Kalman Filters

1 Tracking

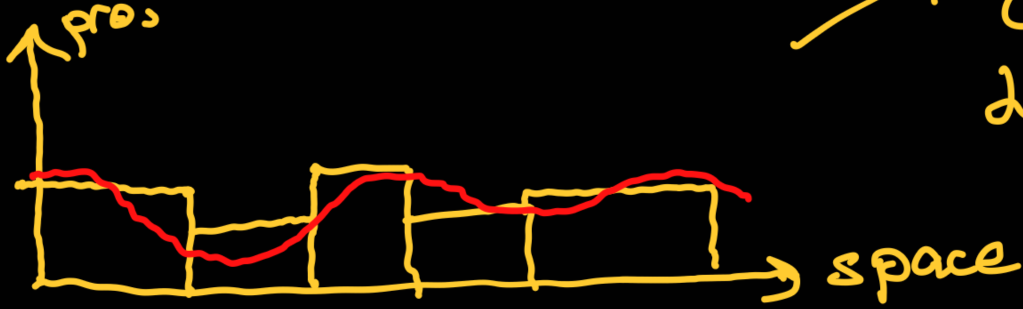
1 Kalman Filter
Continuous
Uni-modal

2 Monte Carlo Localization
Discrete
Multi-modal

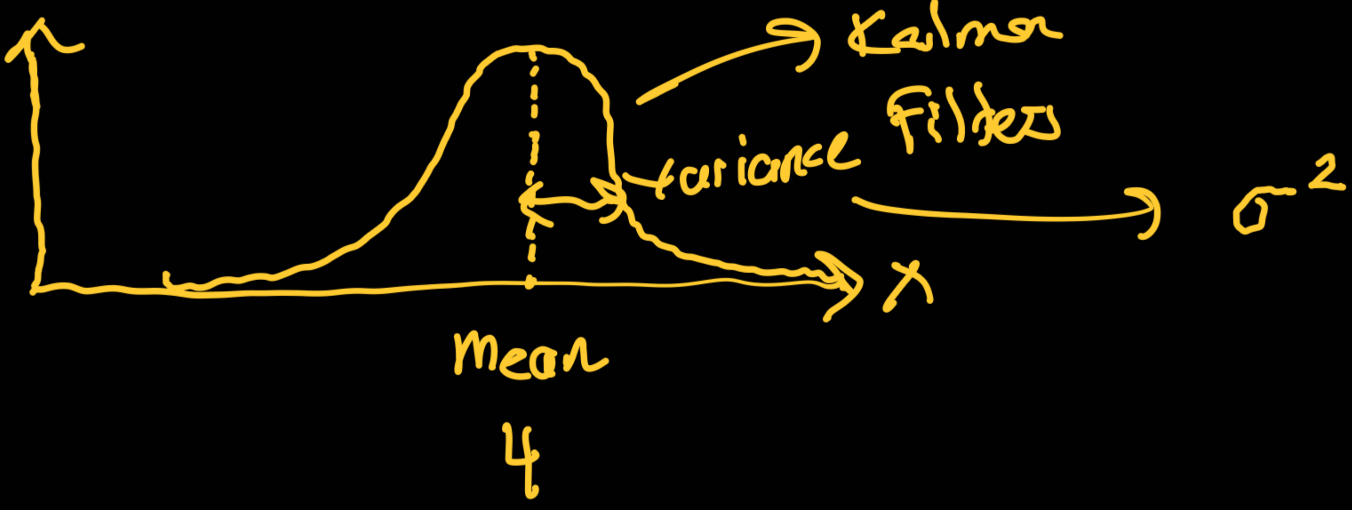
3 Particle Filters
Continuous
multi-modal

