NYU

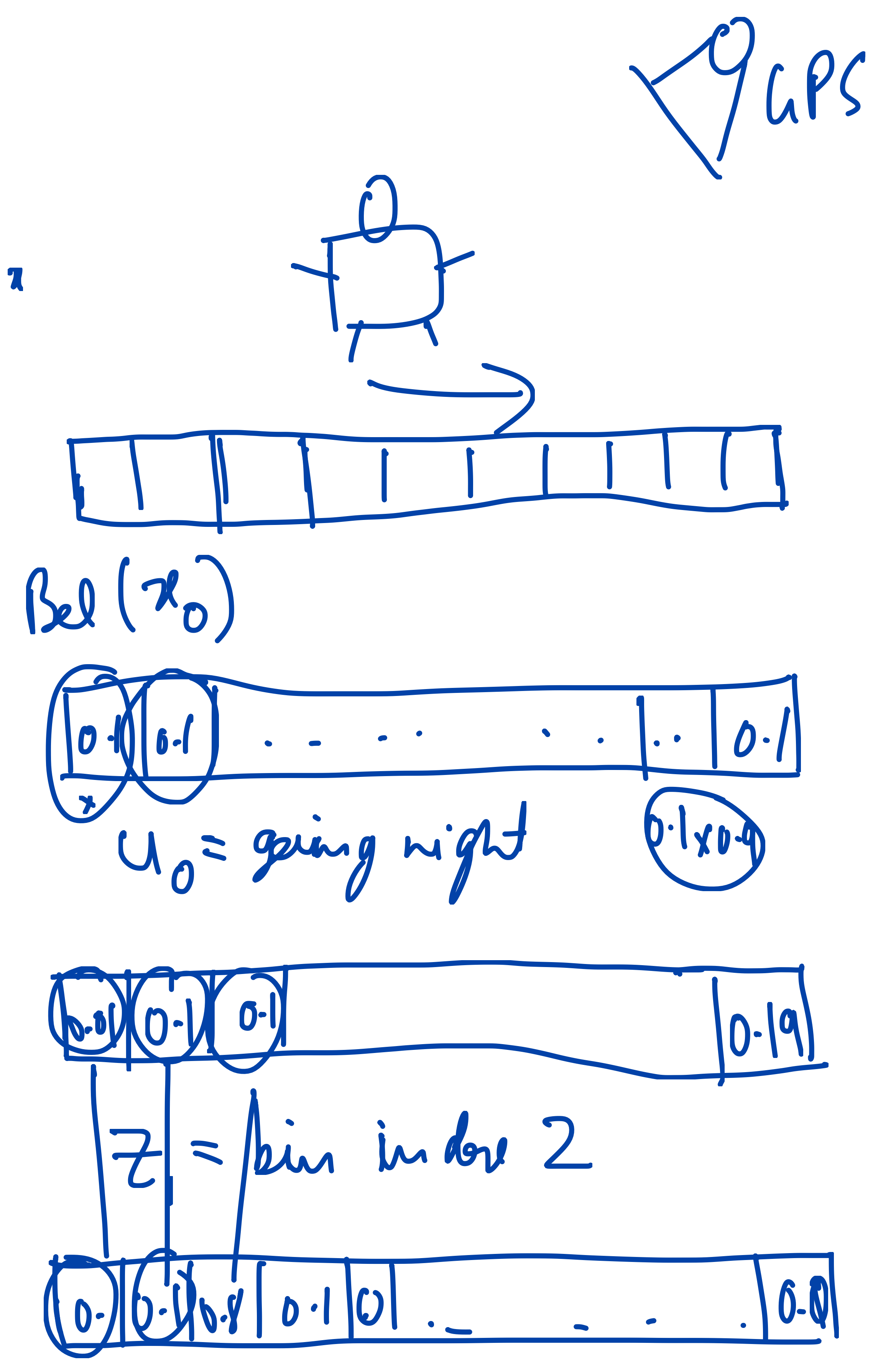
Introduction to Robot Intelligence
[Spring 2023]

