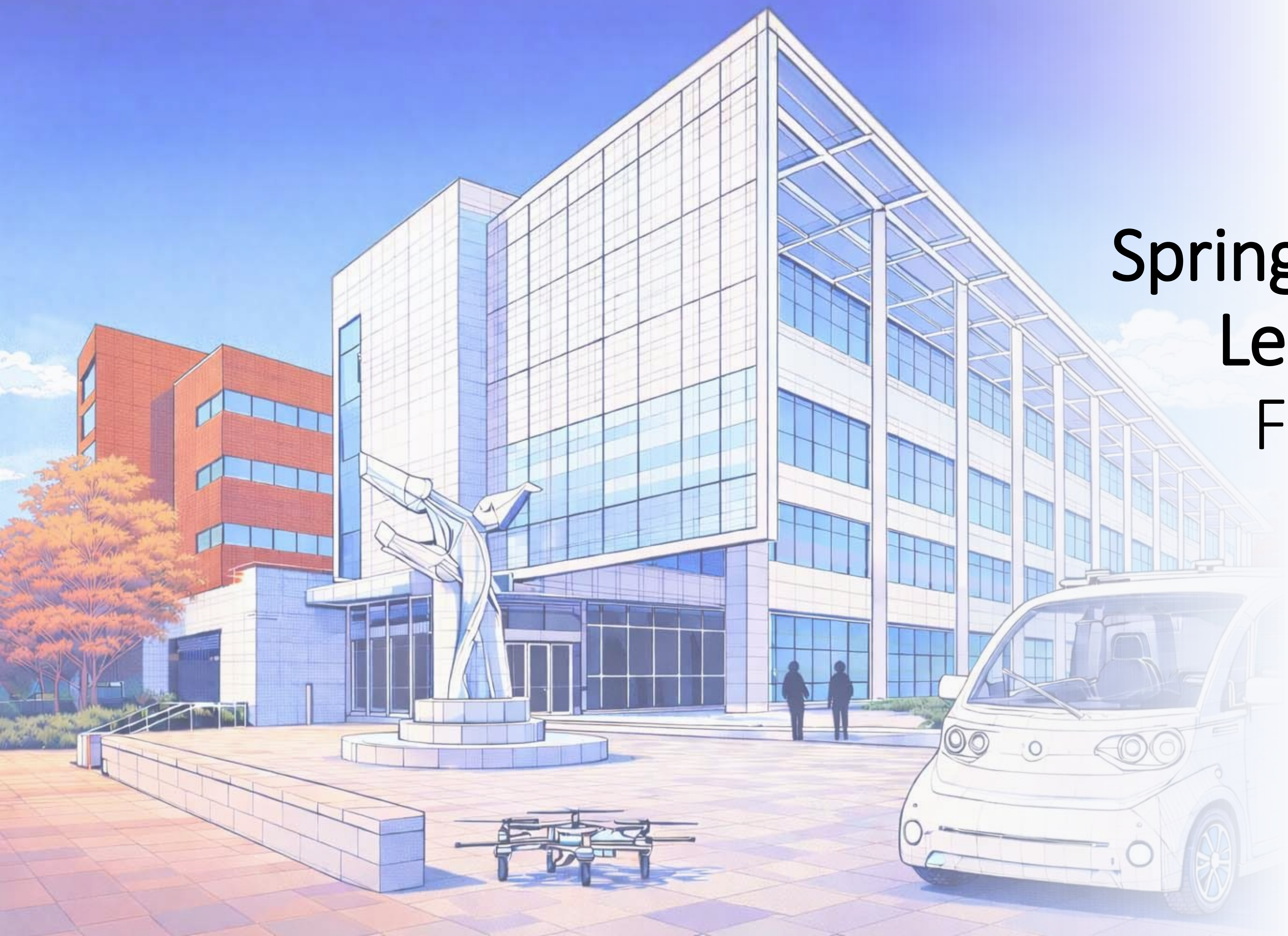


Spring 26 ECE484 Lecture 11

Filtering

Sayan Mitra

Filtering problem

Keep track of unknown, changing state (position, attitude, etc.) using imperfect models (ODEs, automata, Markov chains) and noisy sensors (camera, lidar, etc.)

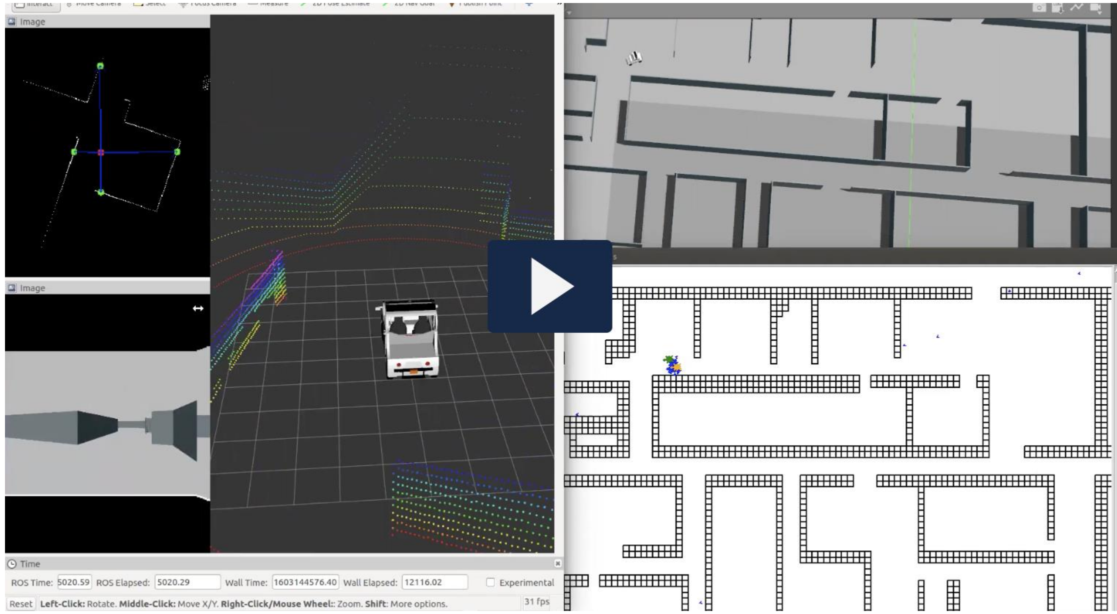
- At each time step:
  - The state moves according to known physics/control plus disturbances
  - sensors/perception give partial, noisy, sometimes missing measurements
- Goal: combine model + measurements + uncertainty to produce the best current estimate (and confidence) of the state in real time
- Output: an estimate of “what’s happening now” (and optionally a short history), robust to noise, occlusions, and outliers

State estimation and localization problem (MP3)

- Controller needs current state

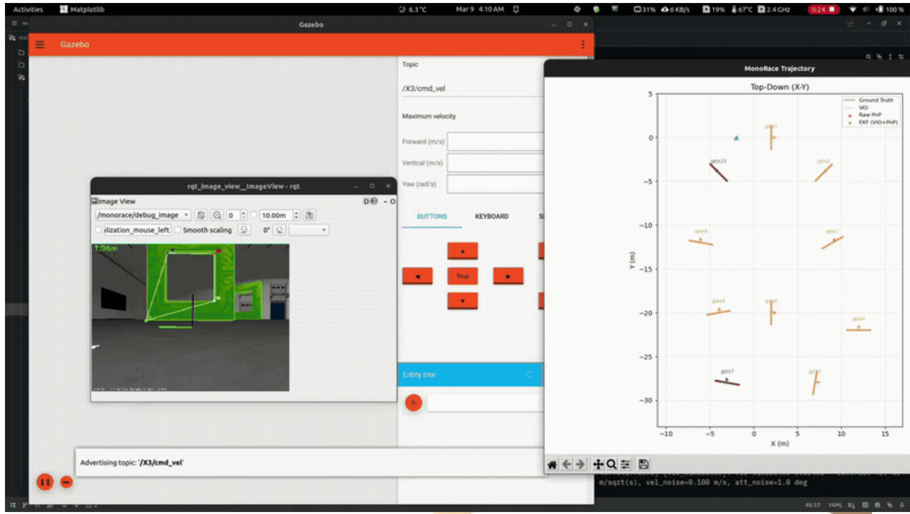
$$x_{t+1} = f(x_t, u_t);\ u_t = g(x_t)$$

- Typically, the state x_t is not available. We have observables $z_t = h(x_t)$ that are available.
  - Images, lidar scans, GPS, IMU
- Compute a state estimate $\hat{x}_t$ from observations z_t
- Ideally, $\hat{x}_t \approx x_t$ and we use $u_t = g(\hat{x}_t)$
- Localization special case of state estimation where $\hat{x}_t$ is the pose in a *given map* of the environment



