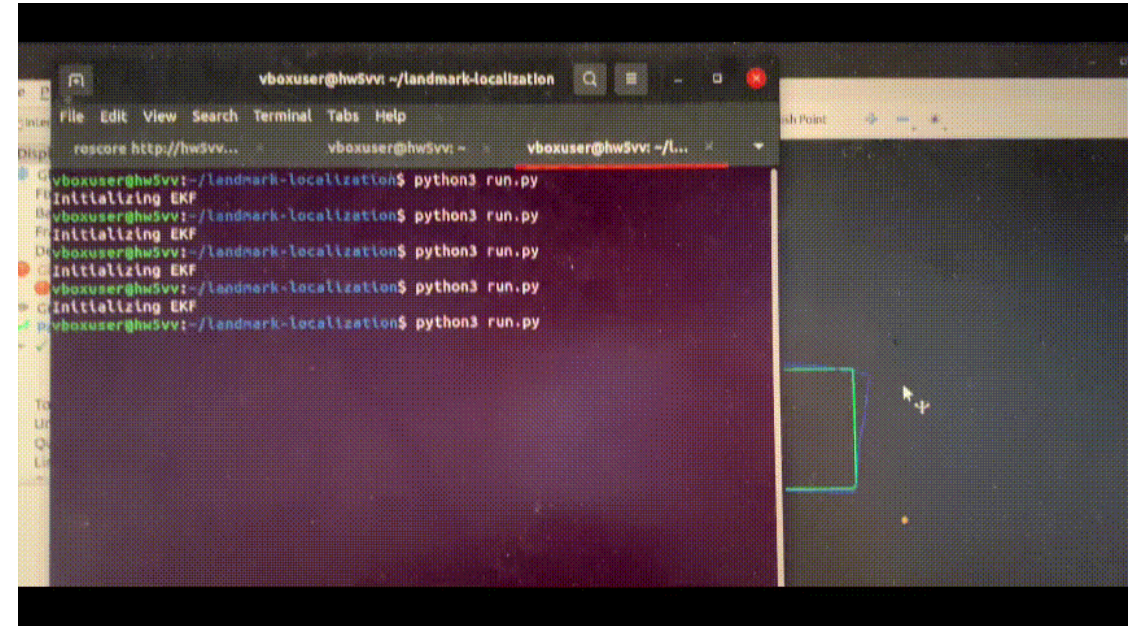


# EKF, UKF, PF

Harry Chen

# EKF result

- **Green path**: the ideal command path
- **Blue path**: the path that the robot moves due to action noise
- **Red arrow**: the robot pose
- **Red ellipse**: the covariance



# Extended Kalman filter

- Prediction step
  - Predict the next state with the motion model
  - Update covariance
- Correction step
  - Compute Kalman gain
  - Correct state and covariance

$$\begin{aligned}x_{k+1} &= g(x_k, u_k) \\ P_{k+1} &= GP_kG^T + VMV^T\end{aligned}$$

$$\begin{aligned}K_k &= P_k H_k^T (H_k P_k H_k^T + Q)^{-1} \\ \hat{x}_k &= x_k + K_k (z_k - h(x_k)) \\ \hat{p}_k &= (1 - K_k H_k) P_k\end{aligned}$$

# Unscented Kalman filter

- Prediction step
  - Compute sigma points
  - Propagate sigma points
  - Predict the next state and update covariance
- Correction step
  - Predict measurements from redrawn sigma points
  - Estimate mean and covariance of predicted measurements
  - Compute Kalman gain
  - Correct state and covariance

$$\hat{\mathbf{L}}_{k-1} \hat{\mathbf{L}}_{k-1}^T = \hat{\mathbf{P}}_{k-1}$$

$$\hat{\mathbf{x}}_{k-1}^{(0)} = \hat{\mathbf{x}}_{k-1}$$

$$\hat{\mathbf{x}}_{k-1}^{(i)} = \hat{\mathbf{x}}_{k-1} + \sqrt{N + \kappa} \text{col}_i \hat{\mathbf{L}}_{k-1} \quad i = 1 \dots N$$

$$\hat{\mathbf{x}}_{k-1}^{(i+N)} = \hat{\mathbf{x}}_{k-1} - \sqrt{N + \kappa} \text{col}_i \hat{\mathbf{L}}_{k-1} \quad i = 1 \dots N$$