Probabilistic Robotics II

April 13, 2023

Lerrel Pinto

Bayes Filter



Observation model

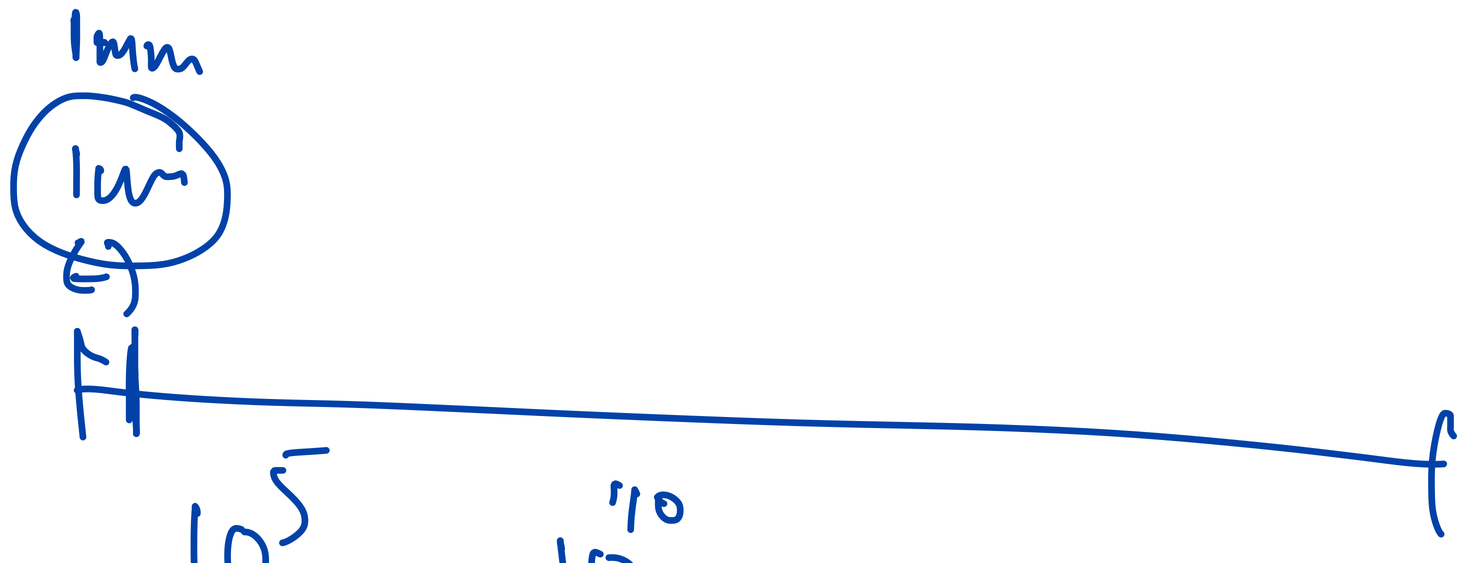
$$p(z_{t+1} \mid x_{t+1})$$

Dynamics model

$$p(x_{t+1} \mid x_t, u_t)$$

Prior model

$$Bel(x_0)$$



100 m

10,000 bin

10^8

10^{12}

1m×1m

100m

100m

Prediction

$$Bel(x_{t+1})$$

$$p(x_{t+1} \mid z_0, \ldots z_{t+1}; u_0 \ldots u_t) = \eta \, p(z_{t+1} \mid x_{t+1}) \sum_{x_t \sim X} p(x_{t+1} \mid x_t, u_t) \cdot Bel(x_t)$$

http://ais.informatik.uni-freiburg.de/teaching/ss14/robotics/slides/22-summary.pdf

From Discrete to Continuous Variables

Discrete

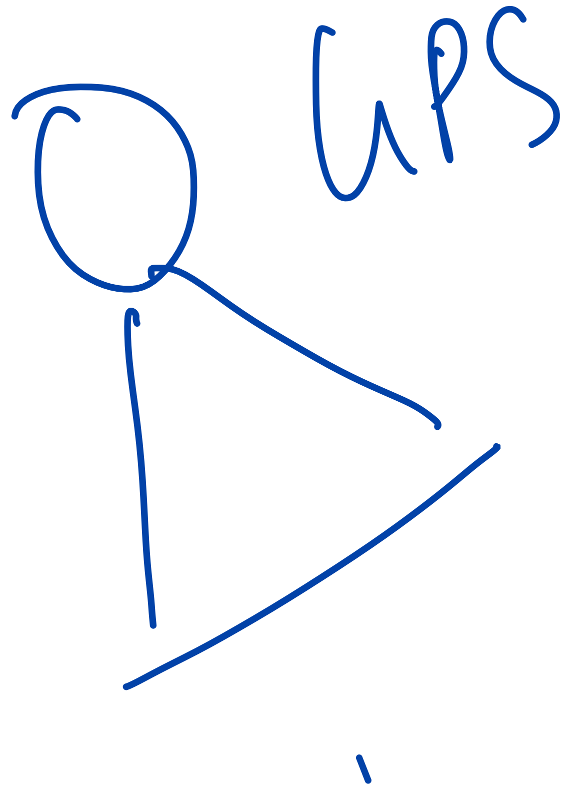
$$Bel(x_{t+1}) = \eta \; p(z_{t+1} \mid x_{t+1}) \cdot \sum_{x_t \sim X} p(x_{t+1} \mid x_t, u_t) \cdot Bel(x_t)$$