2 Gaussian Intro

0.3	0.1	0.5	0.15	0.28

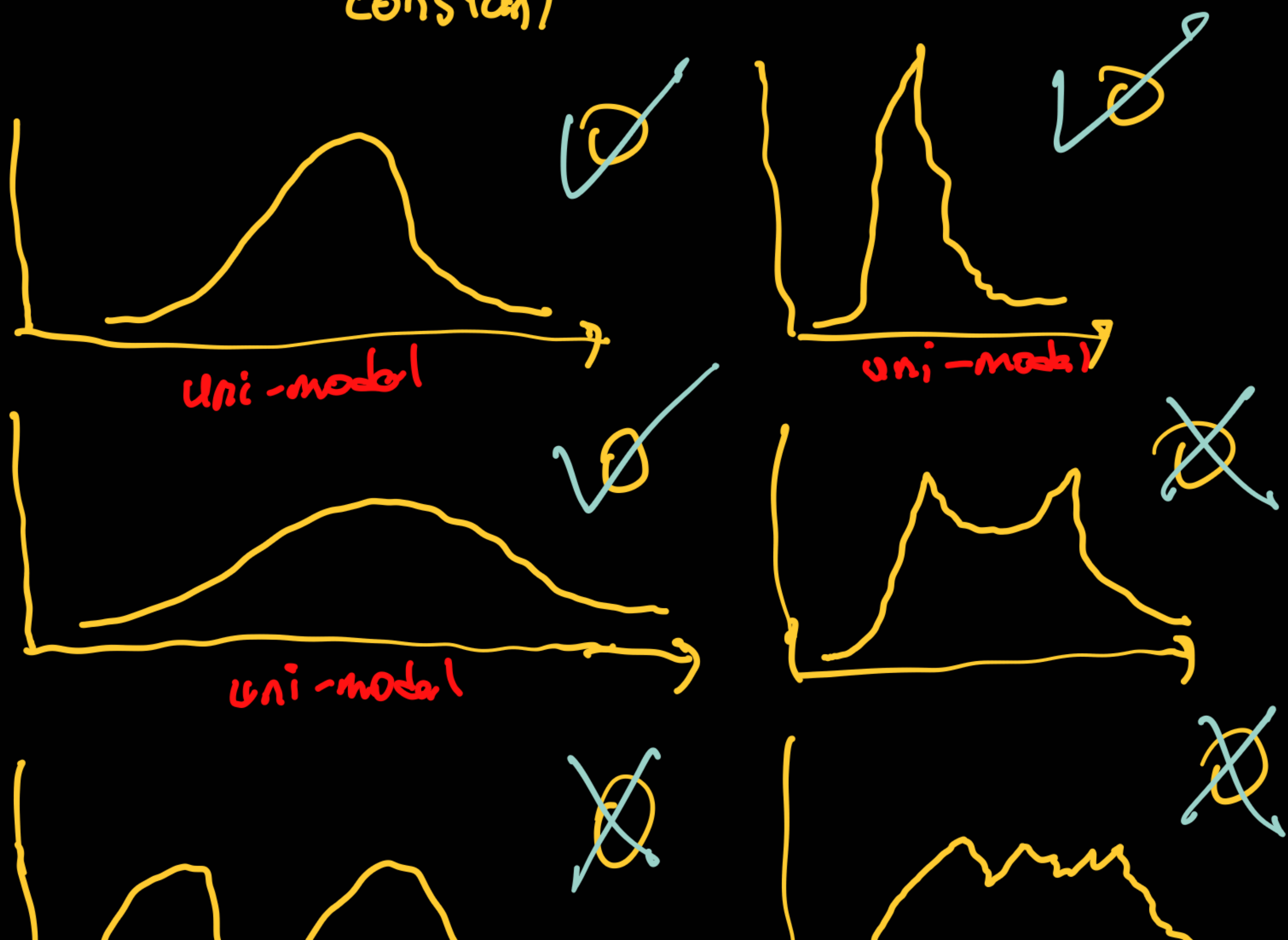


monte carlo localization



1-D Gaussian (μ, σ^2)

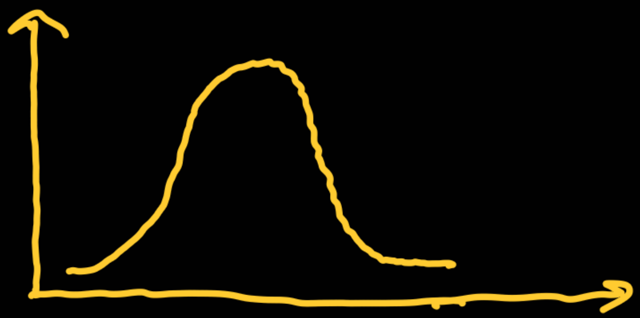
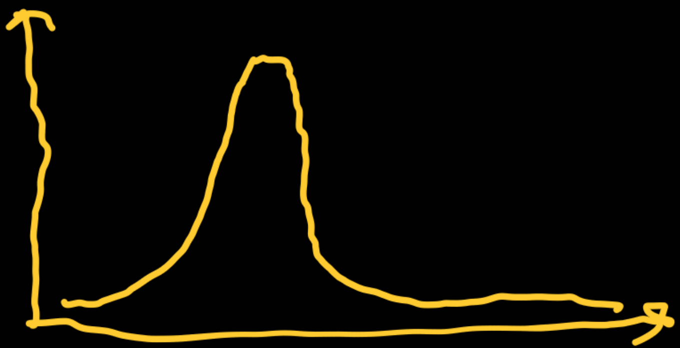
$$f(x) = \underbrace{\frac{1}{\sqrt{2\pi\sigma^2}}}_{\text{constant}} \exp^{-\frac{1}{2}\cdot\frac{(x-\mu)^2}{\sigma^2}}$$



