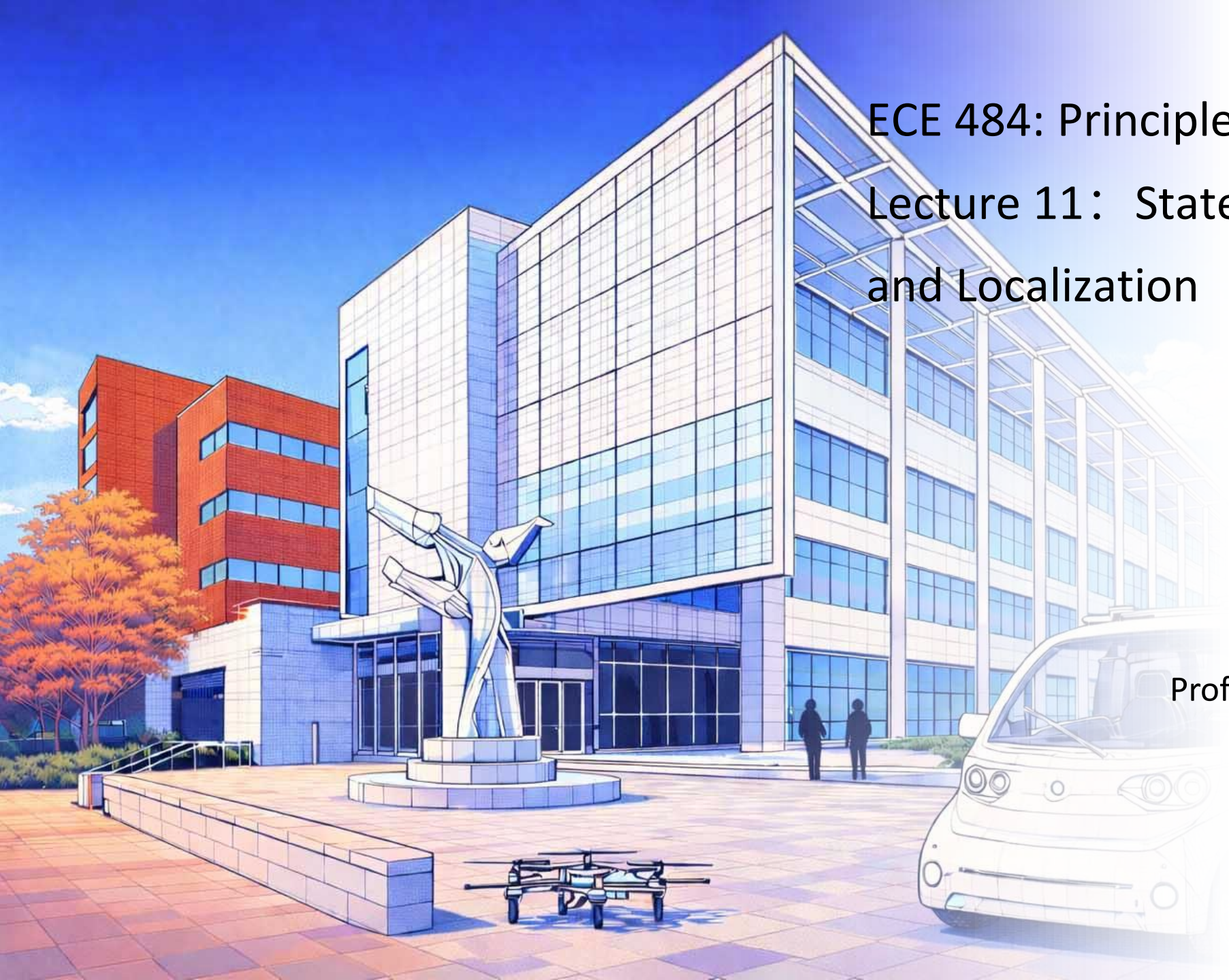


ECE 484: Principles of Safe Autonomy

Lecture 11： State Estimation, Filtering and Localization

Professor: Huan Zhang

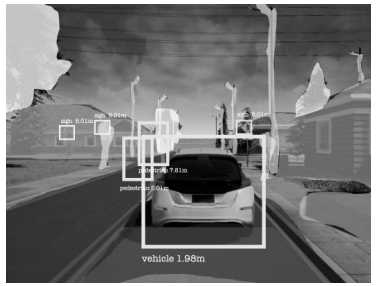
Fall 2026

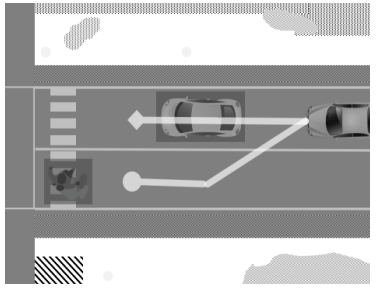
Autonomy pipeline

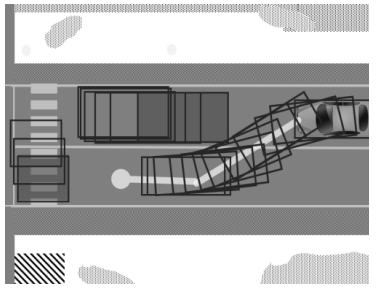
GEM platform





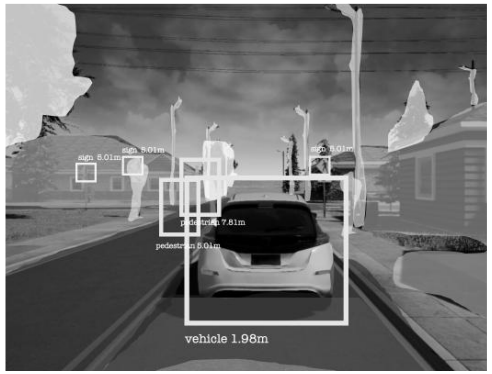






Sensing	Perception	Decisions and planning	Control