Continuous

$$Bel(x_{t+1}) = \eta \; p(z_{t+1} \mid x_{t+1}) \int_{x_t} p(x_{t+1} \mid x_t, u_t) \cdot Bel(x_t) \cdot dx_t$$

http://www.cs.cmu.edu/~16385/s17/Slides/16.3_Kalman_Filtering.pdf

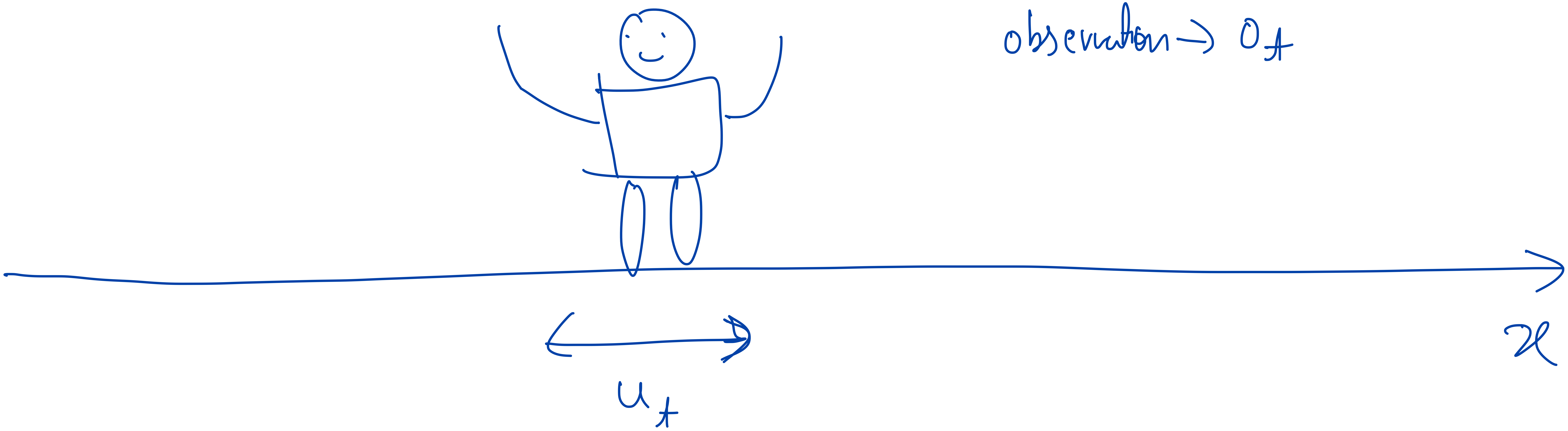
A simple 1D example