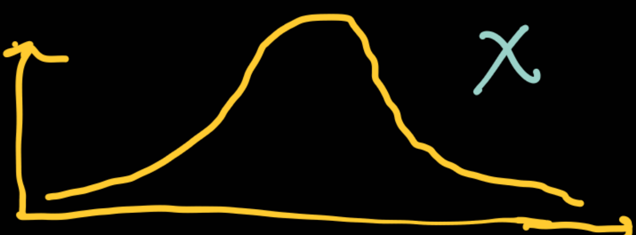
Bi-modal dist

3) Variance Comparision

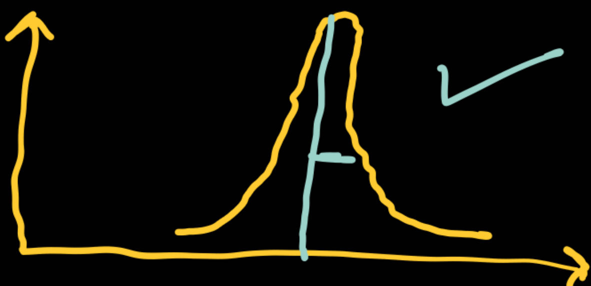
$\sigma^2 \rightarrow$ variance

	Large	medium	Small
		✓	
			✓
	✓		

4) Preferred Gaussian



X



low variance

✓

We would definitely prefer a narrow Gaussian, since that means we are confident our location.

⑤ Evaluate Gaussian

$$f(x) = 0.12$$

$\mu = 10$
$\sigma^2 = 4$
$x = 8$

⑥ Maximize Gaussian

How do I have to modify x the 8 to get the maximum return value for function f? $x = 10$ → the same value as μ