Physics-based models of camera, LIDAR, RADAR, GPS, etc.	Programs for object detection, lane tracking, scene understanding, etc.	Programs and multi-agent models of pedestrians, cars, etc.	Dynamical models of engine, powertrain, steering, tires, etc.

This class: from perception to state estimation



Perception

Programs for object detection, lane tracking, scene understanding, etc.

Outline

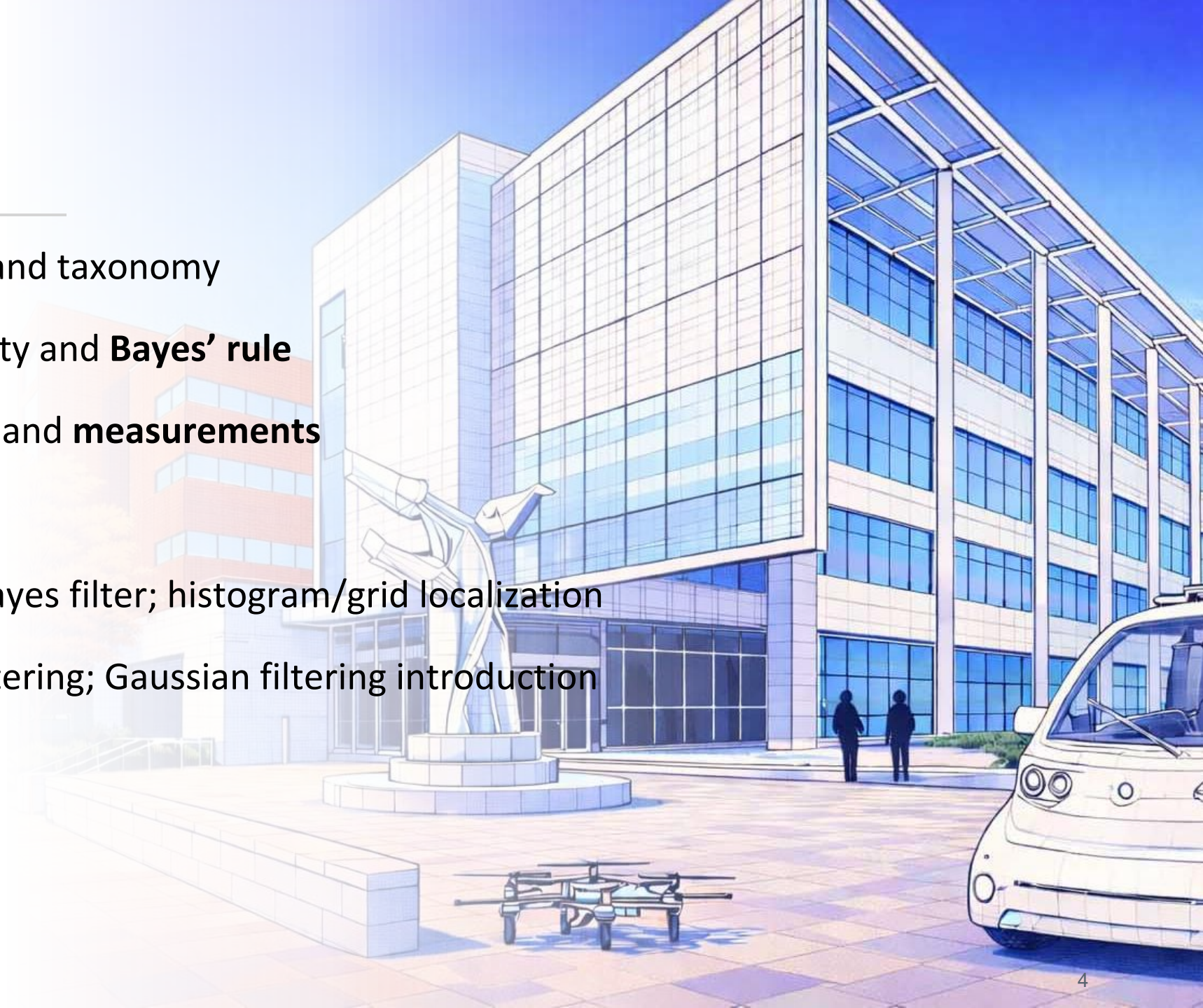
Today: localization problem and taxonomy

