f}{\partial n}\right|_{\mu_{t-1}, u_t, 0}$

  $\Big\}$ Linearization
  - $F_t = I + \delta t\, A_t$
  - $V_t = \delta t\, U_t$

  $\Big\}$ Discretization

- Prediction step:
  - $\bar{\mu}_t = \mu_{t-1} + \delta t\, f(\mu_{t-1}, u_t, 0)$
  - $\bar{\Sigma}_t = F_t\, \Sigma_{t-1}\, F_t^T + V_t\, Q_t\, V_t^T$

- Measurement Model:
  - $z_t = g(x_t, n_t)$
  - $v_t \sim N(0, R_t)$

  $\Big\}$ Assumptions
  - $C_t = \left.\frac{\partial g}{\partial x}\right|_{\bar{\mu}_t, 0}$
  - $W_t = \left.\frac{\partial g}{\partial n}\right|_{\bar{\mu}_t, 0}$

  $\Big\}$ Linearization

- Update step:
  - $K_t = \bar{\Sigma}_t\, C_t^T \left(C_t\, \bar{\Sigma}_t\, C_t^T + W_t\, R\, W_t^T\right)^{-1}$
  - $\mu_t = \bar{\mu}_t + K_t\,(z_t - g(\bar{\mu}_t, 0))$
  - $\Sigma_t = \bar{\Sigma}_t - K_t\, C_t\, \bar{\Sigma}_t$

Recap

Bayes Filter → Kalman Filter → Extended Kalman Filter → Error State Extended Kalman Filter

- Markov Property

- Prior state is Gaussian distribution
- Motion model is linear with Gaussian noise
- Measurement model is linear with Gaussian noise

- Linearize the motion model
- Linearize the measurement model

$$\mathbf{x} = \hat{\mathbf{x}} \oplus \delta\mathbf{x}$$

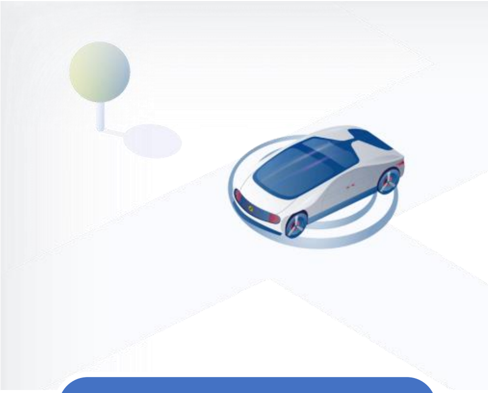
Nominal State Error state

- Linearize on error state

[1] Sola, Joan. "Quaternion kinematics for the error-state Kalman filter." *arXiv preprint arXiv:1711.02508* (2017).