state $\rightarrow x_t$

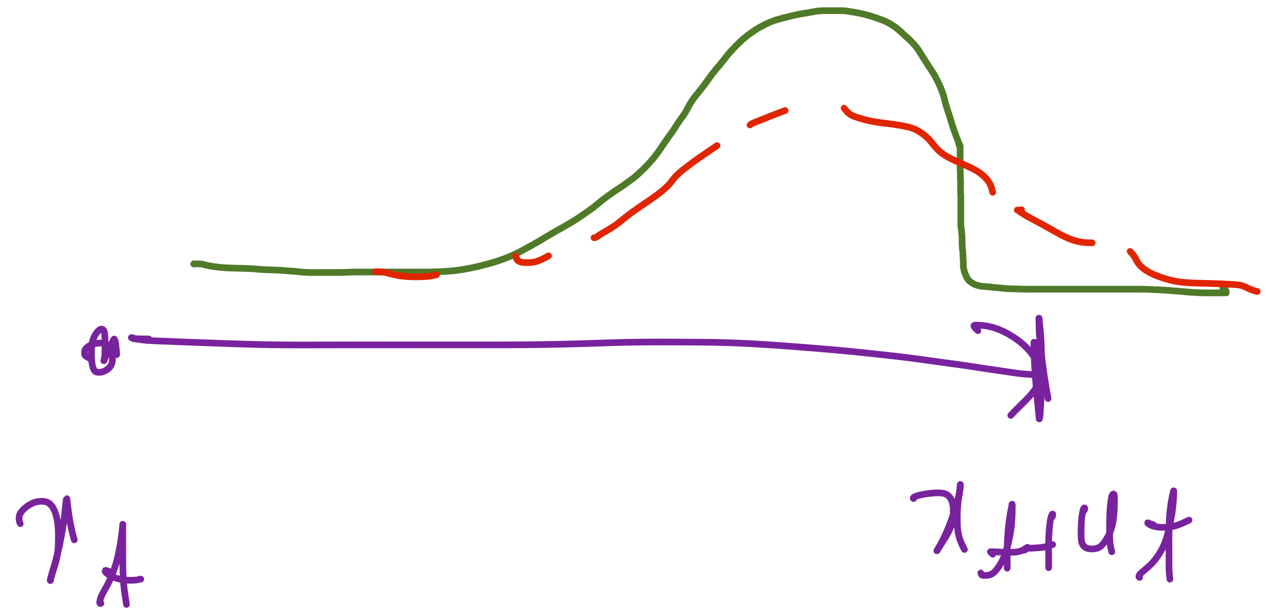
action $\rightarrow u_t$

observation $\rightarrow o_t$



$\rightarrow$ commanded velocity.

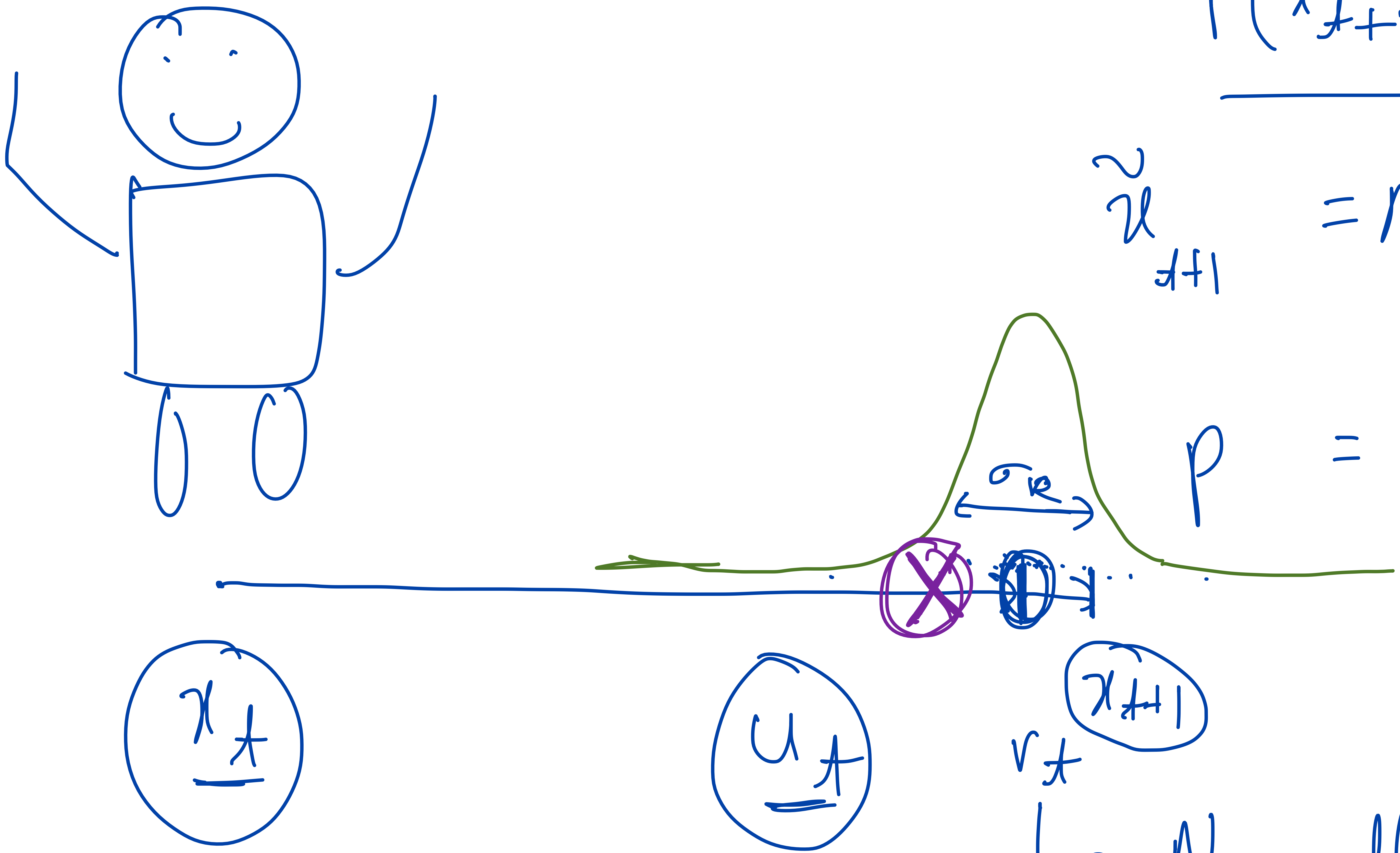
http://www.cs.cmu.edu/~16385/s17/Slides/16.3_Kalman_Filtering.pdf