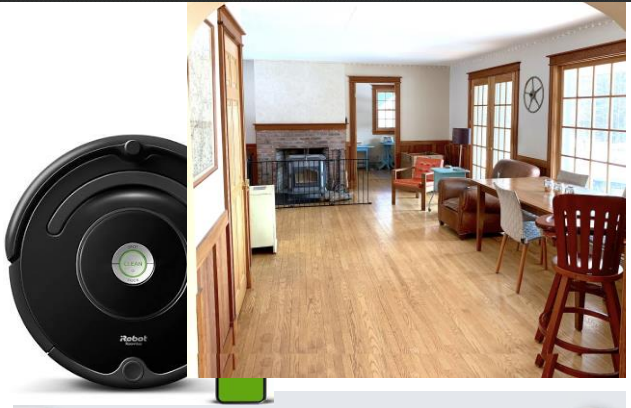
Applications

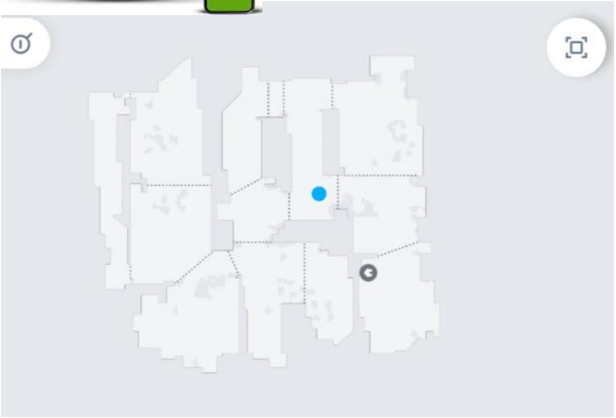
Drone racing systems use measurements from camera and IMU to estimate the pose of the drone



iRobot Roomba uses filtering SLAM algorithm to create maps for cleaning areas

SLAM: Simultaneous Localization and Mapping





Underground, underwater, and space robots, in GPS-denied environments

Outline of state estimation module

- Probabilistic models
- Discrete Bayes Filter
- Kalman Filter

Setup: State evolution and measurement models

Familiar Deterministic model:

System evolution: $x_{t+1} = f(x_t, u_t)$

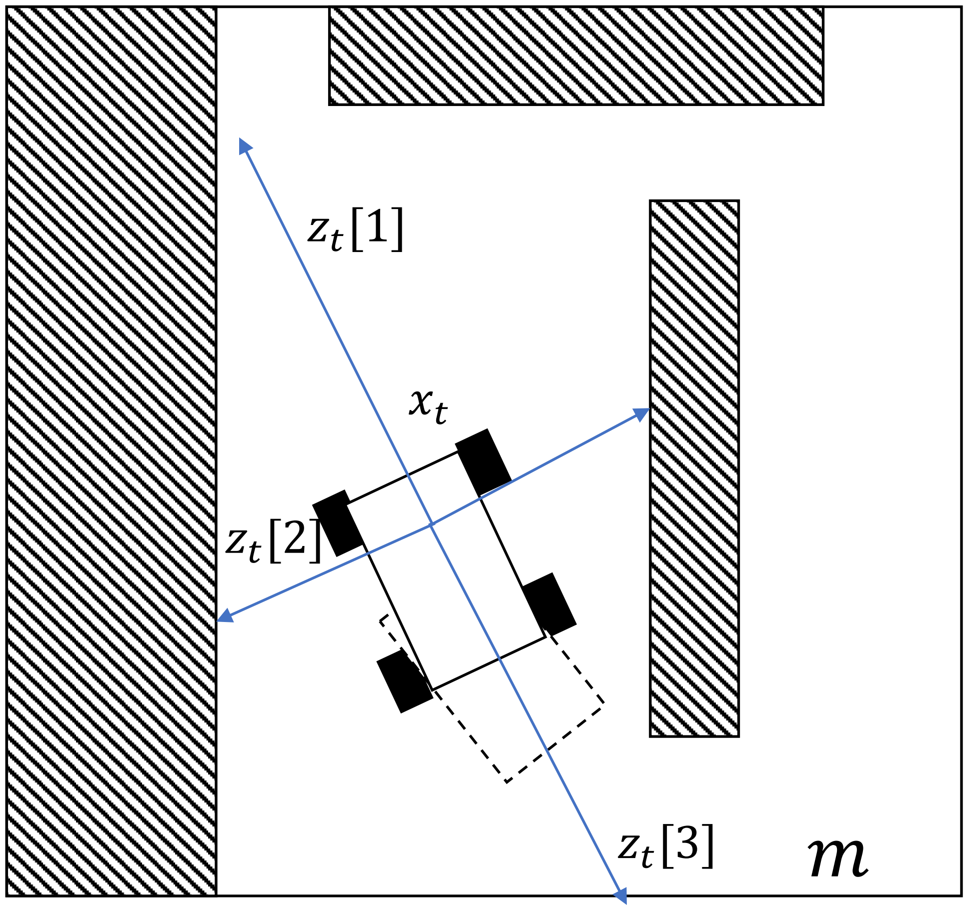
- x_t: unknown state of the system at time t
- u_t: known control input at time t, $u_t = g(\hat{x}_t)$
- f: known dynamic function, possibly stochastic

Measurement or observation: $z_t = h(x_t, m)$

- z_t: known measurement of state x_t at time t
- m: unknown underlying map
- h: known measurement function

Problem: Given the sequence of measurements $z_1, z_2, \ldots z_{t-1}$ and control inputs $u_1, u_2, \ldots u_{t-1}$

We will use probabilistic models going forward



This is not exactly the measurement model of MP3

Setup, notations

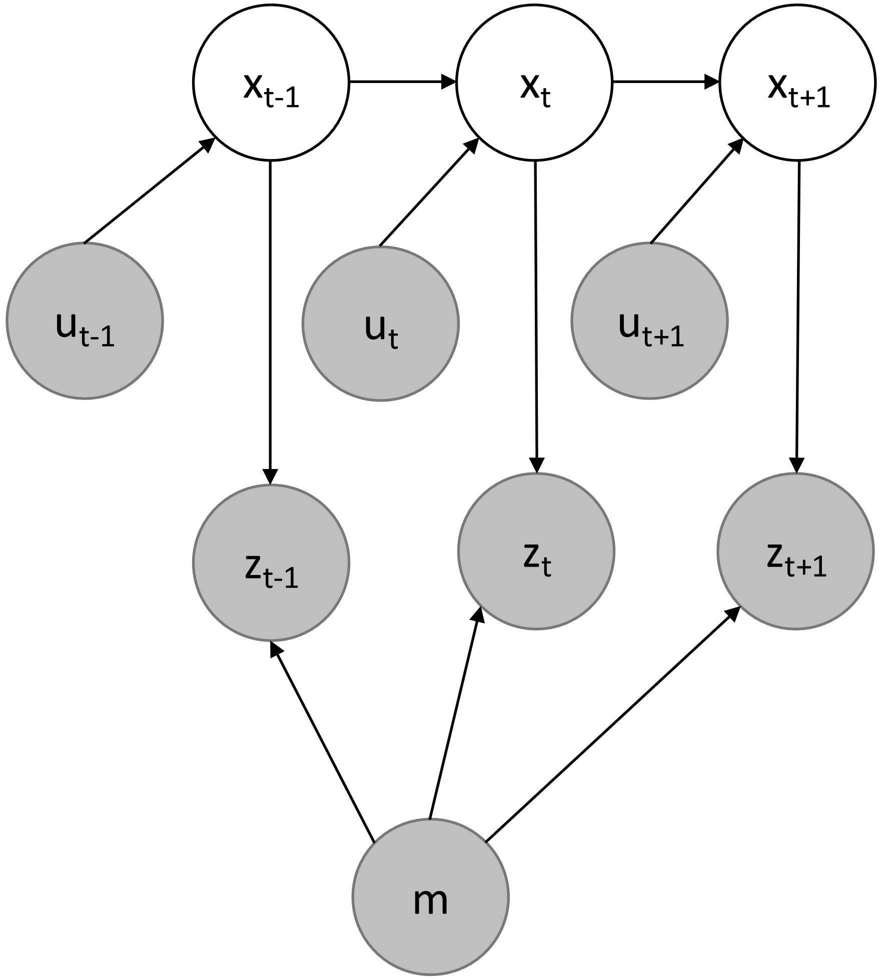
- $x_{t_1:t_2} = x_{t_1}, x_{t_1+1}, x_{t_1+2}, \ldots, x_{t_2}$ sequence of states t_1 to t_2
- Robot takes one measurement at a time
  - $z_{t_1:t_2} = z_{t_1}, \ldots, z_{t_2}$ sequence of all measurements (observations) from t_1 to t_2
- Control also exercised at discrete steps
  - $u_{t_1:t_2} = u_{t_1}, u_{t_1+1}, u_{t_1+2}, \ldots, u_{t_2}$ sequence control inputs

Localization as coordinate transformation

Shaded known:
map (m), control inputs (u), measurements(z). White nodes to be determined (x)

maps (m) are described in global coordinates. Localization = establish *coord transf.* between m and robot's local coordinates

Transformation used for objects of interest (obstacles, pedestrians) for decision, planning and control



Review of conditional probabilities

A random variable is a function $X: \Omega \rightarrow \mathbb{R}^n$ that assigns numerical values to the outcomes of a random experiment. Ω is the sample space.

