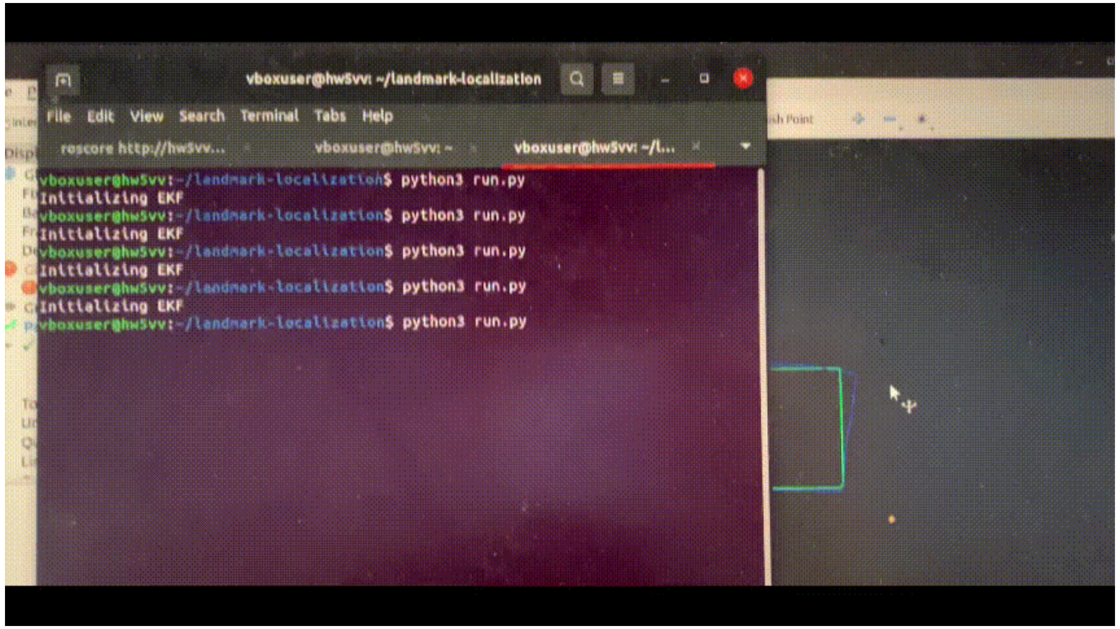
EKF, UKF, PF

Harry Chen

EKF result

- Green path: the ideal path without action noise
- Blue path: the actual path that the robot moves due to action noise
- Red arrow: the estimated 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

$$x_{k+1} = g(x_k, u_k)$$
$$P_{k+1} = GPG^T + VMV^T$$

$$K_k = P_k H_k^T (H_k P_k H_k^T + Q)^{-1}$$
$$\hat{x}_k = x_k + K_k(z_k - h(x_k))$$
$$\hat{p}_k = (1 - \mathrm{K}_k \mathrm{H}_k) P_k$$

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

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

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

$$\hat{\mathbf{x}}_{k-1}^{(i)} = \hat{\mathbf{x}}_{k-1} + \sqrt{N+\kappa}\,\mathrm{col}_i\hat{\mathbf{L}}_{k-1} \qquad i = 1...N$$

$$\hat{\mathbf{x}}_{k-1}^{(i+N)} = \hat{\mathbf{x}}_{k-1} - \sqrt{N+\kappa}\,\mathrm{col}_i\hat{\mathbf{L}}_{k-1} \qquad i = 1...N$$

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

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

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

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

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

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

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

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

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

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

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

Particle filter

- Prediction step
  - Generate samples by applying random motion $\hat{u}_k$ to particles
- Correction step
  - Update particle weights w_i 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,…,M and $r \in \left[0, \frac{1}{M}\right]$

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

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

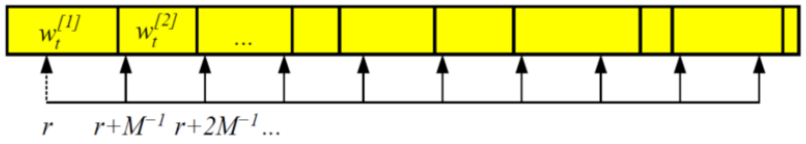
$$\hat{\gamma} = \epsilon_\gamma\,, \quad \epsilon_\gamma \sim \mathcal{N}(0, \alpha_5 v^2 + \alpha_6 \omega^2).$$