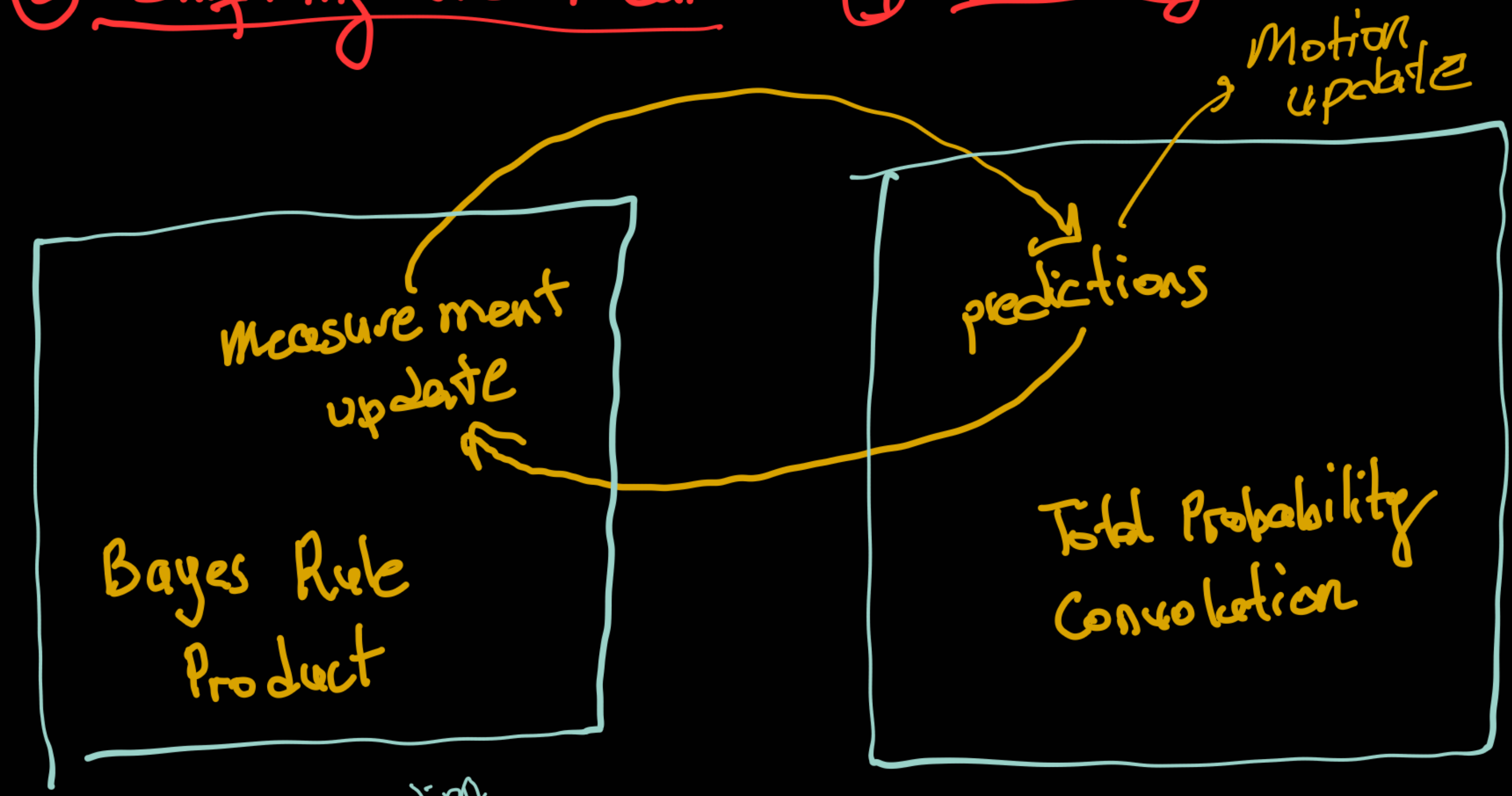
⑦ Measurement and Motion

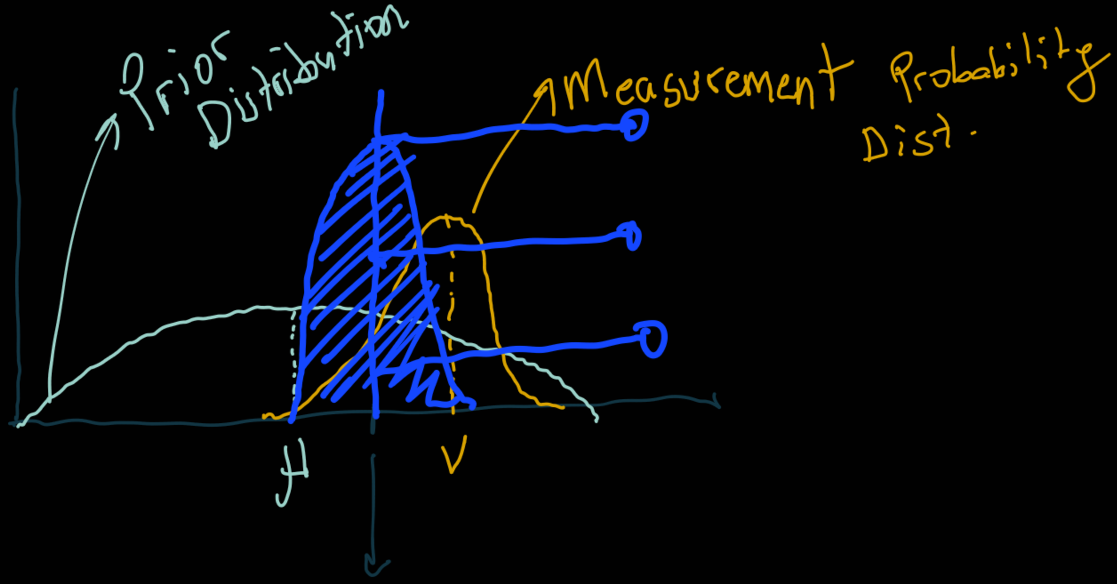
1) Measurement Update
   - Requires product
   - Uses Bayes rule

2) Motion Update

- Involves a convolution
- Uses total probability

8 Shifting the Mean - 9 Predicting Peak





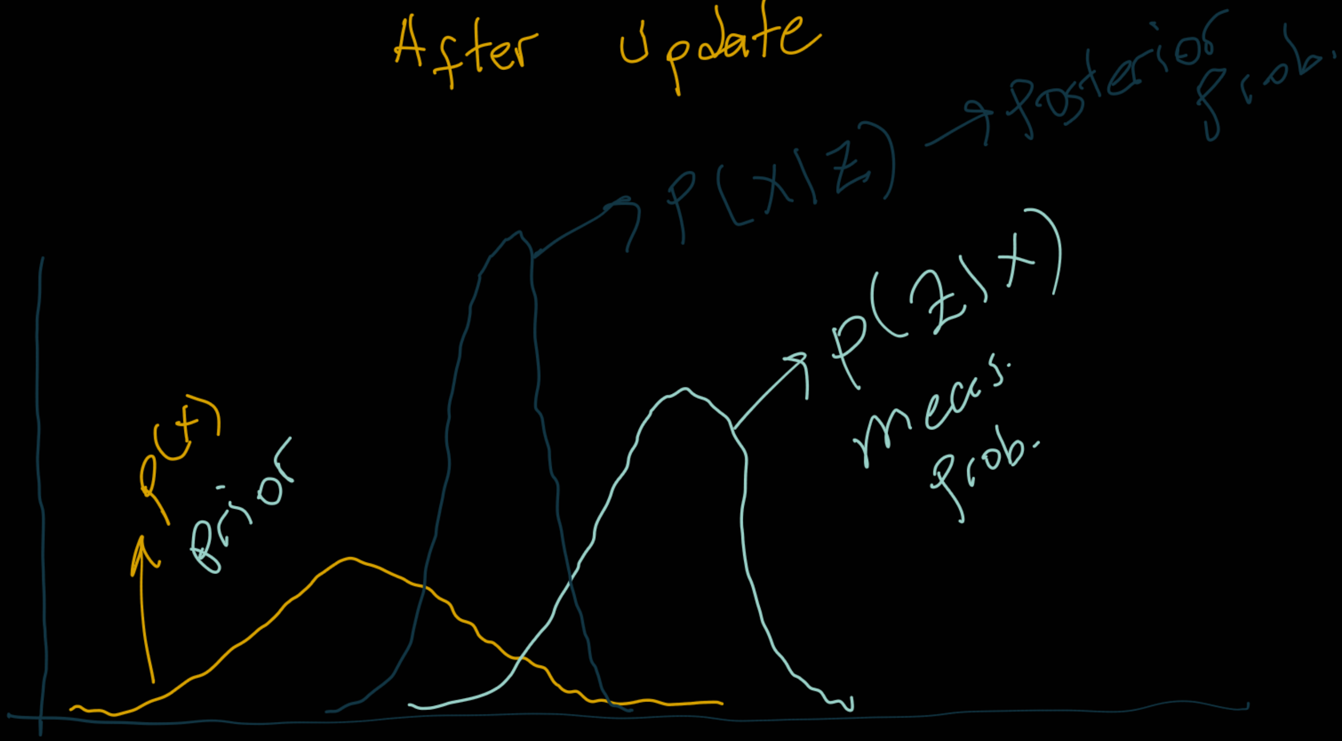
$\frac{\mu + \nu}{2} \rightarrow$ new mean