Random variable X takes values $x_1, x_2 \in \mathbb{R}^n$

Example: Result of a dice roll (X) and $x_i = 1, \dots, 6$

$P(X = x)$ is written as $P(x)$

$P(X = x, Y = y)$ is written as $P(x, y)$

Conditional probability: $P(X = x \mid Y = y) = P(x|y) = \frac{P(x,y)}{P(y)}$ provided $P(y) > 0$

$P(x, y) = P(x|y)P(y)$

$= P(y|x)P(x)$

Substituting in the definition of Conditional Prob. we get Bayes Rule

$P(x|y) = \frac{P(y|x)P(x)}{P(y)}$, provided $P(y) > 0$

Using measurements to update state estimates

$P(x|y) = \frac{P(y|x)P(x)}{P(y)}$, provided $P(y) > 0$ *(*)*

X *: Robot position,* Y *: measurement,*

$P(x)$: *Prior distribution (before measurement)*

$P(x|y)$: *Posterior distribution (after measurement)*

$P(y|x)$*: Measurement model / inverse conditional / generative model*

$P(y)$: does not depend on x; normalization constant

Evolution: probabilistic Markov Chains

A probability distribution $\pi \in \boldsymbol{P}(Q)$ over a finite set of states Q can be represented by a vector $\pi \in \mathbb{R}^{|Q|}$ where $\sum \pi_i = 1$

Recall discrete transition $D: Q \to Q$

p_D: probability distribution $\mathrm{p_D}: Q \to \boldsymbol{P}(Q)$ according to which the next state is chosen

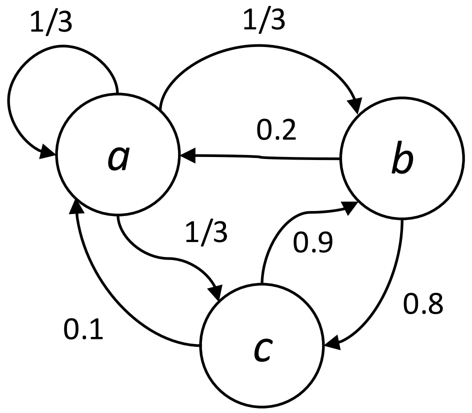
E.g., $p_D(X_{t+1} = b \,|X_t = a) = \frac{1}{3}$

State machine model called a Markov chain

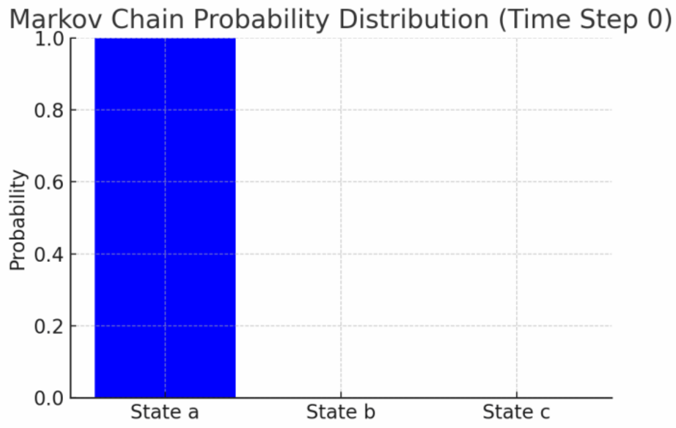
p_D can be represented by a matrix $\boldsymbol{D} \in \mathbb{R}^{|Q|\times|Q|}$ where D_{ij} give the probability of state i to transition to j

Evolution of the probability π over states:

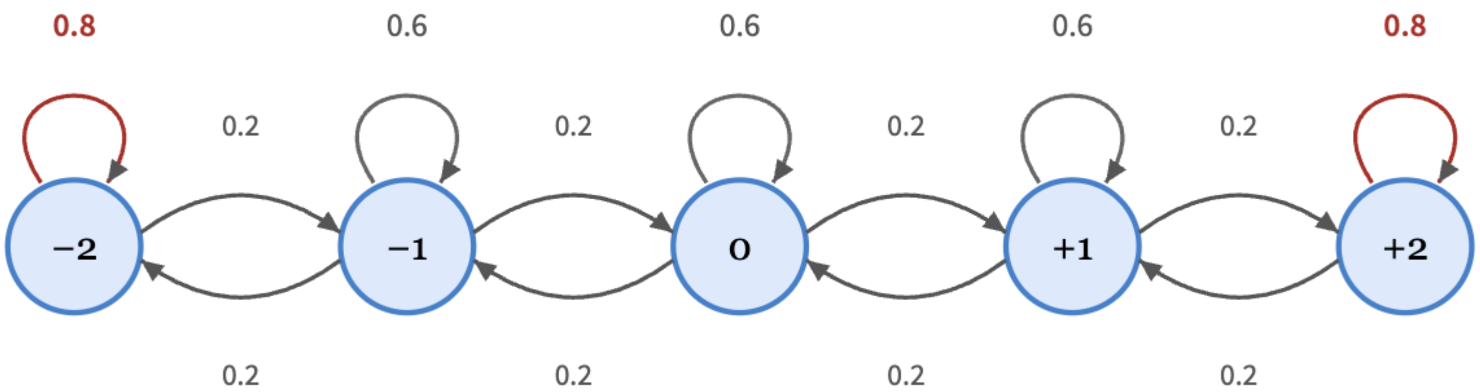
$\pi_{t+1} = \boldsymbol{D}\pi_t$ starting with an initial distribution $\pi_0 \in \boldsymbol{P}(Q)$



$$\boldsymbol{D} = \begin{bmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{5} & 0 & \frac{4}{5} \\ \frac{1}{10} & \frac{9}{10} & 0 \end{bmatrix}$$



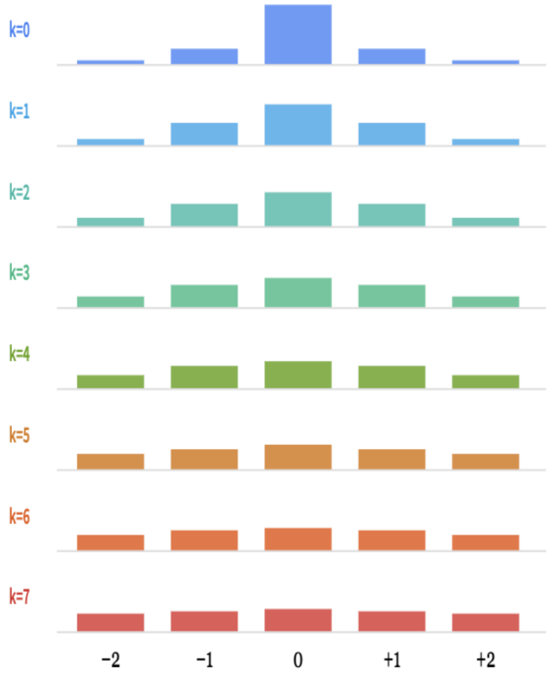
Example 1



- $\pi_{t+1} = \boldsymbol{D}\pi_t$
- $\pi_0 = [0.05, 0.15, 0.60, 0.15, 0.05]$

$$\boldsymbol{D} = \begin{bmatrix} 0.8 & 0.2 & 0 & 0 & 0 \\ 0.2 & 0.6 & 0.2 & 0 & 0 \\ 0 & 0.2 & 0.6 & 0.2 & 0 \\ 0 & 0 & 0.2 & 0.6 & 0.2 \\ 0 & 0 & 0 & 0.2 & 0.8 \end{bmatrix}$$

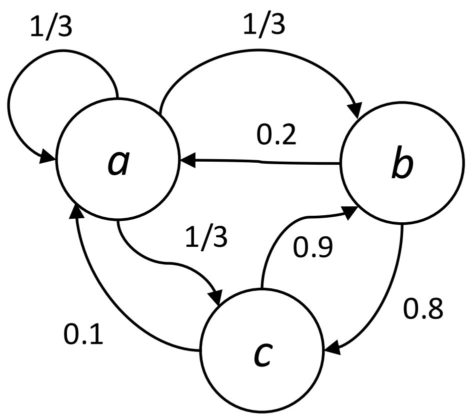
k	−2	−1	0	+1	+2
0	0.05	0.15	**0.60**	0.15	0.05
1	0.07	0.22	**0.42**	0.22	0.07
2	0.10	0.23	**0.34**	0.23	0.10
3	0.13	0.23	**0.30**	0.23	0.13
4	0.15	0.22	**0.27**	0.22	0.15
5	0.16	0.22	**0.25**	0.22	0.16
6	0.17	0.21	**0.24**	0.21	0.17
7	0.18	0.21	**0.23**	0.21	0.18