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

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

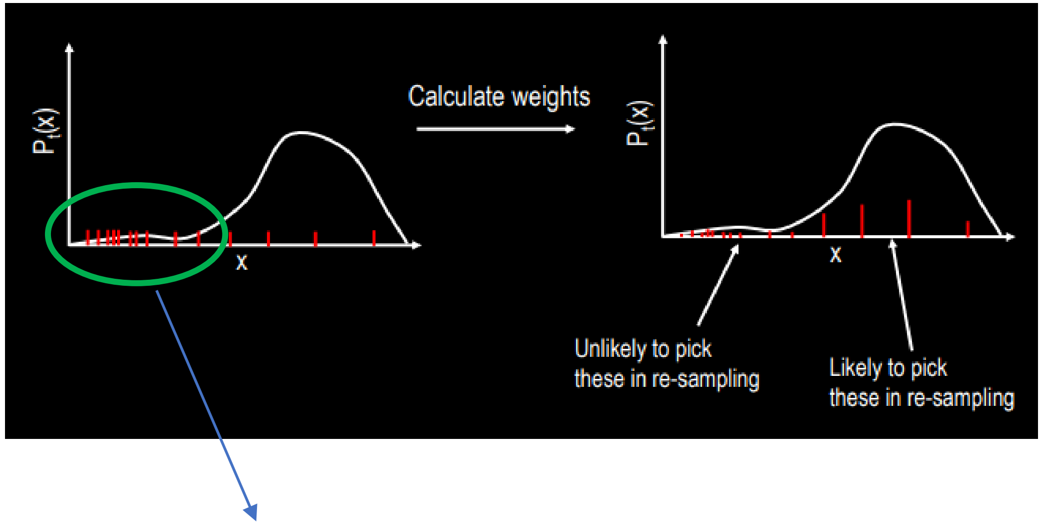
$$w_i = P(diff | N(0, Q))$$

$$w_i = \frac{w_i}{\Sigma w_i}$$



Resampling

- Weight collapse
  - As time goes by, the particle distribution get more and more skewed and lose track of true state
- Resampling
  - The particles with negligible weights are replaced by nearby particles with higher weights



Over-representing unlikely state

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

$$\begin{pmatrix} x' \\ y' \\ \theta' \end{pmatrix} = \begin{pmatrix} x \\ y \\ \theta \end{pmatrix} + \begin{pmatrix} -\frac{\hat{v}}{\hat{\omega}} \sin\theta + \frac{\hat{v}}{\hat{\omega}} \sin(\theta + \hat{\omega}\Delta t) \\ \frac{\hat{v}}{\hat{\omega}} \cos\theta - \frac{\hat{v}}{\hat{\omega}} \cos(\theta + \hat{\omega}\Delta t) \\ \hat{\omega}\Delta t + \hat{\gamma}\Delta t \end{pmatrix}$$

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

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

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

[1] Probabilistic Robotics

Measurement model

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

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

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

$$G = \begin{bmatrix} 1 & 0 & -\frac{\hat{v}}{\hat{\omega}}cos\theta_k + \frac{\hat{v}}{\hat{\omega}}cos(\theta_k + \hat{\omega}\Delta t) \\ 0 & 1 & -\frac{\hat{v}}{\hat{\omega}}sin\theta_k + \frac{\hat{v}}{\hat{\omega}}sin(\theta_k + \hat{\omega}\Delta t) \\ 0 & 0 & 1 \end{bmatrix}$$

$$V = \begin{bmatrix} \frac{-sin\theta + sin(\theta_k + \hat{\omega}\Delta t)}{\hat{\omega}} & \frac{v(sin\theta - sin(\theta_k + \hat{\omega}\Delta t))}{\hat{\omega}^2} + \frac{vcos(\theta_k + \hat{\omega}\Delta t))}{\hat{\omega}} \\ \frac{cos\theta - cos(\theta_k + \hat{\omega}\Delta t)}{\hat{\omega}} & -\frac{v(cos\theta - cos(\theta_k + \hat{\omega}\Delta t))}{\hat{\omega}^2} + \frac{vsin(\theta_k + \hat{\omega}\Delta t))}{\hat{\omega}} \end{bmatrix}$$

$$H = \begin{bmatrix} \frac{m_y - \bar{\mu}_{k+1,y}}{(m_x - \bar{\mu}_{k+1,x})^2 + (m_y - \bar{\mu}_{k+1,y})^2} & -\frac{m_x - \bar{\mu}_{k+1,x}}{(m_x - \bar{\mu}_{k+1,x})^2 + (m_y - \bar{\mu}_{k+1,y})^2} & -1 \\ -\frac{m_x - \bar{\mu}_{k+1,x}}{\sqrt{(m_x - \bar{\mu}_{k+1,x})^2 + (m_y - \bar{\mu}_{k+1,y})^2}} & -\frac{m_y - \bar{\mu}_{k+1,y}}{\sqrt{(m_x - \bar{\mu}_{k+1,x})^2 + (m_y - \bar{\mu}_{k+1,y})^2}} & 0 \end{bmatrix}$$

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