Review: conditional probability and **Bayes' rule**

Probabilistic models: **motion** and **measurements**

Next lectures:

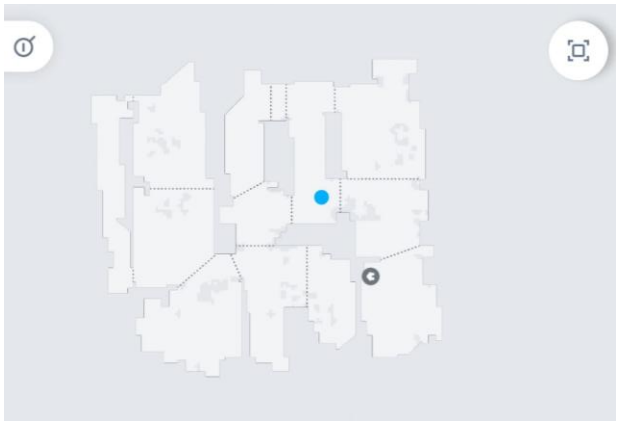
- Lecture 12: discrete Bayes filter; histogram/grid localization
- Lecture 13: particle filtering; Gaussian filtering introduction



Roomba mapping







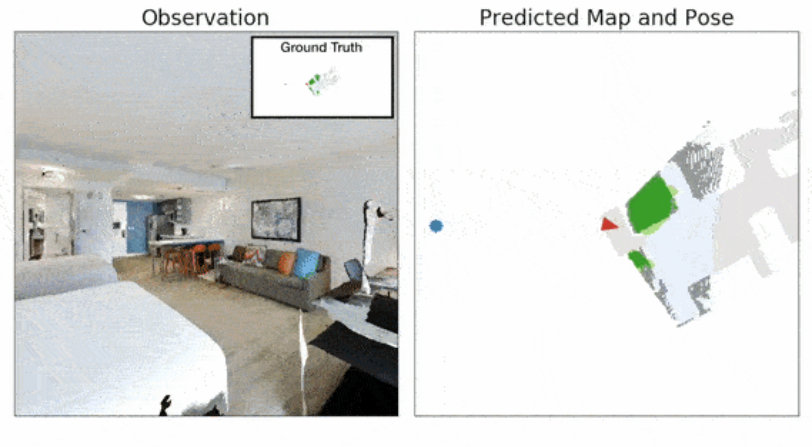


Image credit: Devendra Singh Chaplot

Mapping-capable Roomba models **build maps** and **estimate their pose**.

SLAM: simultaneous localization and mapping

Also used underground, underwater and in space, where GPS may be unavailable.

State estimation and localization problem

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.