- $\pi^* = [\frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}]$ stationary distribution

Markov Decision Processes (MDP)

Generally, transitions depend on input $D: Q \times U \rightarrow P(Q)$

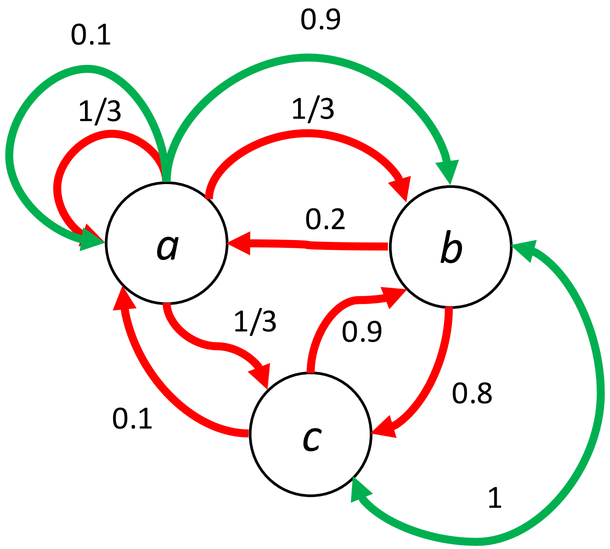


E.g. $p_D(X_{t+1} = b \,|X_t = a, U_t = red) = \frac{1}{3}$

Markov Decision Process (MDP)

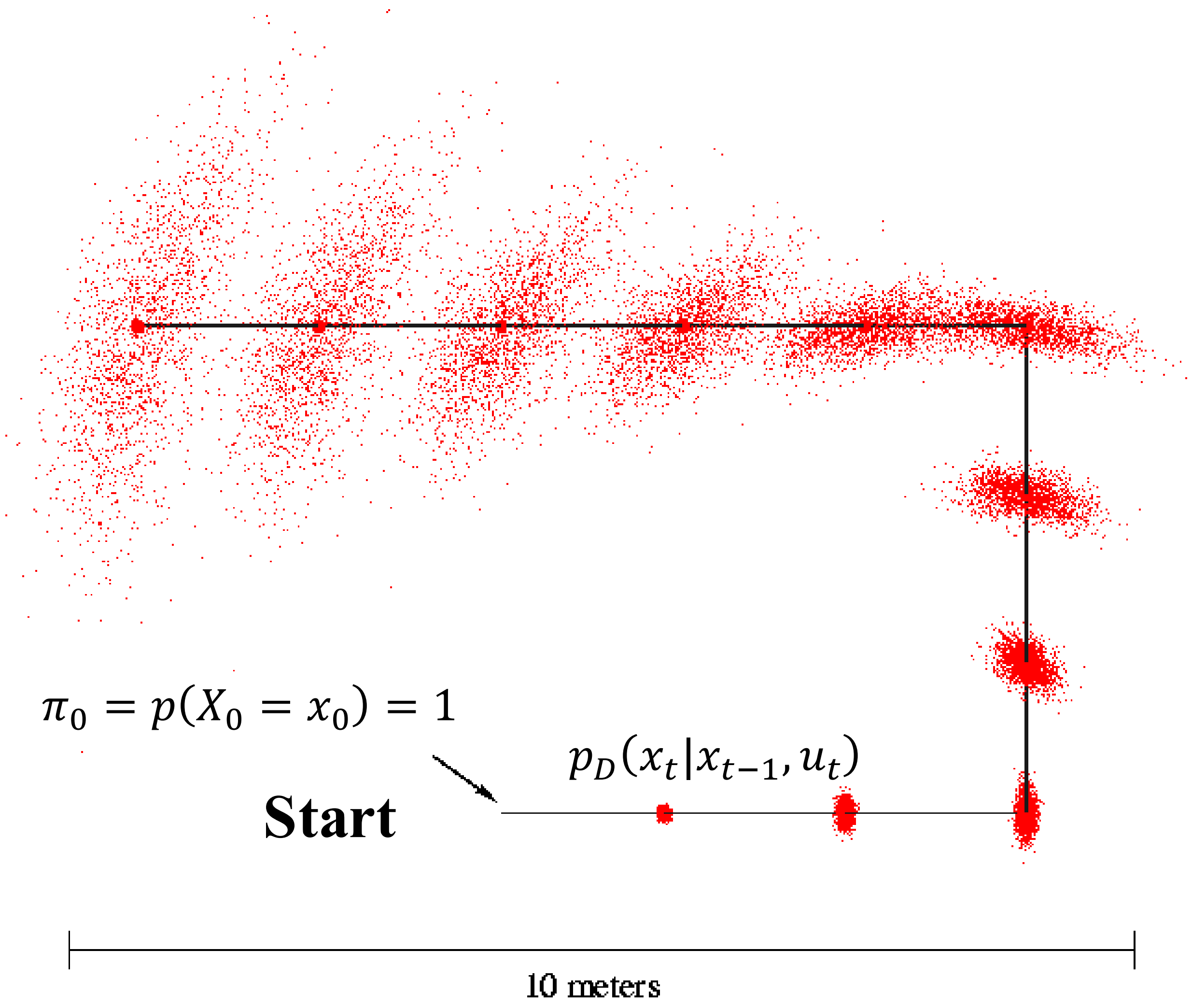
Transitions $\boldsymbol{D}$ represented by a collection of matrices

$\boldsymbol{D}: U \rightarrow \mathbb{R}^{|Q|\times|Q|}$ where $D_{ij}(u)$ gives the probability of state $\boldsymbol{i}$ to transition to j under action u



$p_D(x'|x, u)$ if transition probabilities are time invariant

Example Motion Model without measurements



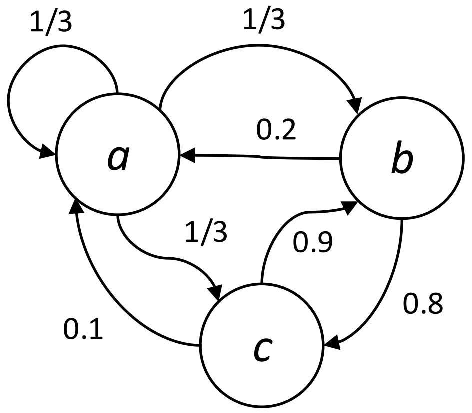
The state transition probabilities are defined by
$x_{t+1} = f(x_t, u_t) + \omega_t$

where $\omega_t \sim N(0,1)$

Evolution and measurement: probabilistic models

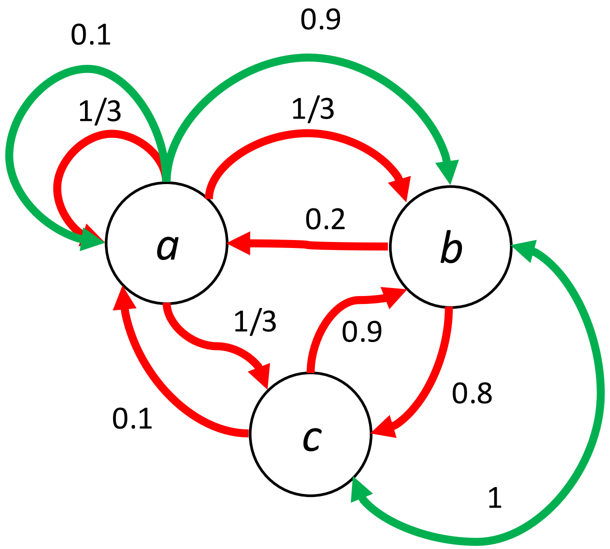
Even more generally, transitions depend on outputs and history

$p_D(X_t = x_t \,|X_0 = x_0, \dots X_{t-1} = x_{t-1,} Z_1 = z_1, \dots Z_{t-1} = z_{t-1,}, U_1 = u_1, \dots U_t = u_{t,})$ describes state evolution model



$p_D(x_t \,|x_{0:t-1}, z_{1:t-1}, u_{1:t})$ describes motion/state evolution model