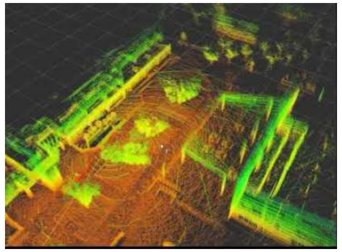
Content

Example: EKF for Vehicle 2D State Estimation



$$\boldsymbol{x} = [x, y, \theta]$$

GNSS



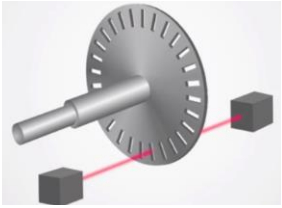
Lidar

Move

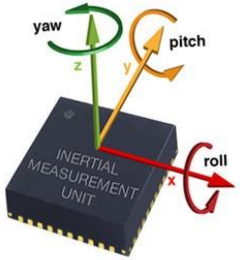
Predict

Sense

Update

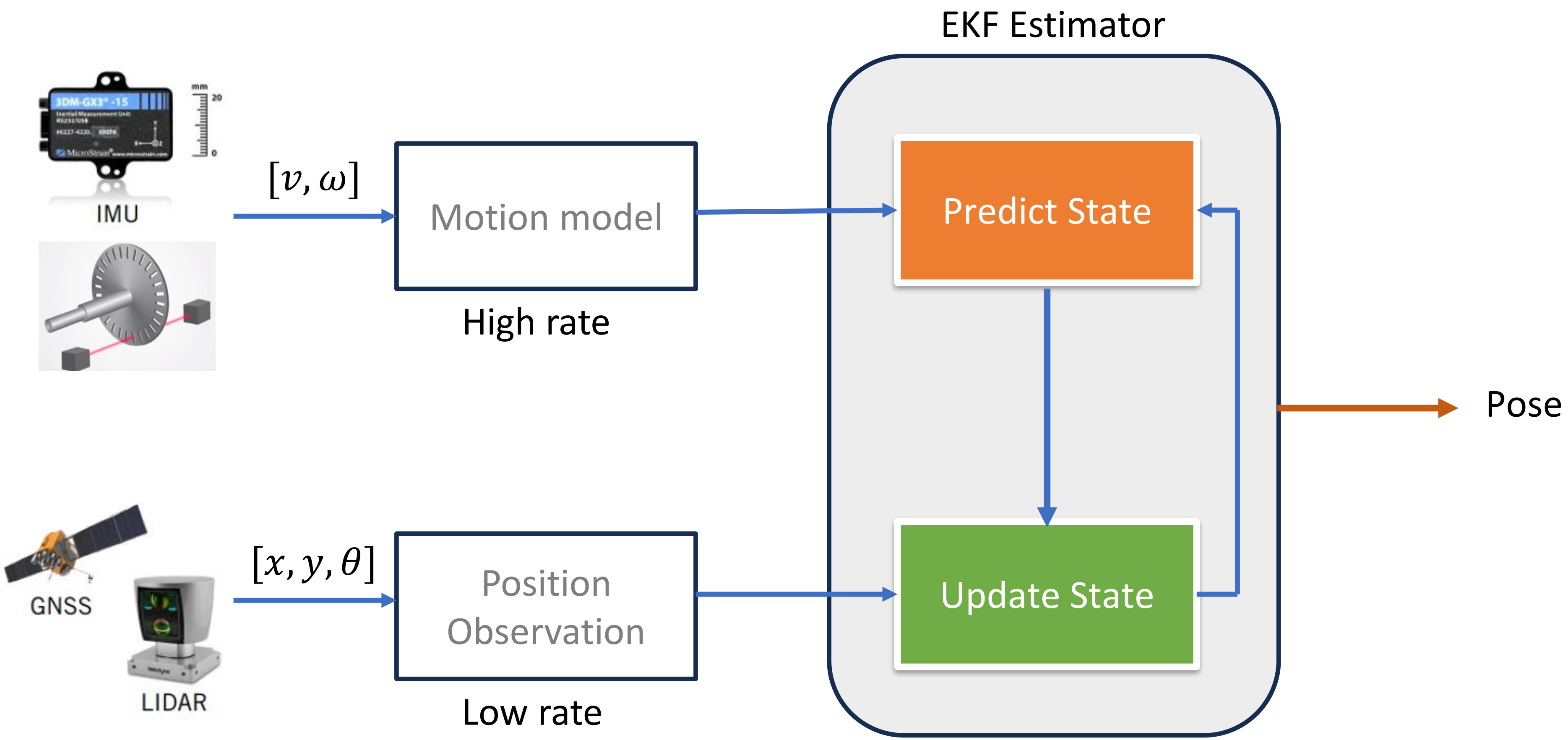


Wheel odometer

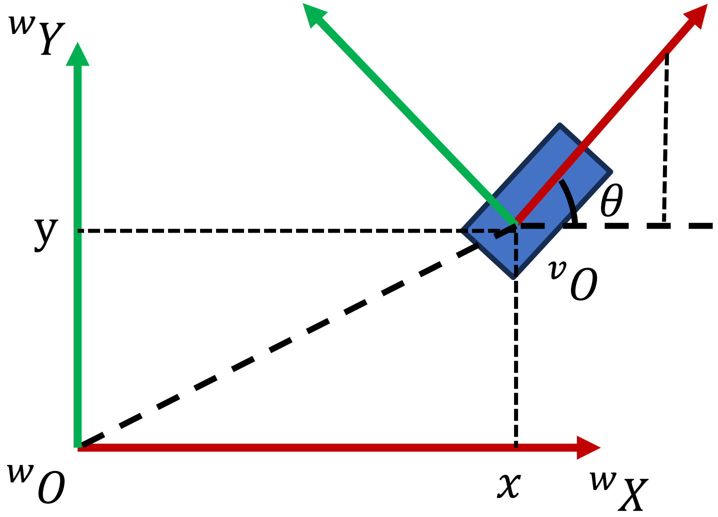


IMU
(only use angular velocity along z-axis)

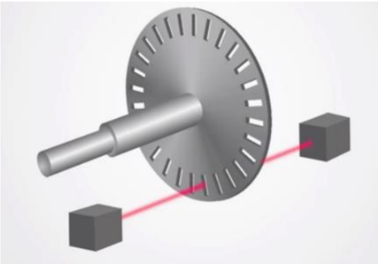
Example: EKF for Vehicle 2D State Estimation



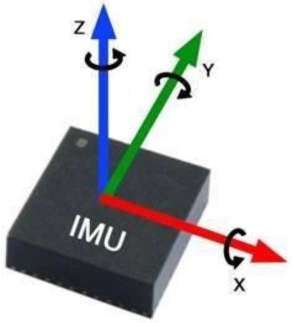
Example: EKF for Vehicle State Estimation



Motion sensor:



Velocity



Angular velocity

Vehicle 2D State: $\boldsymbol{x} = [x, y, \theta]$

Motion signal： $\boldsymbol{u} = [v, \omega]$ velocity, angular velocity