$$\check{\mathbf{x}}_k^{(i)} = \mathbf{f}_{k-1}(\hat{\mathbf{x}}_{k-1}^{(i)}, \mathbf{u}_{k-1}, \mathbf{0}) \quad i = 0 \dots 2N$$

$$\alpha^{(i)} = \begin{cases} \frac{\kappa}{N + \kappa} & i = 0 \\ \frac{1}{2} \frac{1}{N + \kappa} & \text{otherwise} \end{cases}$$

$$\check{\mathbf{x}}_k = \sum_{i=0}^{2N} \alpha^{(i)} \check{\mathbf{x}}_k^{(i)}$$

$$\check{\mathbf{P}}_k = \sum_{i=0}^{2N} \alpha^{(i)} \left( \check{\mathbf{x}}_k^{(i)} - \check{\mathbf{x}}_k \right) \left( \check{\mathbf{x}}_k^{(i)} - \check{\mathbf{x}}_k \right)^T + \underset{\uparrow}{\mathbf{Q}_{k-1}}$$

$$\hat{\mathbf{y}}_k^{(i)} = \mathbf{h}_k(\check{\mathbf{x}}_k^{(i)}, \mathbf{0}) \quad i = 0, \dots, 2N$$

$$\hat{\mathbf{y}}_k = \sum_{i=0}^{2N} \alpha^{(i)} \hat{\mathbf{y}}_k^{(i)}$$

$$\mathbf{P}_y = \sum_{i=0}^{2N} \alpha^{(i)} \left( \hat{\mathbf{y}}_k^{(i)} - \hat{\mathbf{y}}_k \right) \left( \hat{\mathbf{y}}_k^{(i)} - \hat{\mathbf{y}}_k \right)^T + \underset{\uparrow}{\mathbf{R}_k}$$

$$\mathbf{P}_{xy} = \sum_{i=0}^{2N} \alpha^{(i)} \left( \check{\mathbf{x}}_k^{(i)} - \check{\mathbf{x}}_k \right) \left( \hat{\mathbf{y}}_k^{(i)} - \hat{\mathbf{y}}_k \right)^T$$

$$\mathbf{K}_k = \mathbf{P}_{xy} \mathbf{P}_y^{-1}$$

$$\hat{\mathbf{x}}_k = \check{\mathbf{x}}_k + \mathbf{K}_k (\mathbf{y}_k - \hat{\mathbf{y}}_k)$$

$$\hat{\mathbf{P}}_k = \check{\mathbf{P}}_k - \mathbf{K}_k \mathbf{P}_y \mathbf{K}_k^T$$

# Particle filter

- Prediction step
  - Generate samples by applying random motion  $\hat{u}_k$  to particles
- Correction step
  - Update particle weights by computing likelihoods of the current measurements
  - Normalize weights
- Resampling step
  - Choose a random number  $r$  and select those particles that correspond to  $r + (m - 1)M^{-1}$  where  $m = 1, \dots, M$  and  $r \in \left[0, \frac{1}{M}\right]$

$$\hat{v} = v + \epsilon_v, \quad \epsilon_v \sim \mathcal{N}(0, \alpha_1 v^2 + \alpha_2 \omega^2),$$

$$\hat{\omega} = \omega + \epsilon_\omega, \quad \epsilon_\omega \sim \mathcal{N}(0, \alpha_3 v^2 + \alpha_4 \omega^2),$$

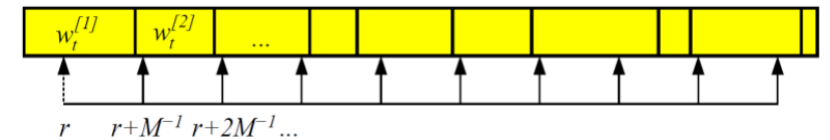
$$\hat{\gamma} = \epsilon_\gamma, \quad \epsilon_\gamma \sim \mathcal{N}(0, \alpha_5 v^2 + \alpha_6 \omega^2).$$

$$x_{k+1} = g(x_k, \hat{u}_k)$$

$$diff = z_k - h(x_k)$$

$$w_i = P(diff | \mathcal{N}(0, Q))$$

$$w_i = \frac{w_i}{\sum w_i}$$



# Motion model

- Motion model
  - Inputs are linear velocity  $\hat{v}$  and angular velocity  $\hat{\omega}$
  - The third input  $\hat{r}$  is used to prevent degeneration where all posterior poses are located on a two-dimensional manifold [1]
  - Assume Gaussian motion noises