Motion Model

$$p(x_{t+1} \mid \underline{x_t}, \underline{u_t})$$

$$\tilde{x}_{t+1} = N(x_t + u_t, \sigma_R)$$

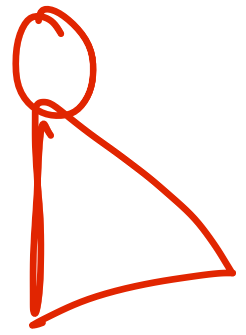


$$p = \frac{1}{\sigma_R\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{\tilde{x}_{t+1}-x_t-u_t}{\sigma_R}\right)^2}$$

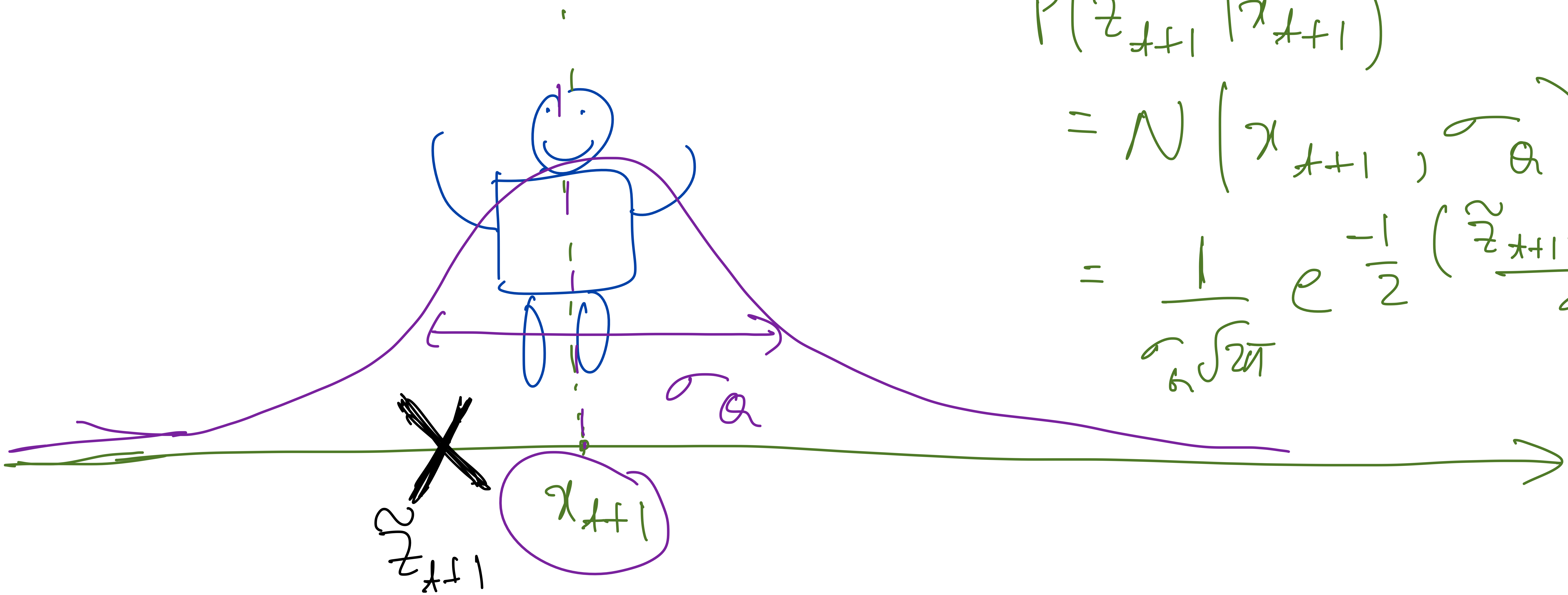
r_t → Normally distributed

$N(0, \sigma_R)$