with Gaussian white noise

$n_v \sim N(0, Q_v),\ n_\omega \sim N(0, Q_\omega)$

Motion model:

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = f(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{n}) = \begin{bmatrix} \cos(\theta) \cdot (v + n_v) \\ \sin(\theta) \cdot (v + n_v) \\ \omega + n_\omega \end{bmatrix}$$

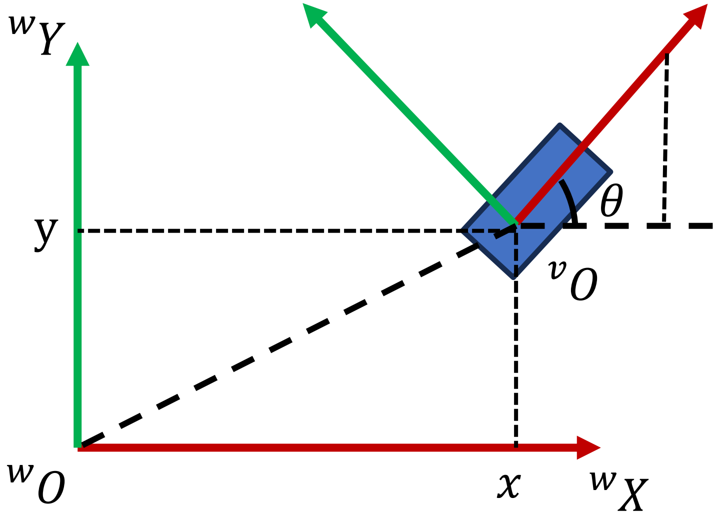
Linearization:

$$A_t = \frac{\partial f}{\partial \boldsymbol{x}}\bigg|_{\boldsymbol{x}_{t-1}, \boldsymbol{u}_t, 0} = \begin{bmatrix} 0 & 0 & -v \cdot \sin(\theta) \\ 0 & 0 & v \cdot cos(\theta) \\ 0 & 0 & 0 \end{bmatrix}$$

$$U_t = \frac{\partial f}{\partial \boldsymbol{n}}\bigg|_{\boldsymbol{x}_{t-1}, \boldsymbol{u}_t, 0} = \begin{bmatrix} \cos(\theta) & 0 \\ \sin(\theta) & 0 \\ 0 & 1 \end{bmatrix}$$