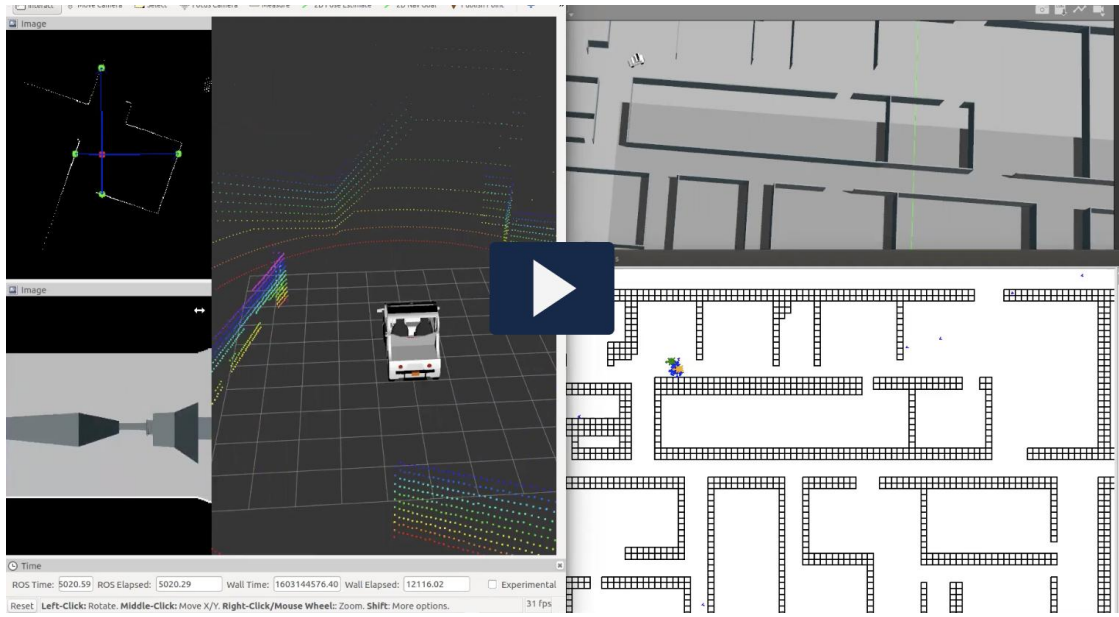
- Images, lidar scans, GPS, IMU

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



Setup: State evolution and measurement models

Deterministic state evolution:

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

x_t: unknown state at time t; f: known transition function.

u_t: known input from t to $t+1$; $u_t = g(\hat{x}_t)$.