http://www.cs.cmu.edu/~16385/s17/Slides/16.3_Kalman_Filtering.pdf

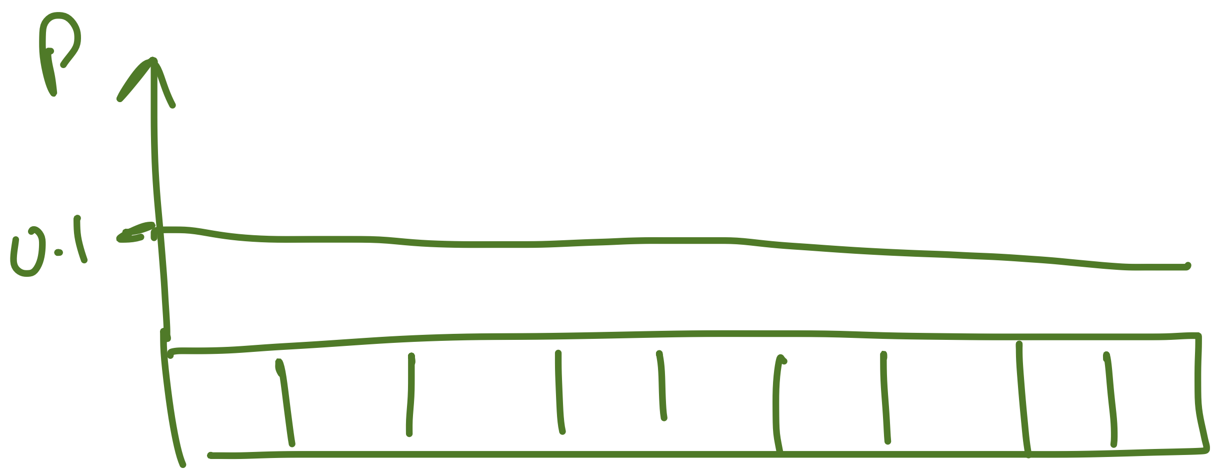


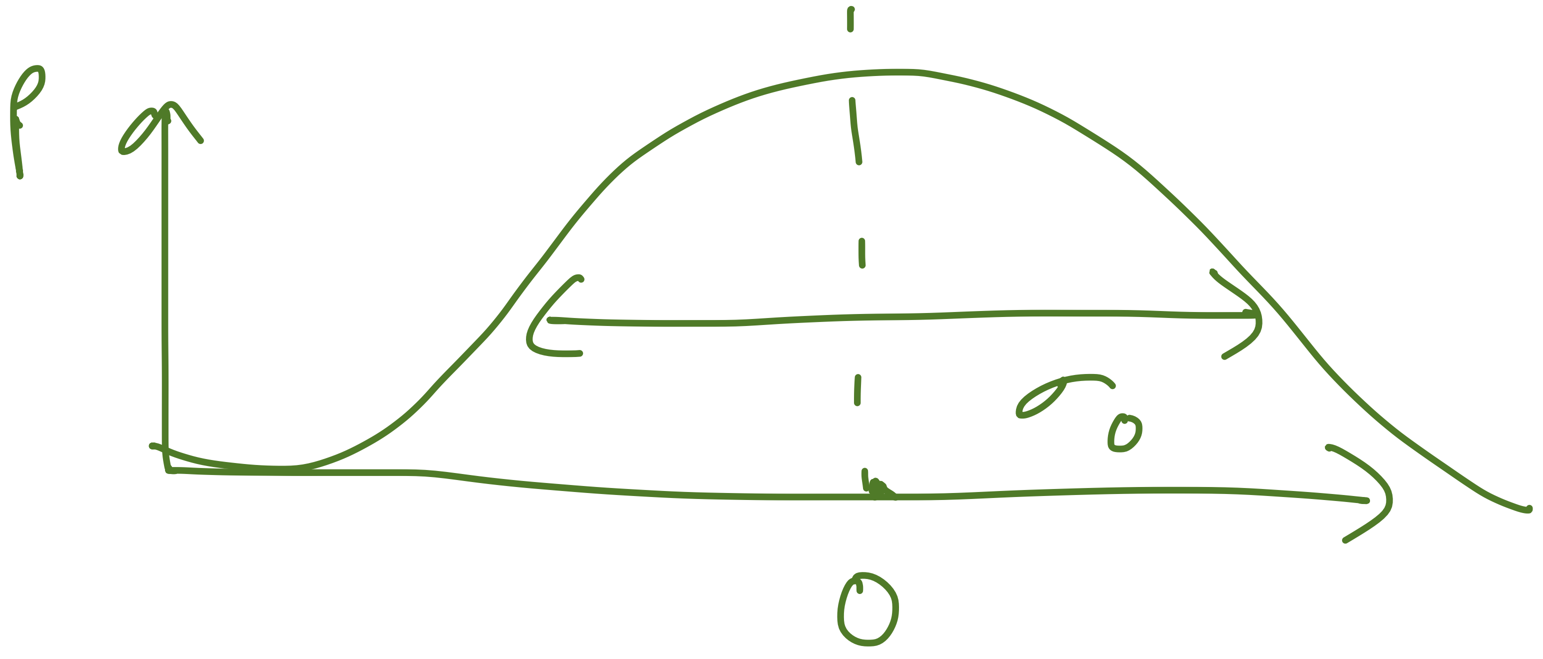
Observation Model



$$p(z_{t+1} \mid x_{t+1})$$
$$= \mathcal{N}(x_{t+1}, \sigma_a)$$
$$= \frac{1}{\sigma_a\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{\tilde{z}_{t+1} - x_{t+1}}{\sigma_a}\right)^2}$$