Example: EKF for Vehicle State Estimation

Vehicle 2D State: $\boldsymbol{x} = [x, y, \theta]$



Measurement: $\boldsymbol{z} = [x, y, \theta]$ position and yaw angle

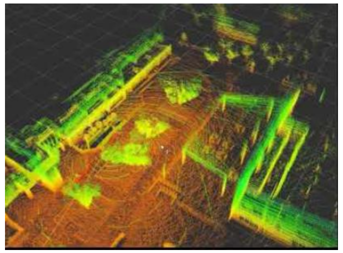
with Gaussian white noise

$\boldsymbol{n}_z \sim N(0, R_z)$

Measurement model:

$$z_t = g(x_t, n_t) = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix} + \boldsymbol{n}_z$$

Localization sensor:



Position and orientation

Linearization:

$$C_t = \left.\frac{\partial g}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_t,0} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$W_t = \left.\frac{\partial f}{\partial \boldsymbol{n}}\right|_{\boldsymbol{x}_t,0} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Example: EKF for Vehicle State Estimation

- Motion Model:
  - $\dot{x} = f(x, u, n)$
  - $n_t \sim N(0, Q_t)$

  Assumptions

  - $A_t = \left.\frac{\partial f}{\partial x}\right|_{\mu_{t-1}, u_t, 0}$
  - $U_t = \left.\frac{\partial f}{\partial n}\right|_{\mu_{t-1}, u_t, 0}$

  Linearization

  - $F_t = I + \delta t\, A_t$
  - $V_t = \delta t\, U_t$

  Discretization

- Prediction step:
  - $\bar{\mu}_t = \mu_{t-1} + \delta t\, f(\mu_{t-1}, u_t, 0)$
  - $\bar{\Sigma}_t = F_t\, \Sigma_{t-1}\, F_t^T + V_t\, Q_t\, V_t^T$

- Measurement Model:
  - $z_t = g(x_t, n_t)$
  - $v_t \sim N(0, R_t)$

  Assumptions

  - $C_t = \left.\frac{\partial g}{\partial x}\right|_{\bar{\mu}_t, 0}$
  - $W_t = \left.\frac{\partial g}{\partial n}\right|_{\bar{\mu}_t, 0}$

  Linearization

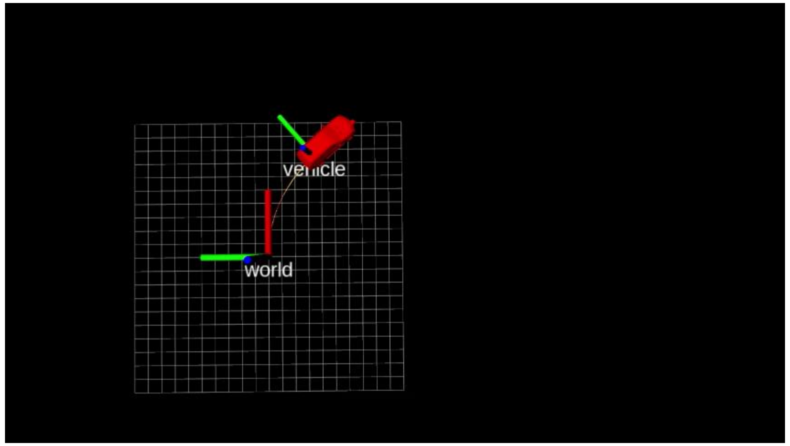
- Update step:
  - $K_t = \bar{\Sigma}_t\, C_t^T \left(C_t\, \bar{\Sigma}_t\, C_t^T + W_t\, R\, W_t^T\right)^{-1}$
  - $\mu_t = \bar{\mu}_t + K_t\, (z_t - g(\bar{\mu}_t, 0))$
  - $\Sigma_t = \bar{\Sigma}_t - K_t\, C_t\, \bar{\Sigma}_t$