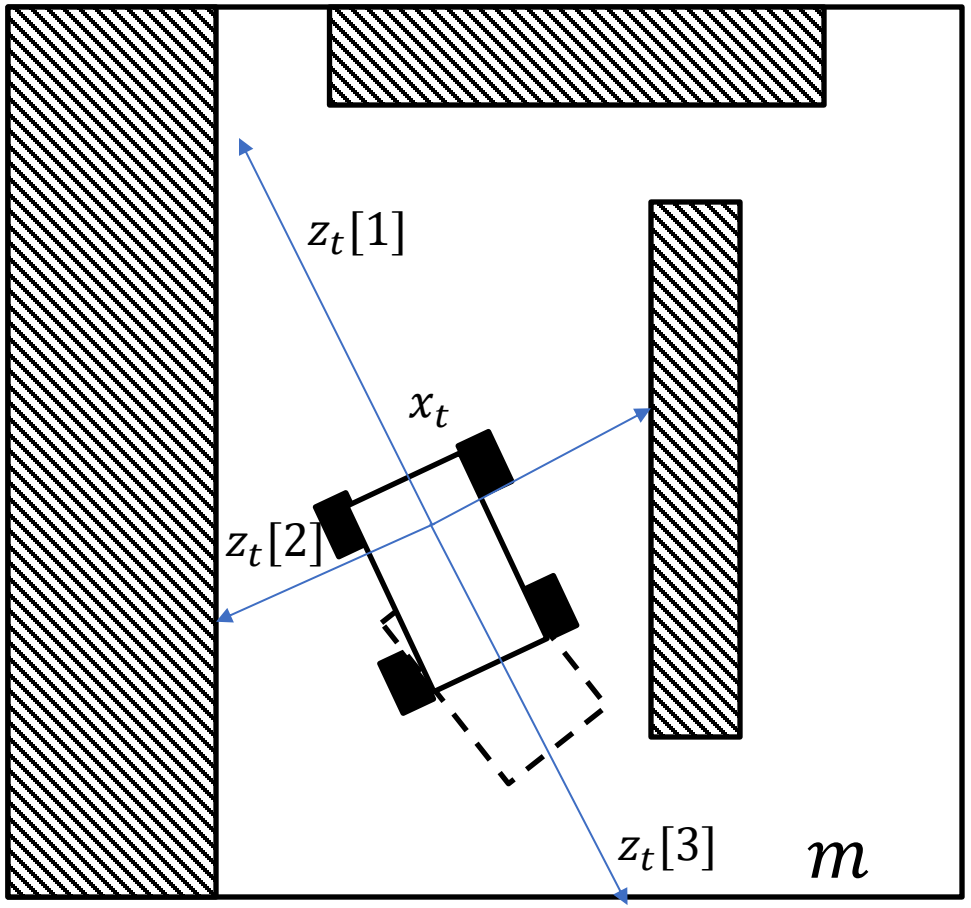
Ideal measurement: $z_t = h(x_t, m)$

z_t: observed sensor output; h: measurement function.

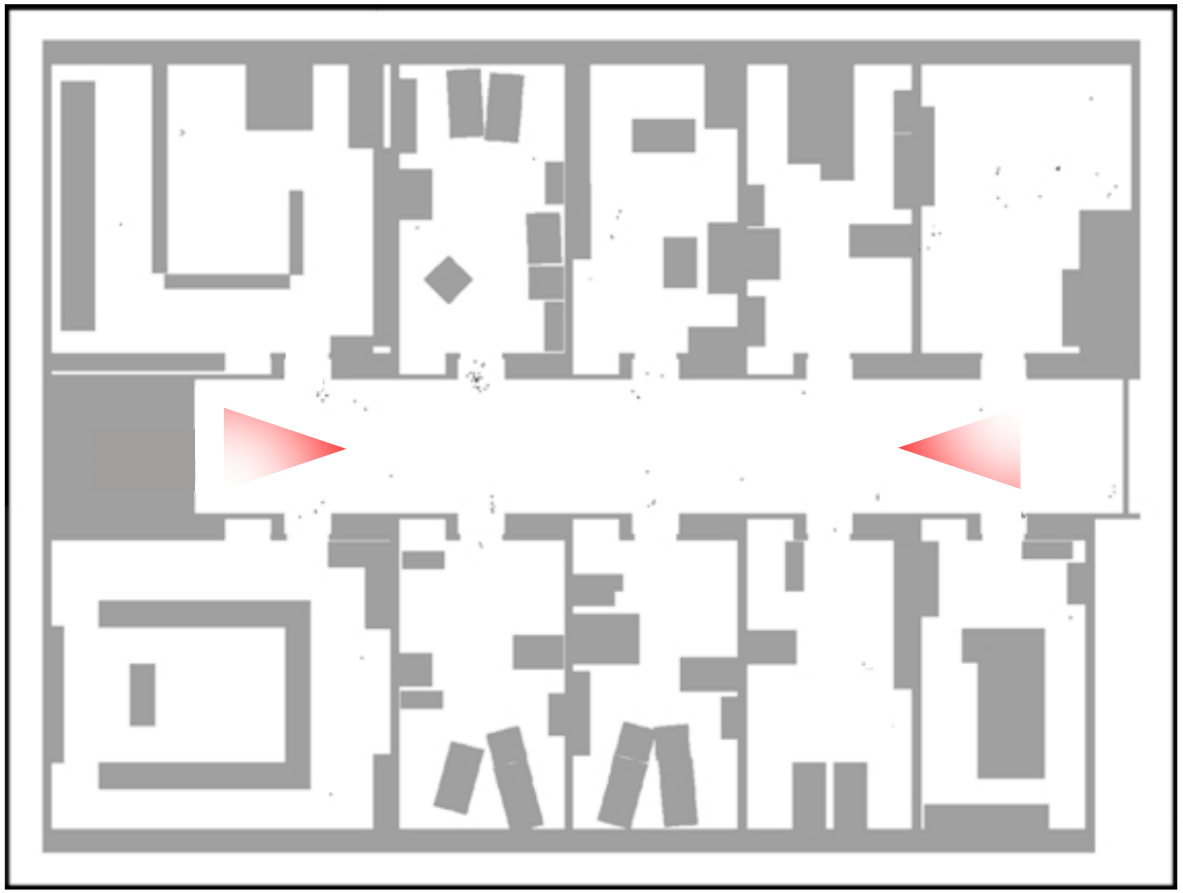
m: **given map** for localization (unknown map → SLAM).