10 Parameter Update

μ, σ^2 — variance

ν, r^2 — covariance

$$\underbrace{\mu' = \frac{r^2 \cdot \mu + \sigma^2 \cdot \nu}{r^2 + \sigma^2} \qquad \sigma^2 = \frac{1}{\frac{1}{r^2} + \frac{1}{\sigma^2}}}_{\text{After Update}}$$

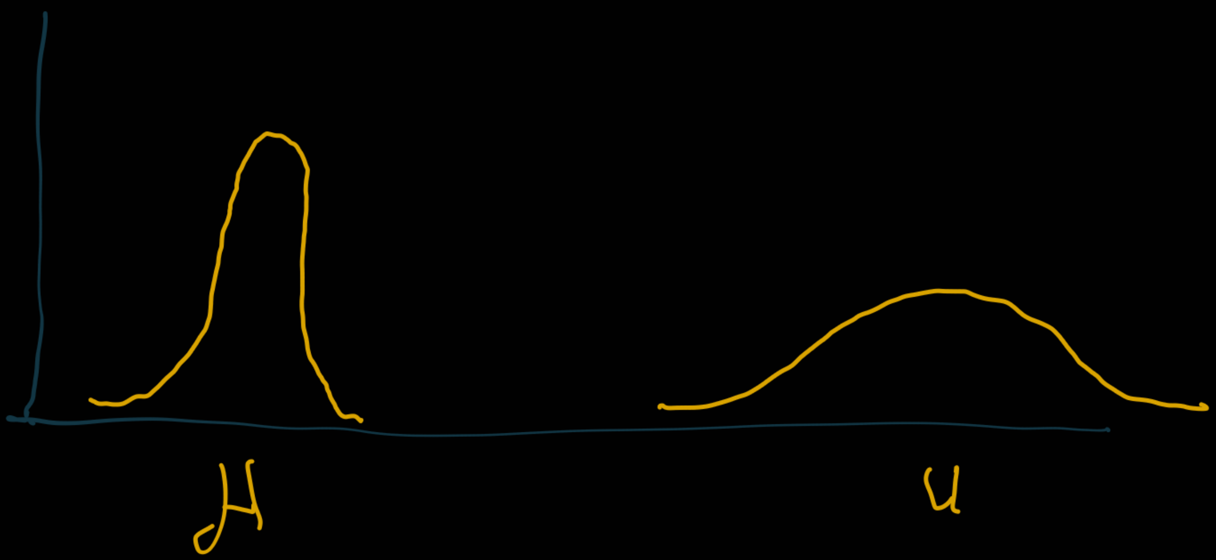


$$\mu' = \frac{1}{\sigma^2 + r^2}\left[r^2 \cdot \mu + \sigma^2 \cdot \nu\right]$$