If state is complete, sufficient summary of the history then



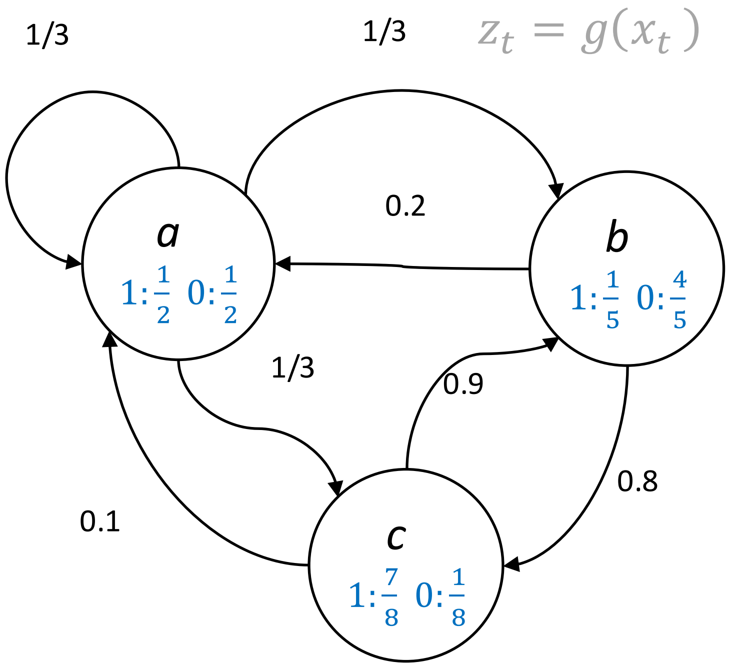
Markov Assumption

- $p_D(x_t \,|x_{0:t-1}, z_{0:t-1}, u_{0:t-1}) = p_D(x_t \,|x_{t-1}, u_t)$

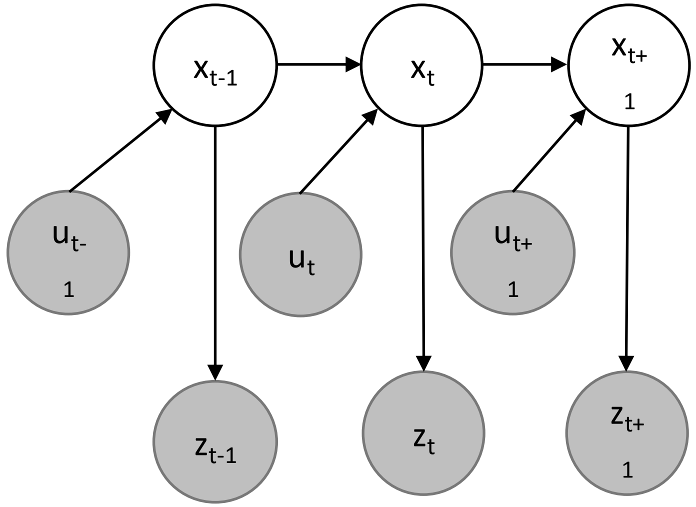
Time invariant or stationary MDP

- $p_D(x'|x, u)$

Probabilistic measurements



State a produces output 1 and 0 each with probability 0.5



Measurement model gives the output/observation probability for a given state

$$p_M(z_t = 1 | x_t = a) = \frac{1}{2}$$

Generally, measurements can depend on history

$$p_M(z_t \,| x_{0:t}, z_{1:t-1}, u_{0:t-1})$$

- If state is complete $p_M(z_t \,| x_{0:t}, z_{1:t-1}, u_{1:t}) = \ p(z_t \,| x_t)$
- $p_M(z_t \,| x_t)$: measurement probability
- $p_M(z \,| x)$: time invariant measurement probability

Example: Measurement Model p_M

Discrete Gaussian sensor model. Given the drone is truly at x, the sensor reading z_k is distributed as:

$$p_M(z_k \mid x) \propto \exp\left(-\frac{(x - z_k)^2}{2\sigma^2}\right), \sigma = 1\, m$$

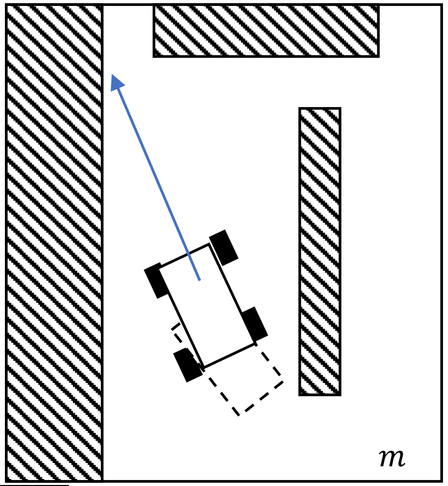
The likelihood vector $\ell(x) = p_M(z_k \mid x)$ evaluated at each state — given a sensor reading — tells us how well each state explains that reading.

Normalized likelihood values for z_k = +1 m:

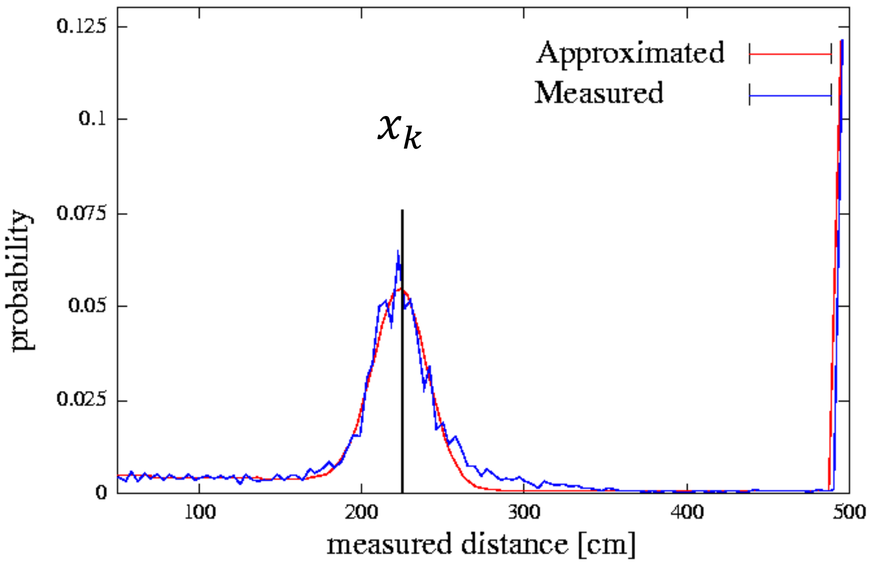
x	−2	−1	0	+1	+2
p_M	0.02	0.08	0.25	**0.45**	0.20

Peak at x = +1 (matches z_k); falls off for states farther away. Slight asymmetry (0.25 vs 0.20) is from discrete normalization.

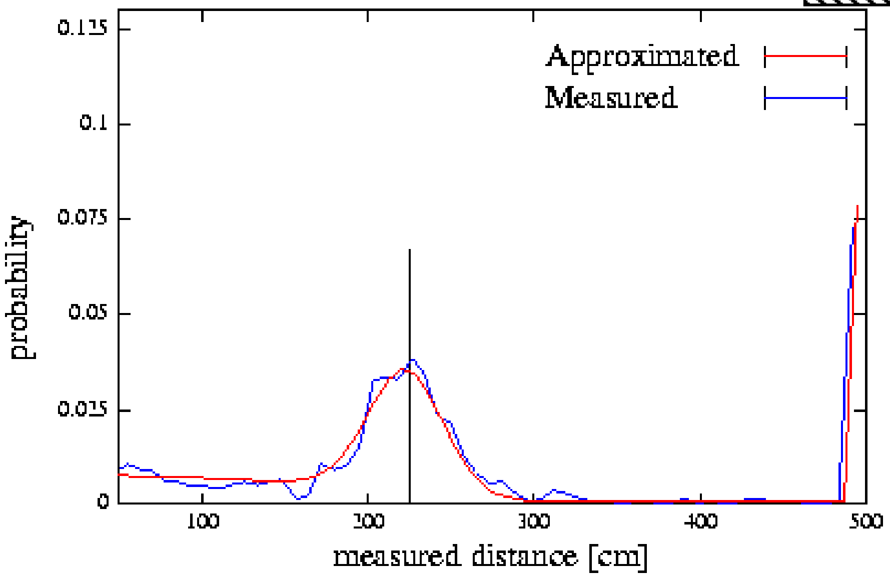
Example Proximity Sensor Measurement Models



$p_M(z_t \,|X_t = x_k)$



Laser sensor



Sonar sensor

Summary so far: Evolution and measurement

$p_D(x_t \,|x_{0:t-1}, z_{1:t-1}, u_{1:t})$ describes motion/state evolution model