Given $z_{1:t}$ and $u_{0:t-1}$, estimate x_t.

Next: use probability models for uncertain motion and sensing.



Global localization: similar places can look alike



Localization as coordinate transformation

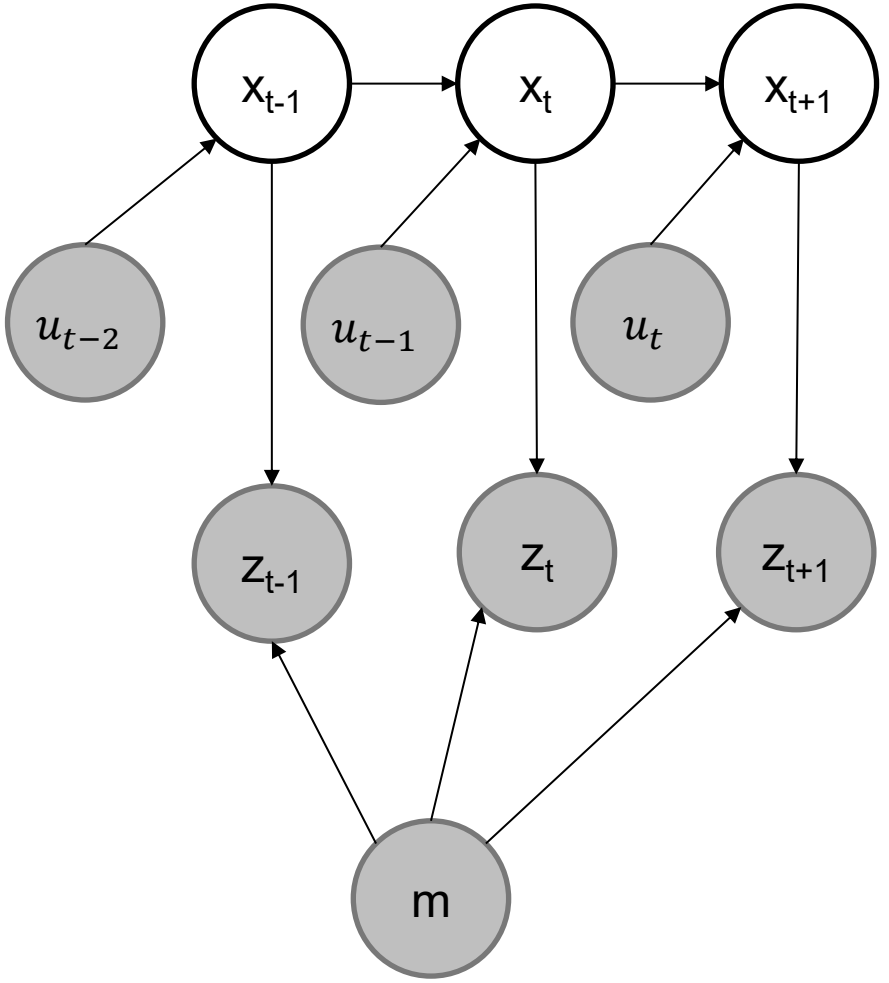
Shaded nodes are known: map m, inputs u and observations z.

White nodes are **unknown states** x.

The map is in global coordinates.

Localization finds the **transform** between the map frame and the robot frame.

Use this transform to locate obstacles and pedestrians for decisions, planning and control.



Localization taxonomy

Global versus local

- Local: initial pose estimate is known; track uncertainty from motion and measurements.
- Global: initial pose is unknown; includes position tracking once localized.
- Kidnapped robot: unexpected movement makes the old pose estimate wrong.

Static versus dynamic environments

Single versus multi-robot localization

Passive versus active approaches

- Passive: estimate pose while motions are chosen for other tasks.
- Active: choose motions that help determine pose.

Discrete time model: Automaton with inputs/outputs

Describe states, inputs and outputs as discrete-time sequences.

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

- x_t: state at time t; u_t: input applied from t to $t+1$.

Ideal measurement: $z_t = h(x_t, m)$

- z_t: measurement at x_t; m: **given map**.

Earlier: nondeterministic automata describe possible transitions.

Now: **probability distributions** describe uncertainty in motion and measurements.

Setup, notations

$x_{t_1:t_2} = (x_{t_1}, x_{t_1+1}, \dots, x_{t_2})$: state sequence, including both endpoints.

t_1, t_2: integer time indices, $t_1 \leq t_2$.