$$\begin{pmatrix} x' \\ y' \\ \theta' \end{pmatrix} = \begin{pmatrix} x \\ y \\ \theta \end{pmatrix} + \begin{pmatrix} -\frac{\hat{v}}{\hat{\omega}} \sin \theta + \frac{\hat{v}}{\hat{\omega}} \sin(\theta + \hat{\omega} \Delta t) \\ \frac{\hat{v}}{\hat{\omega}} \cos \theta - \frac{\hat{v}}{\hat{\omega}} \cos(\theta + \hat{\omega} \Delta t) \\ \hat{\omega} \Delta t + \hat{\gamma} \Delta t \end{pmatrix}$$
$$\begin{aligned} \hat{v} &= v + \epsilon_v, & \epsilon_v &\sim \mathcal{N}(0, \alpha_1 v^2 + \alpha_2 \omega^2), \\ \hat{\omega} &= \omega + \epsilon_\omega, & \epsilon_\omega &\sim \mathcal{N}(0, \alpha_3 v^2 + \alpha_4 \omega^2), \\ \hat{\gamma} &= \epsilon_\gamma, & \epsilon_\gamma &\sim \mathcal{N}(0, \alpha_5 v^2 + \alpha_6 \omega^2). \end{aligned}$$

# Measurement model

- Landmark position  $(m_x, m_y)$
- Assume Gaussian measurement noises

$$z_k = \begin{bmatrix} \text{atan2}(m_y - y_k, m_x - x_k) - \theta_k \\ \sqrt{(m_y - y_k)^2 + (m_x - x_k)^2} \end{bmatrix} + q_k, \quad q_k \sim \mathcal{N}(0, Q_k).$$

# Jacobians

- Motion model Jacobians
  - G: Jacobian of motion model w.r.t state
  - V: Jacobian of motion model w.r.t motion noise
- Measurement model Jacobians
  - H: Jacobian of measurement model w.r.t state

$$\begin{aligned}
 G &= \begin{bmatrix} 1 & 0 & -\frac{\hat{v}}{\hat{\omega}} \cos \theta_k + \frac{\hat{v}}{\hat{\omega}} \cos(\theta_k + \hat{\omega} \Delta t) \\ 0 & 1 & -\frac{\hat{v}}{\hat{\omega}} \sin \theta_k + \frac{\hat{v}}{\hat{\omega}} \sin(\theta_k + \hat{\omega} \Delta t) \\ 0 & 0 & 1 \end{bmatrix} \\
 V &= \begin{bmatrix} \frac{-\sin \theta + \sin(\theta_k + \hat{\omega} \Delta t)}{\hat{\omega}} & \frac{v(\sin \theta - \sin(\theta_k + \hat{\omega} \Delta t))}{\hat{\omega}^2} + \frac{v \cos(\theta_k + \hat{\omega} \Delta t)}{\hat{\omega}} \\ \frac{\cos \theta - \cos(\theta_k + \hat{\omega} \Delta t)}{\hat{\omega}} & -\frac{v(\cos \theta - \cos(\theta_k + \hat{\omega} \Delta t))}{\hat{\omega}^2} + \frac{v \sin(\theta_k + \hat{\omega} \Delta t)}{\hat{\omega}} \end{bmatrix} \\
 H &= \begin{bmatrix} \frac{m_y - \bar{\mu}_{k+1,y}}{(m_x - \bar{\mu}_{k+1,x})^2 + (m_y - \bar{\mu}_{k+1,y})^2} & -\frac{m_x - \bar{\mu}_{k+1,x}}{(m_x - \bar{\mu}_{k+1,x})^2 + (m_y - \bar{\mu}_{k+1,y})^2} & -1 \\ -\frac{m_x - \bar{\mu}_{k+1,x}}{\sqrt{(m_x - \bar{\mu}_{k+1,x})^2 + (m_y - \bar{\mu}_{k+1,y})^2}} & -\frac{m_y - \bar{\mu}_{k+1,y}}{\sqrt{(m_x - \bar{\mu}_{k+1,x})^2 + (m_y - \bar{\mu}_{k+1,y})^2}} & 0 \end{bmatrix}
 \end{aligned}$$



# EKF vs. PF vs. UKF

- EKF
  - Linearization error
- UKF
  - Can handle non-linear dynamics better than EKF using sigma points
  - Slightly slower due to extra computation cost
- PF
  - Can solve nonlinear, non-Gaussian estimation problem using particles
  - Gold standard for indoor localization without GPS

END OF SLIDE