$$\sigma^2 = \frac{1}{\frac{1}{r^2} + \frac{1}{\sigma^2}}$$

σ^2

(11) Prediction Motion Update



$$\mu' \leftarrow \mu + u$$

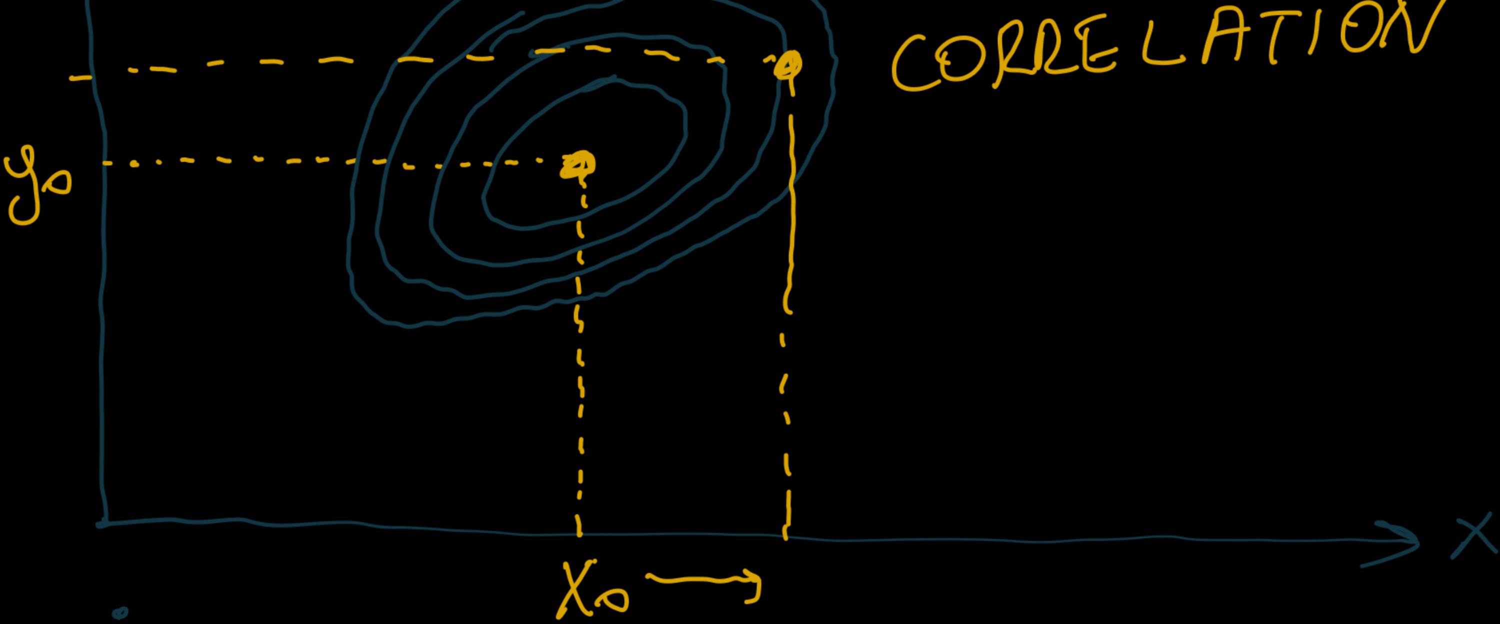
$$\sigma^{2'} \leftarrow \sigma^2 + r^2$$