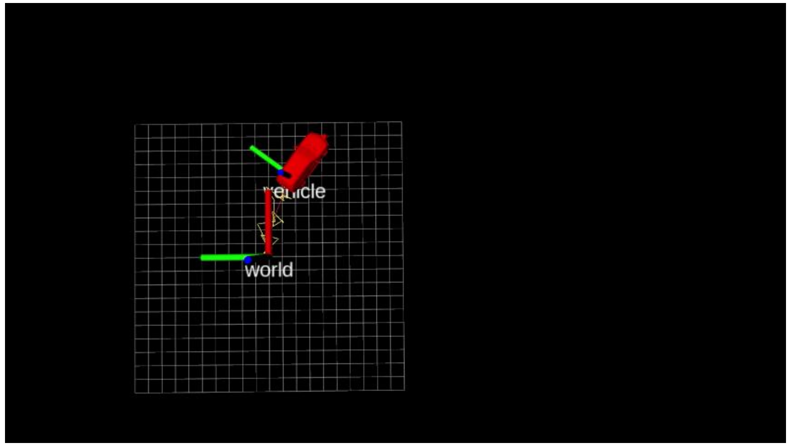
Example: EKF for Vehicle State Estimation

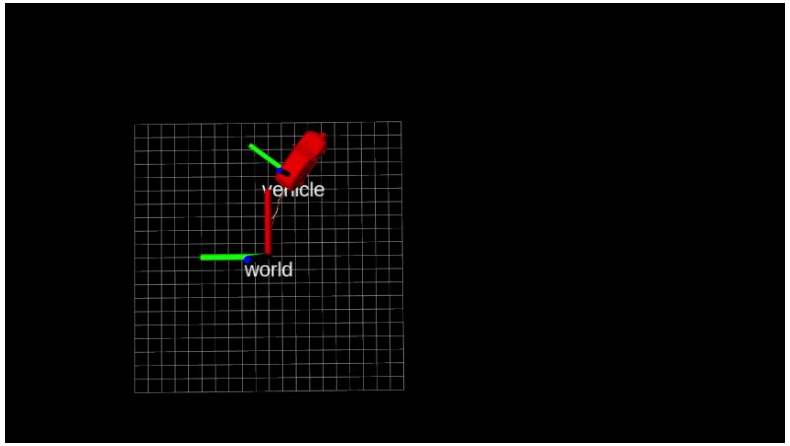
- Motion Prediction Only:
  (IMU + Wheel)
  - Smooth
  - Drift

- Measurement Only:
  (GPS)
  - Noisy
  - No drift

- EKF Fusion:
  (IMU + Wheel + GPS)
  - Smooth
  - No drift







Content

1. State Estimation Background

2. Extended Kalman Filter Derivation

3. Example: Vehicle State Estimation

4. Assignment

Assignment

Implement the EKF example with code:

- Predict with motion model (IMU + Wheel)
  - complete the function *EkfPredict()*

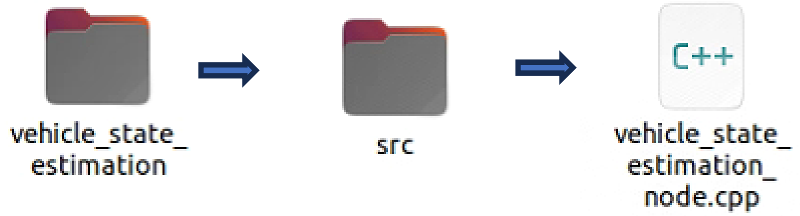
```
void EkfPredict(State& state, const double &time, const double &velocity, const double &yaw_rate) {
    // printf("time %lf, velocity %lf, yaw_rate %lf \n", time, velocity, yaw_rate);
    // YOUR_CODE_HERE
    // todo: implement the EkfPredict function


    // printf("after predict x: %lf, y: %lf, yaw: %lf \n", state.x, state.y, state.yaw);
}
```