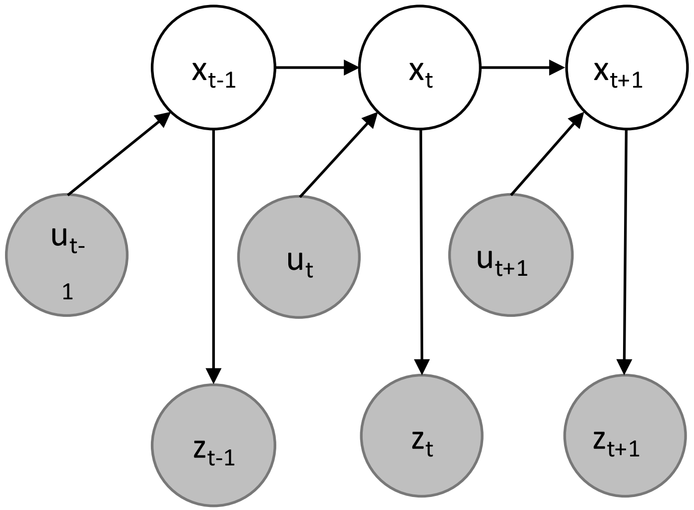
If state is complete, sufficient summary of the history then

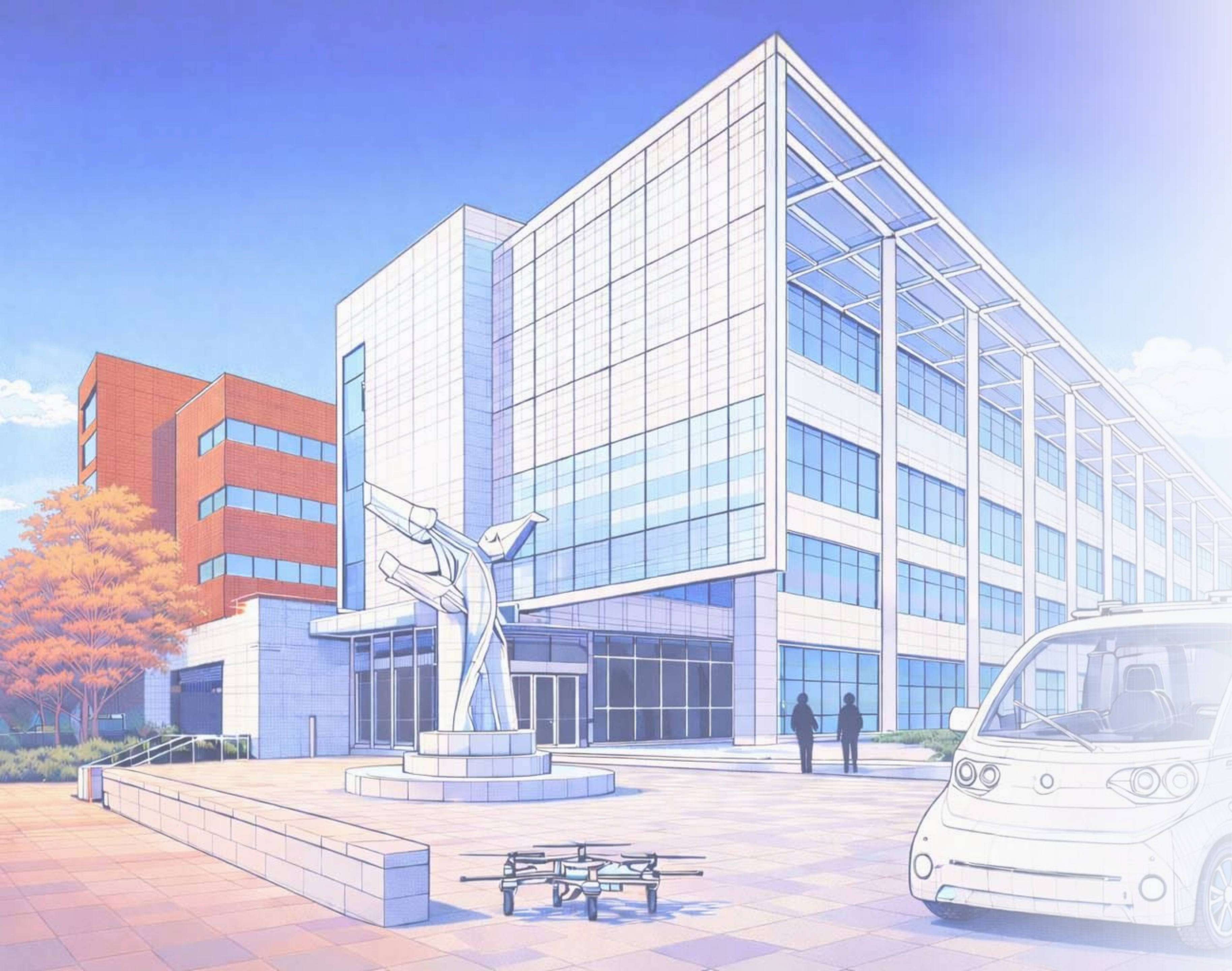
- $p_D(x_t \,|x_{0:t-1}, z_{0:t-1}, u_{0:t-1}) = p_D(x_t \,|x_{t-1}, u_t)$ motion model
- $p_D(x'|x, u)$ time invariant Markov

$p_M(z_t \,|x_{0:t}, z_{1:t-1}, u_{0:t-1})$ describes measurement

If state is complete

- $p_M(z_t \,|x_{0:t}, z_{1:t-1}, u_{1:t}) = \; p(z_t \,|x_t)$ measurement model
- $p_M(z \,|x)$: time invariant measurement probability





Outline

Probabilistic models
Discrete Bayes Filter
Kalman Filter

Beliefs

Belief: Robot's knowledge about the state

True state x_t is not measurable robot must estimate state $\hat{x}_t$ from measurements and this in Bayesian filtering the estimate is the *belief*

$bel(x_t) = p(x_t|z_{1:t}, u_{1:t}) \in \boldsymbol{P}(Q)$

Posterior over states at time t given all past measurements and control.

Calculated in two steps:

Initially: $bel(x_0) = \pi_0$

1. Prediction: $\overline{bel}(x_t) = p(x_t|z_{1:t-1}, u_{1:t})$ based ***all but last*** measurements & all control
2. Correction: $bel(x_t)$ from $\overline{bel}(x_t)$ based on last measurement z_t

Bayes Filter: Prediction and Correction

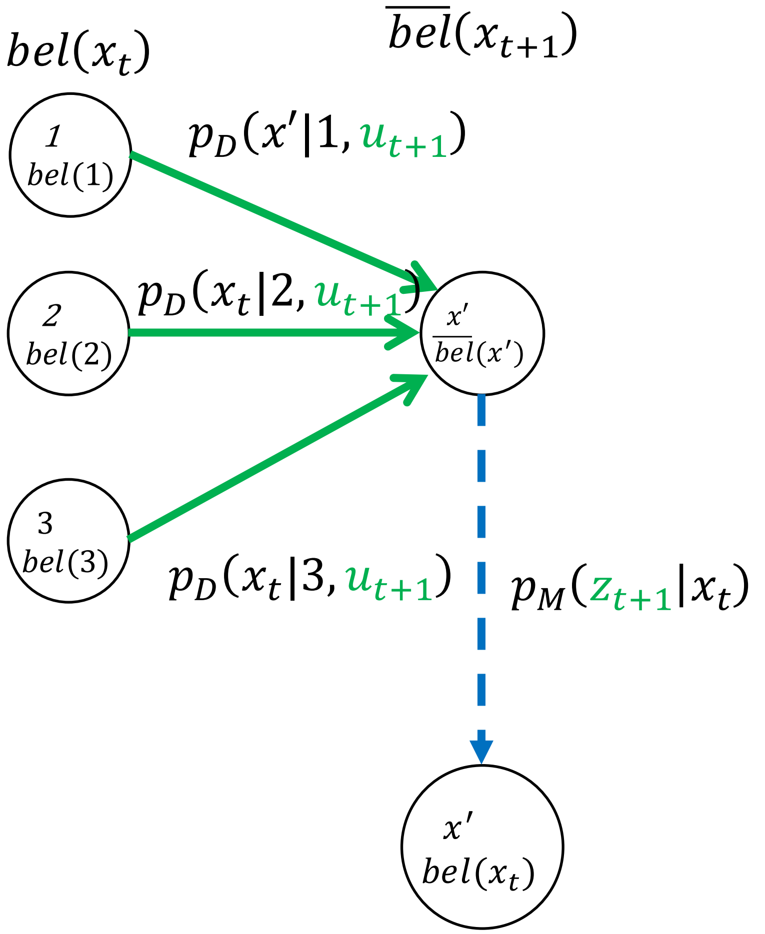
Bayes_filter($bel(x_t), u_{t+1}, z_{t+1}$) calculates $bel(x_{t+1})$

$$\overline{bel}(x_{t+1}) = p(x_{t+1}|z_{1:t}, u_{1:t+1}) = p(x_{t+1}|u_{t+1})$$

intermediate belief without $z_{1:t+1}$

For discrete distributions for each $x' \in Q$

$$\boxed{\overline{bel}(X_{t+1} = x') = \sum_{x \in Q} p_D(X_{t+1} = x'|X_t = x, U_{t+1} = u_{t+1})\, bel(X_t = x)}$$