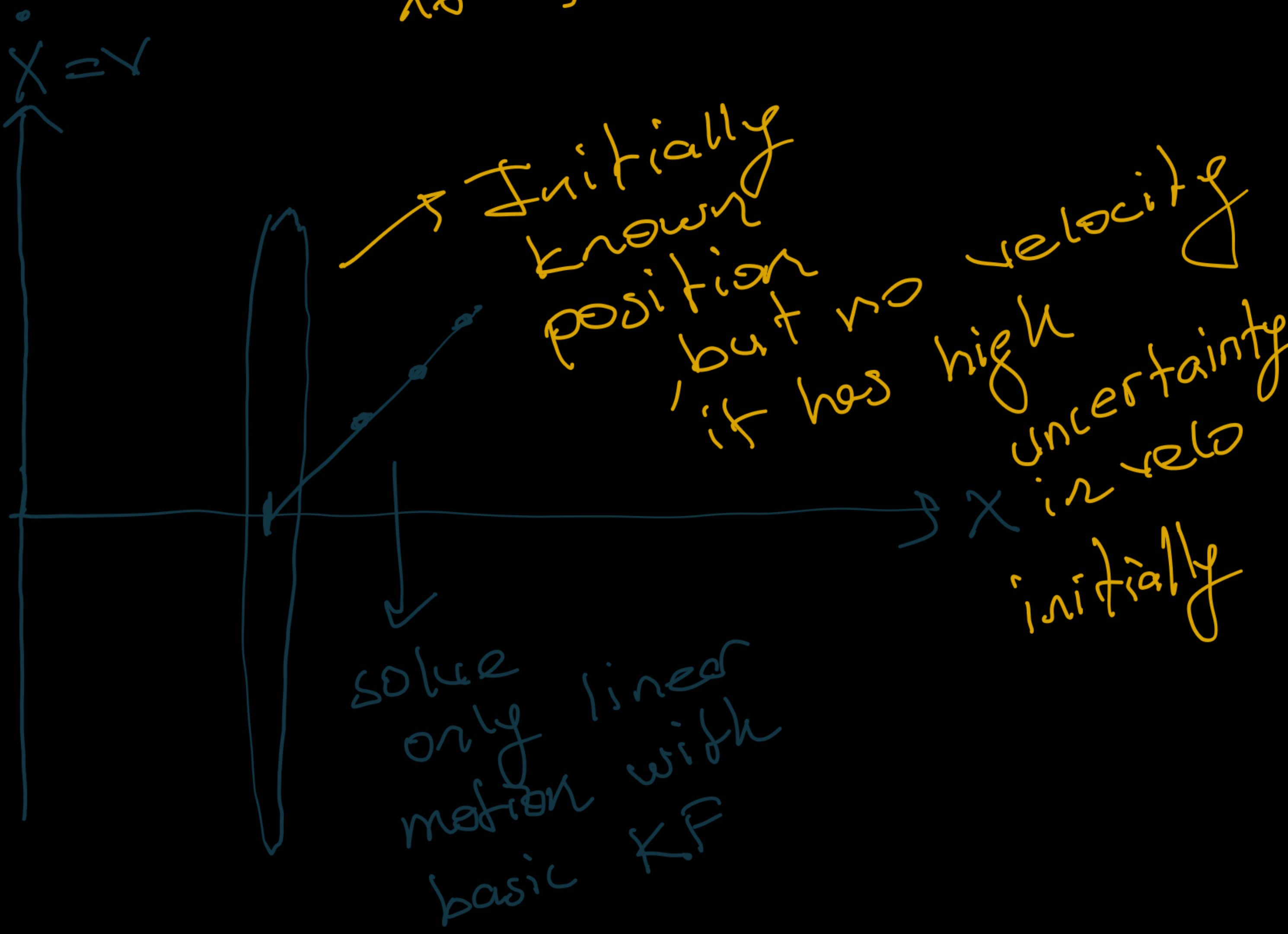
(12) Multivariate Gaussians

$$\mu = \begin{bmatrix} \mu_0 \\ \vdots \\ \mu_D \end{bmatrix} \qquad \Sigma = \begin{bmatrix} \cdots & \cdots \\ \vdots & \\ \vdots & \end{bmatrix}$$

cov. matrix

CORRELATION





Kalman Filter States → Observations X
→ Hidden $\dot{v} = v$

Learn correlati.

states

They are provided using measurement equation

$$\boxed{x' = x + \dot{x} \cdot dt}$$

---

13 KF Design

$\begin{pmatrix} x \\ \dot{x} \end{pmatrix}$ $x \rightarrow$ observation, $\dot{x} \rightarrow$ hidden state

$\underbrace{}_{\text{states}}$

$F \rightarrow$ state transition function

$$\begin{bmatrix} x' \\ \dot{x}' \end{bmatrix} = \overbrace{\begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix}}^{F} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} \rightarrow x' = x + \dot{x} \cdot dt$$

$H \rightarrow$ measurement func.

$$z \leftarrow \overbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}^{H} \begin{bmatrix} x \\ \dot{x} \end{bmatrix}$$

zero because
velocity is not
measured directly,
it's hidden

X = estimate

P = uncertainity cov.

F = state transition matrix

u = motion vector

z = measurement

H = measurement function

R = meas. noise

I = identity matrix

<u>Prediction</u>

$x' = Fx + u$

$P' = F \cdot P \cdot F^T$

<u>Measurement Update</u>

$err = z - H \cdot x$

$S = H \cdot P \cdot H^T + R$

$K = P \cdot H^T \cdot S^{-1}$

$$x' = x + K \cdot err$$

$$P' = (I - K \cdot H) \cdot P$$

Fusing LiDAR - Radar Async.

k	
$k+1$	predict update lidar
$k+2$	predict update radar
$k+3$	update lidar - radar → the same time

Linear Motion model with Constant Velocity

$$\boxed{\begin{aligned} p' &= p + v \cdot \Delta t \\ v' &= v \end{aligned}}$$

State Transition Function

$$\begin{bmatrix} p' \\ v' \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p \\ v \end{bmatrix}$$

process model

measurement model

Measurement Function

$$z = p'$$

$$z = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} p' \\ v' \end{bmatrix}$$

sensor measures only position

constant velocity model

Ex: Track $x, y, \dot{x}, \dot{y}$

hidden

$$x = \begin{bmatrix} px \\ py \\ vx \\ vy \end{bmatrix}$$ state vector

• Liner motion model with constant velocity:

$px' = px + v_x \Delta t$ + noise $\rightarrow \dfrac{a_x \Delta t^2}{2}$

$py' = py + v_y \cdot \Delta t$ + noise $\rightarrow \dfrac{a_y \Delta t^2}{2}$

$vx' = vx$ + noise $\rightarrow a_x \cdot \Delta t$

$vy' = vy + \text{noise} \rightarrow ay \cdot \Delta t$

- State Transition Eq:

$$\underbrace{\begin{bmatrix} px' \\ py' \\ vx' \\ vy' \end{bmatrix}}_{x'} = \underbrace{\begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{F} \underbrace{\begin{bmatrix} px \\ py \\ vx \\ vy \end{bmatrix}}_{x} + \underbrace{\begin{bmatrix} \text{noise} \\ \text{noise} \\ \text{noise} \\ \text{noise} \end{bmatrix}}_{\text{motion noise}}$$

- Noise due to velocity is not constant!

Note that: Radar requires the nonlinear H function → EKF

Jacobians → partial derivatives respect to states

use F_j instead of F

use H_j instead of H

Unscented Kalman Filter

- It handles the nonlinear process and measurement models.
- It doesn't use linearizing a nonlinear function.
- It uses the sigma points to approximate the probability distribution.
  - ↳ better than linearization
- So, it's not necessary to calculate Jacobian matrix.