http://www.cs.cmu.edu/~16385/s17/Slides/16.3_Kalman_Filtering.pdf





Prior Model

$x_0 \sim N(0, \sigma_0) \longrightarrow$ Prior model

$x_{t+1} \sim N(x_t + u_t, \sigma_R) \longrightarrow$ Motion model

$z_{t+1} \sim N(x_{t+1}, \sigma_Q) \longrightarrow$ Observation model

$$\underline{p(x_{t+1} \mid x_0, u_{0:t}, z_{0:t+1})} \; ???$$

$$\downarrow$$

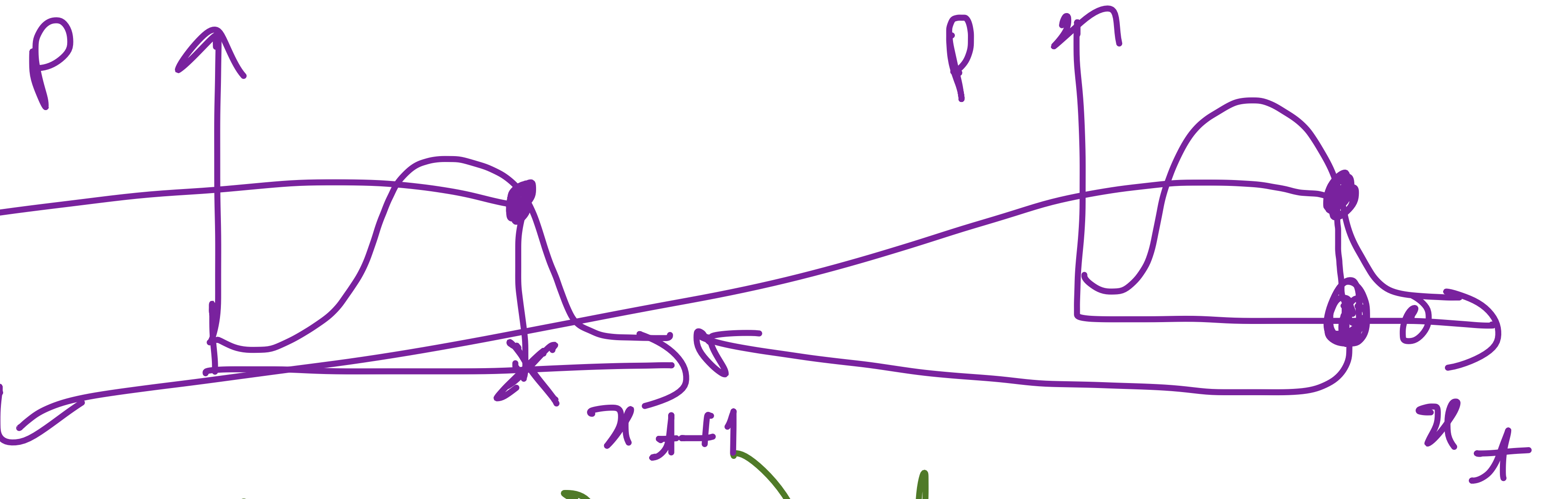
$$Bel(x_{t+1})$$

http://www.cs.cmu.edu/~16385/s17/Slides/16.3_Kalman_Filtering.pdf

Continuous

$Bel(x_{t+1}) = \eta\ p(z_{t+1}|x_{t+1}) \int_{x_t} p(x_{t+1}|x_t, u_t) \cdot Bel(x_t) \cdot dx_t$

Inference



Bayes Filter rule

$$Bel(x_{t+1}) = \eta\ \underbrace{p(z_{t+1}|x_{t+1})}_{\text{Observation model}} \int_{x_t} \underbrace{\underbrace{p(x_{t+1}|x_t, u_t)}_{\text{Motion model}} \cdot \underbrace{p(x_t|x_0, u_{0:t-1}, z_{0:t})}_{Bel(x_t)}\, dx_t}$$

$$x_0 \longrightarrow u_0 \longrightarrow z_1 \quad \cdots\cdots \quad u_t \longrightarrow z_{t+1}$$

$$z_1 \xrightarrow{Bel} x_1 \qquad z_{t+1} \xrightarrow{Bel} x_{t+1}$$

http://www.cs.cmu.edu/~16385/s17/Slides/16.3_Kalman_Filtering.pdf

Inference with 'Kalman' Gain

$$Bel(x_1) = \eta \underbrace{p(z_1 \mid x_1)}_{Obs} \int_{x_0} \underbrace{p(x_1 \mid x_0, u_0)}_{Motion} \cdot \underbrace{p(x_0)}_{Bel(x_0)} \cdot dx_0$$

$$N(x_1, \sigma_Q) \qquad N(x_0 + u_0, \sigma_R) \qquad N(0, \sigma_0)$$