Bayes Filter: Prediction and Correction

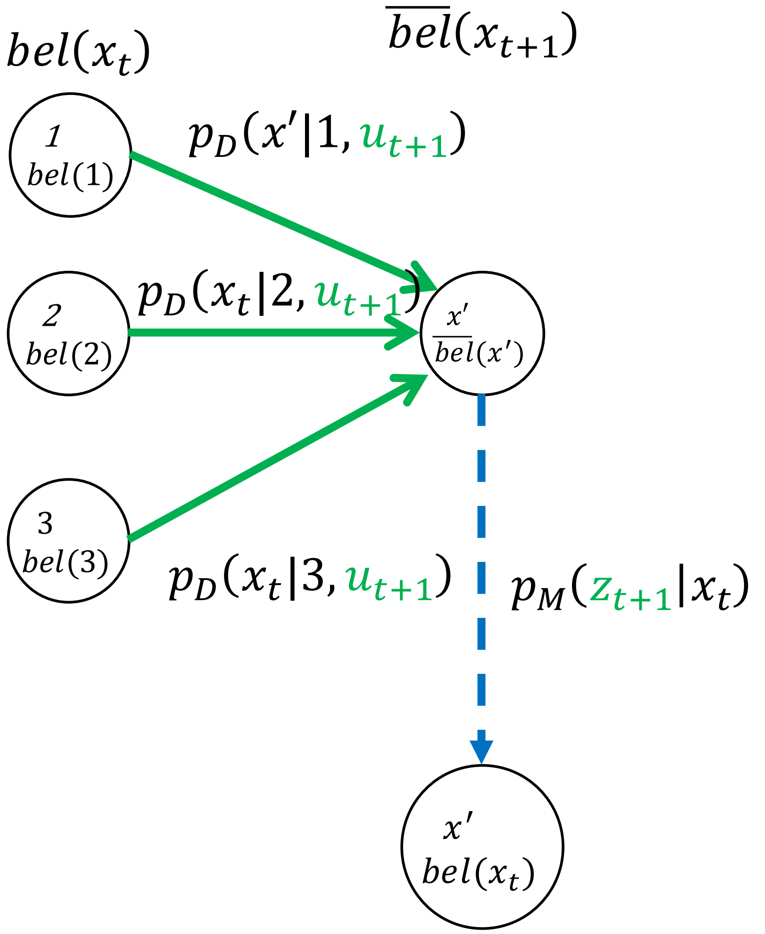
Bayes_filter($bel(x_t), u_{t+1}, z_{t+1}$) calculates $bel(x_{t+1})$

$$\overline{bel}(X_{t+1} = x') = \sum_{x \in Q} p_D(X_{t+1} = x' | X_t = x, U_{t+1} = u_{t+1})\, bel(X_t = x)$$

$$bel(x') = p(\, x' \mid z_{t+1}\,) = p(z_{t+1} \mid x') \cdot \frac{p(x')}{p(z_{t+1})}$$

Bayes rule

$$\boxed{bel(x') = \eta\, p_M(Z_t = z_{t+1} | X_t = x)\, \overline{bel}(X_t = x)}$$



where η is a normalizing constant to make $bel(x_{t+1}) \in \boldsymbol{P}(Q)$

$\overline{bel}(X_t = x)$ predicted prior; believed state **before** measurement

$p_M(z_{t+1} \mid x')$ is the likelihood of this sensor reading given the state is x′

Histogram Filter or Discrete Bayes Filter

Finitely many states $x_i, x_k, etc.$ Random state vector X_t

$p_{k,t}$: belief at time t for state x_k; discrete probability distribution

Algorithm Discrete_Bayes_filter($\{p_{k,t-1}\}, u_t, z_t$**):**

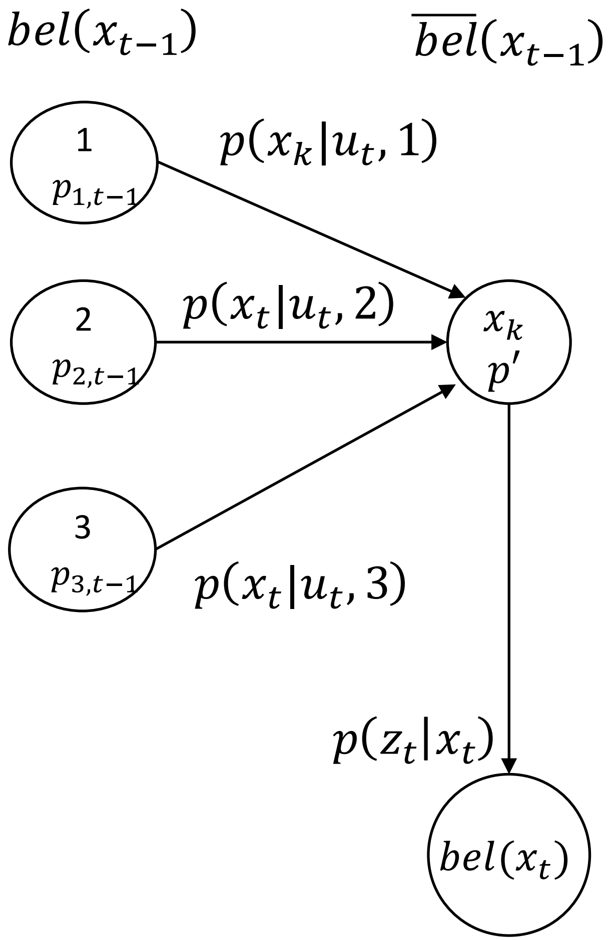
for all k do:

$$\bar{p}_{k,t} = \sum_i p(X_t = x_k | u_t, X_{t-1} = x_i) p_{i,t-1}$$

$$p_{k,t} = \eta \, p(z_t \,| X_t = x_k) \bar{p}_{k,t}$$

end for

return $\{p_{k,t}\}$



Bayes Filter: Continuous Distributions

Algorithm Bayes_filter($bel(x_{t-1}), u_t, z_t$)

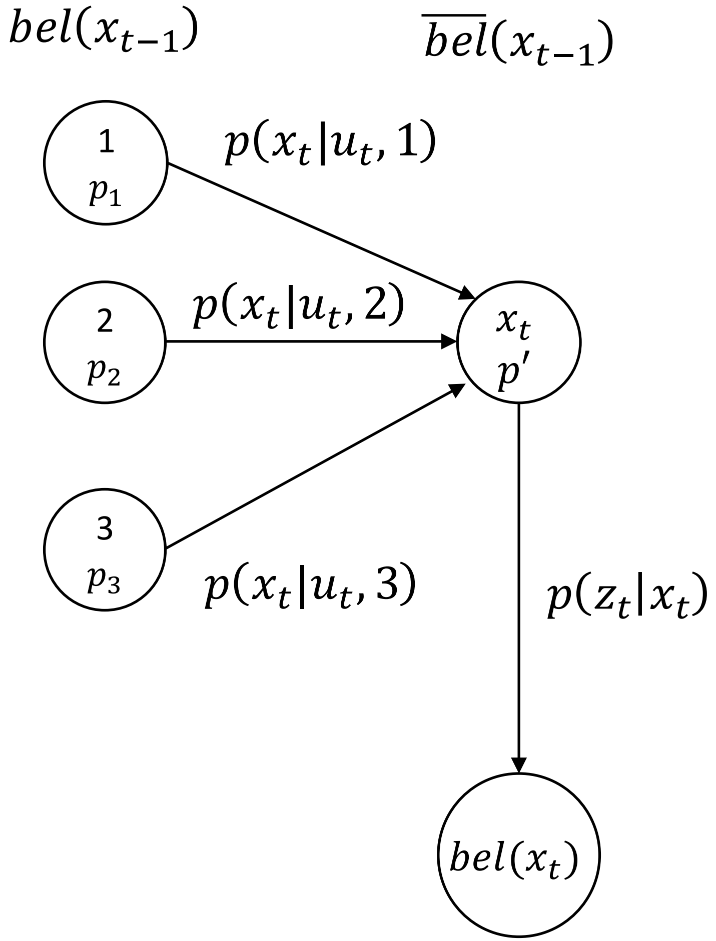
for all x_t do:

$$\overline{bel}(x_t) = \int p(x_t | u_t, x_{t-1}) bel(x_{t-1}) dx_{t-1}$$

$$bel(x_t) = \eta \, p(z_t | x_t) \, \overline{bel}(x_t)$$

end for

return $bel(x_t)$



Histogram Filter Example — Iteration 1

States: $Q = \{-2, -1, 0, +1, +2\}$ True $x_0 = -2$ $bel(x_0) = [0.20, 0.20, 0.20, 0.20, 0.20]$ (uniform)