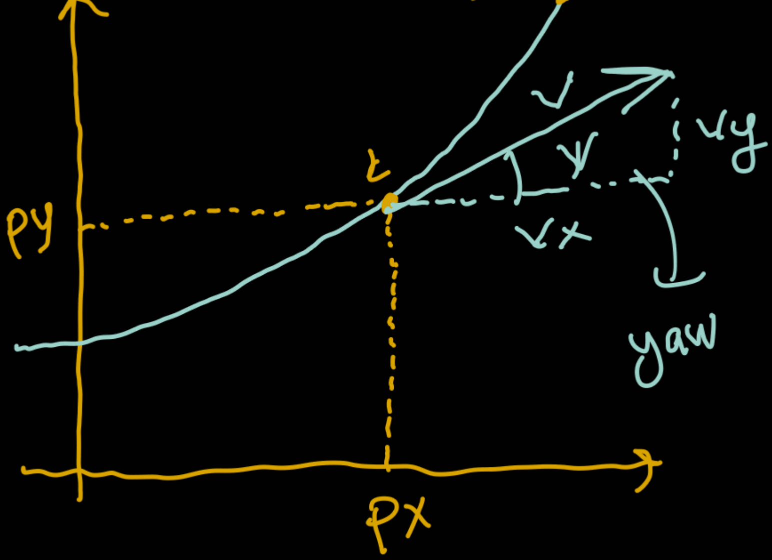
1. Process model
2. How to deals with nonlinear process model.

Motion Models:

- Constant velocity model CV
- Constant turn rate and velocity magnitude model CTRV
- Constant turn rate and acceleration CTRA
- Constant steering angle and velocity CSAV
- Constant curvature and acceleration CCA

all of them can be used both EKF and UKF

CTRV Model State Vector



$$X = \begin{array}{c} \left[\begin{array}{c} p_y \\ v \\ \psi \\ \dot{\psi} \end{array}\right] \end{array} \begin{array}{l} \rightarrow \text{hidden} \\ \rightarrow m/s \\ \rightarrow rad \\ \rightarrow rad/s \end{array}$$

yaw rate

hidden

$$X = \begin{bmatrix} p_x \\ p_y \\ v \\ \psi \\ \dot{\psi} \end{bmatrix}$$

State

$$\dot{X} = \begin{bmatrix} \dot{p}_x \\ \dot{p}_y \\ \dot{v} \\ \dot{\psi} \\ \ddot{\psi} \end{bmatrix}$$

Change Rate of State

$$X_{k+1} = f(X_k, \underbrace{\nu_k}_{\rightarrow \text{noise}})$$

Process Model

$$\dot{X} = g(X)$$

Differential Equation

$$\dot{p}_x = v_x = \cos(\psi) \cdot v$$

$\dot{p}_y = v_y = \sin(\psi) \cdot v$

$$\dot{x} = \begin{bmatrix} \dot{p}_x \\ \dot{p}_y \\ \dot{v} \\ \dot{\psi} \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} v \cdot \cos(\psi) \\ v \cdot \sin(\psi) \\ 0 \\ \dot{\psi} \\ 0 \end{bmatrix}$$

constant velocity mag.

constant yaw rate

Change Rate of State

Integral

$$x_{k+1} = x_k + \int_{t_k}^{t_{k+1}} \begin{bmatrix} \dot{p}_x(t) \\ \dot{p}_y(t) \\ \dot{v}(t) \\ \dot{\psi}(t) \\ \ddot{\psi}(t) \end{bmatrix} dt$$

Time Difference

$$\Delta t = t_{k+1} - t_k$$

$$x_{k+1} = x_k + \begin{bmatrix} \int_{t_k}^{t_{k+1}} v(t) \cdot \cos(\psi(t)) \cdot dt \\ \int_{t_k}^{t_{k+1}} v(t) \cdot \sin(\psi(t)) \cdot dt \\ ? \\ ? \\ ? \end{bmatrix}$$

$$x_{k+1} = x_k + \begin{bmatrix} v_k \int_{t_k}^{t_{k+1}} \cos(\psi_k + \dot{\psi}_k \cdot (t - t_k)) \, dt \\ v_k \int_{t_k}^{t_{k+1}} \sin(\psi_k + \dot{\psi}_k \cdot (t - t_k)) \, dt \\ 0 \\ \dot{\psi}_k \cdot \Delta t \\ 0 \end{bmatrix}$$

Solves: $\frac{v_k}{\dot{\psi}_k} \left(\sin(\psi_k + \dot{\psi}_k \cdot \Delta t) - \sin(\psi_k) \right)$

ψ_k

Solusi $\frac{v_k}{\dot{\psi}_k}\left(-\cos(\psi_k + \dot{\psi}_k \Delta t) + \cos(\psi_k)\right)$

CTRV Zero Yaw Rate

Q1: How should we calculate the change in the x position over time when the yaw rate is 0?

$$v_k \cdot \cos(\psi_k)\,\Delta t$$

Q2: How should we calculate the change in the y position over time when $\dot{\psi}_k = 0$?

$$v_k \cdot \sin(\psi_k)\,\Delta t$$

CTRV Process Noise Vector

$$e = \Delta t \cdot v_{\ddot{\psi},k}$$

Generating Sigma Points

$2n_x + 1 \approx 11 \rightarrow$ sigma count

n_x → state dim

UKF Roadmap

Prediction → Generate sigma points
Predict sigma points
Predict mean and cov.

Update → Predict measurement
Update state