$$N(\hat{x}_1, \hat{\sigma}_1)$$

$$N(\hat{x}_1, \hat{\sigma}_1)$$

$$\hat{x}_1 = u_0$$

$$\hat{\sigma}_1^2 = \sigma_0^2 + \sigma_R^2$$

http://www.cs.cmu.edu/~16385/s17/Slides/16.3_Kalman_Filtering.pdf

Inference with 'Kalman' Gain

$$\text{KalmanFilter}(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t)$$

motion control

prediction mean $\bar{\mu}_t = A_t \mu_{t-1} + B u_t$ 'old' mean

'old' covariance

prediction covariance $\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^\top + R$ Gaussian noise

Prediction

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

Gain

observation model

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

update covariance $\Sigma_t = (I - K_t C_t)\bar{\Sigma}_t$

Update

http://www.cs.cmu.edu/~16385/s17/Slides/16.3_Kalman_Filtering.pdf

Inference with 'Kalman' Gain

Simple Multi-dimensional Implementation
(also 5 lines of code!)

```
[x P] = KF(x,P,z)

  x = A*x;
  P = A*P*A' + Q;

  K = P*C'/(C*P*C' + R);

  x   = x + K*(z - C*x);
  P   = (eye(size(K,1))-K*C)*P;
```

http://www.cs.cmu.edu/~16385/s17/Slides/16.3_Kalman_Filtering.pdf

What do you do with non-linear dynamics?

http://www.cs.cmu.edu/~16385/s17/Slides/16.3_Kalman_Filtering.pdf

More principled approach: Partile Filter

More principled approach: Partile Filter

https://classes.cs.uchicago.edu/archive/2021/spring/20600-1/particle_filter_project.html