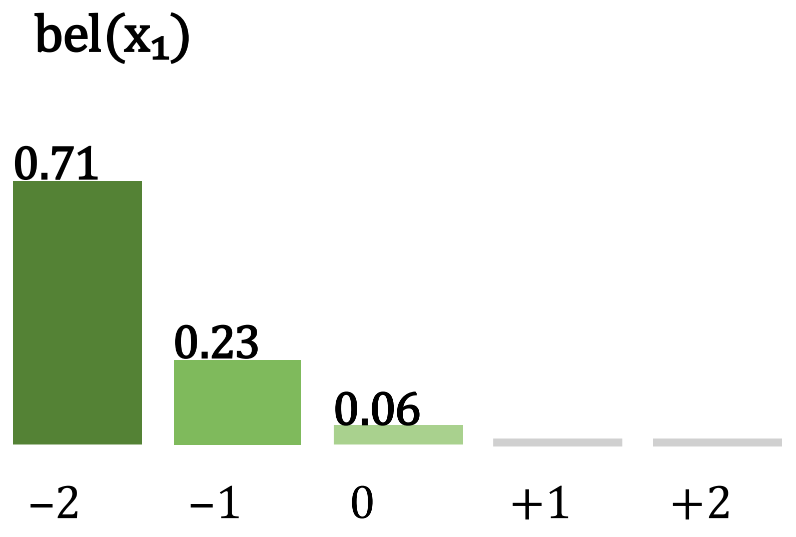
Step 1a — Prediction: $\overline{bel}(x_1) = \Sigma\, P(x_1 \mid x_0) \cdot bel(x_0)$

Since $bel(x_0)$ is uniform, the prediction stays uniform:
$\overline{bel}(x_1) = [0.20, 0.20, 0.20, 0.20, 0.20]$

Step 1b — Update: observe $z_1 = -2$ (true $x_1 = -2$)

$$bel(x_1) = \eta \cdot p_m(z_1 \,/\, x_1) \cdot \overline{bel}(x_1)$$

x_1	-2	-1	0	$+1$	$+2$
$p_m(z_1{=}{-2} \mid x_1)$	0.25	0.08	0.02	0.00	0.00
$\overline{bel}(x_1)$	0.20	0.20	0.20	0.20	0.20
Unnormalized	0.050	0.016	0.004	0.000	0.000
$bel(x_1)$	**0.714**	**0.229**	**0.057**	**0.000**	**0.000**



→ **After $z_1 = -2$: $bel(x_1)$ peaks at $x_1 = -2$ with 71.4% — correctly identifies the true state!**

$\eta = 1/0.070 \approx 14.29$

Histogram Filter Example — Iteration 2

Step 2a — Prediction: $\overline{bel}(x_2) = \Sigma\, P(x_2 \mid x_1) \cdot bel(x_1)$

$\overline{bel}(x_2) = T^T \cdot bel(x_1) = [0.617, 0.291, 0.080, 0.011, 0.000]$
(Transition matrix spreads probability to neighboring states → increased uncertainty)

Step 2b — Update: observe $z_2 = 0$ (true $x_2 = -1$)

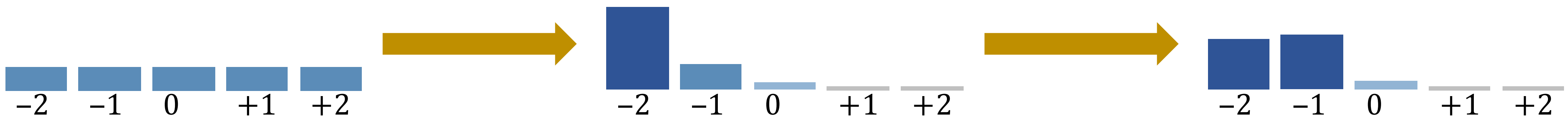
$$bel(x_2) = \eta \cdot p_m(z_2 \mid x_2) \cdot \overline{bel}(x_2)$$

x_2	−2	−1	0	+1	+2
$p_m(z_2=0 \mid x_2)$	0.20	0.45	0.25	0.08	0.02
$\overline{bel}(x_2)$	0.617	0.291	0.080	0.011	0.000
Unnormalized	0.123	0.131	0.020	0.001	0.000
$bel(x_2)$	**0.448**	**0.476**	**0.073**	**0.003**	**0.000**

→ **$bel(x_2)$ peaks at $x_2 = -1$ with 47.6% — tracks the shift!**

Belief Convergence — How the Histogram Peaks

$bel(x_0)$ (uniform) $bel(x_1)$ after $z_1=-2$ $bel(x_2)$ after $z_2=0$



Optimality of Bayes Filter

MLE (Maximum Likelihood Estimate)

$$\hat{x}_t^{MLE} = \text{argmax}_x \; p(z_t \mid x)$$

Finds the state that best explains the measurement alone — no prior.

It does not use $bel(x_t)$ at all, so it ignores all history.

MAP (Maximum A Posteriori)

$$\hat{x}_t^{MAP} = \text{argmax}_x \; bel(x_t) = \text{argmax}_x \; p(x_t \mid z_{1:t}, u_{1:t})$$

Finds the mode of the posterior. Optimal under 0–1 loss:

$$L(\hat{x}, x) = 0 \text{ if } \hat{x} = x, \quad 1 \text{ if } \hat{x} \neq x$$

MMSE (Minimum Mean Square Error)

$$\hat{x}_t^{MMsE} = E[x_t \mid z_{1:t}, u_{1:t}] = \int x \, bel(x_t) \, dx$$

Finds the mean of the posterior. Optimal under squared error loss:

$$L(\hat{x}, x) = (\hat{x} - x)^2$$

Discrete Bayes (Histogram) Filter — MLE, MAP, and MMSE

MLE: Not directly computed. The filter maintains $\mathrm{bel}(x_t)$, not $p(z_t \mid x)$.
However, the measurement update step uses likelihoods internally.

MAP: $\hat{x}_t^{\mathrm{MAP}} = \mathrm{argmax}_x\, \mathrm{bel}(x_t)$. Simply pick the grid cell with highest belief.
This is the most common point estimate from a histogram filter.

MMSE: $\hat{x}_t^{\mathrm{MMsE}} = \Sigma_x\, x \cdot \mathrm{bel}(x_t)$. Computed as a weighted average over the grid.
Unlike the KF, MAP ≠ MMSE in general (the posterior can be multimodal).

Key Properties of Bayes Filter

Nonparametric: No assumption on the shape of $\mathrm{bel}(x_t)$ — it can be multimodal, skewed, or arbitrary.

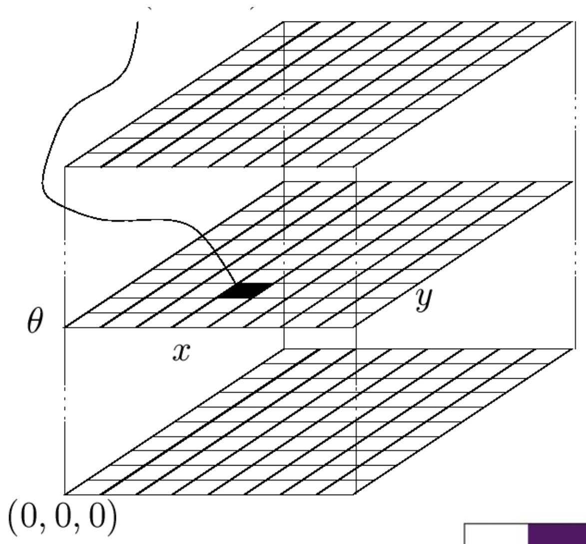
Exact (on the grid): $\mathrm{bel}(x_t)$ is the true Bayesian posterior over the discrete state space — no distributional approximation (unlike the KF, which forces a Gaussian). Therefore MAP and MMSE derived from $\mathrm{bel}(x_t)$ are optimal within the grid. The only source of error is grid resolution, not the algorithm.

Computational cost: $O(|Q|^2)$ per step (prediction requires summing over all state pairs). Scales poorly to high-dimensional state spaces.

No optimality guarantee: Unlike the KF, the histogram filter does not minimize any global error criterion. It is a consistent estimator as $\Delta x \to 0$ but is not MVUE or BLUE in general.

Grid localization with bicycle model + landmarks

$bel(X_t = \langle x, y, \theta \rangle)$



The state space Q is a quantization of position and orientation $q = \langle x, y, \theta \rangle$

A belief is a probability distribution over states $\mathrm{bel}(q_t) \in P(Q)$

Prediction: Fixing an (steering) input u_t compute the new intermediate belief over Q using motion model $p_D(q_{t+1} | q_t, u_{t+1})$

Correction: Update intermediate belief with received distance to landmark z_{t+1} based on measurement model p_M