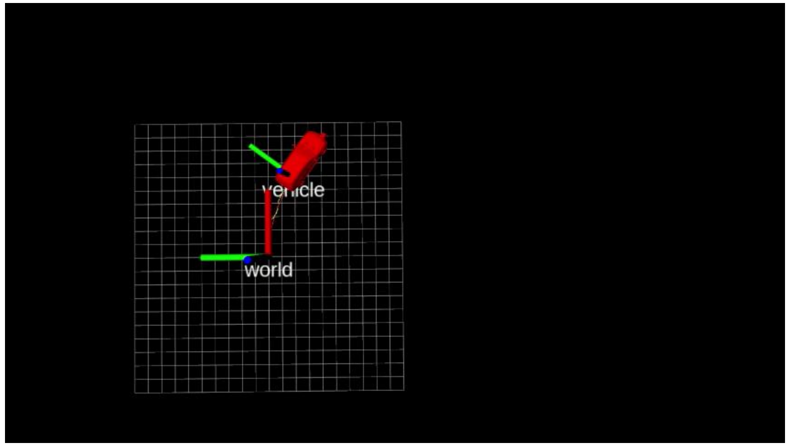
- Update with measurement model (GPS/Lidar)
  - complete the function *EkfUpdate()*

```
void EkfUpdate(State& state,  const double &m_x, const double &m_y, const double &m_yaw) {
    // printf("time :%lf \n", state.time);
    // printf("before update x: %lf, y: %lf, yaw: %lf \n", state.x, state.y, state.yaw);
    // printf("measure x: %lf, y: %lf, yaw: %lf \n", m_x, m_y, m_yaw);
    // YOUR_CODE_HERE
    // todo: implement the EkfUpdate function


    // printf("after update x: %lf, y: %lf, yaw: %lf \n", state.x, state.y, state.yaw);
}
```



Expected result:



Assignment

Implement the EKF example with code:

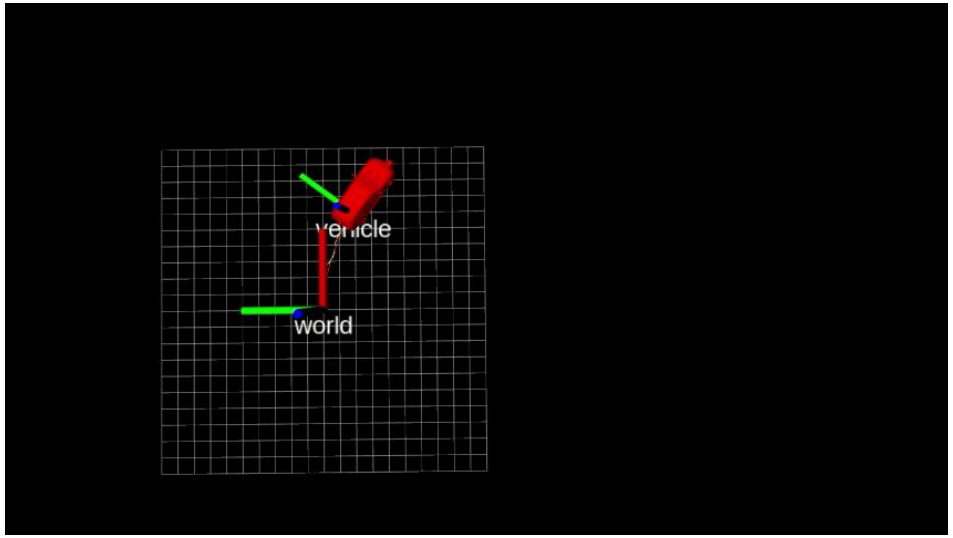
- Predict with motion model (IMU + Wheel)
  - complete the function *EkfPredict()*
- Update with measurement model (GPS)
  - complete the function *EkfUpdate()*

```
cd ~/catkin_ws/src
catkin_make
```

```
roscore
```

```
source ~/catkin_ws/devel/setup.bash
rosrun vehicle_state_estimaiton vehicle_state_estimation
```

```
source ~/catkin_ws/devel/setup.bash
rosrun rviz rviz –d  ~/catkin_ws/src/ vehicle_state_estimaiton /state_estimaiton.rviz
```



Submission:
Write a report to explain how to complete the EKF function, and attach your source code and the screenshot of your result in RVIZ.

深蓝学院
shenlanxueyuan.com

感谢聆听！

Thanks for Listening