One measurement per time step (it may contain several components).

- $z_{t_1:t_2} = (z_{t_1}, \dots, z_{t_2})$: observation sequence.

Controls are applied over discrete time intervals.

- $u_{t_1:t_2} = (u_{t_1}, \dots, u_{t_2})$: control sequence.

u_{t-1} takes $x_{t-1} \to x_t$; u_t takes $x_t \to x_{t+1}$; z_t is taken at x_t.

Review of conditional probabilities

Two **independent**, fair six-sided dice:

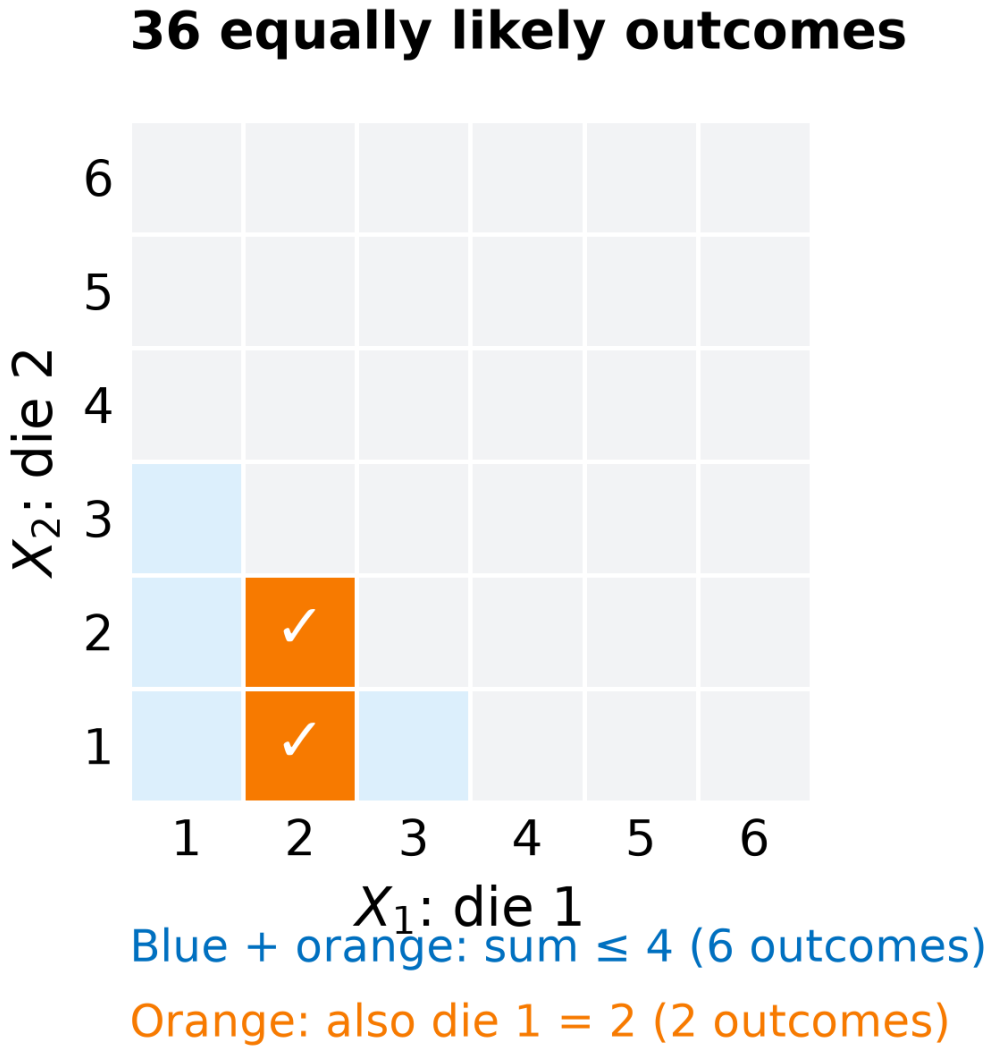
X_1: die 1; X_2: die 2.

Find the probabilities:

$$P(X_1 = 2)$$
$$P(X_1 + X_2 \leq 4)$$
$$P(X_1 = 2 | X_1 + X_2 \leq 4)$$

For the last question, restrict attention to the colored cells.





Probability review: random variables and conditioning

A **random variable** $X: \Omega \to \mathbb{R}^n$ assigns values to outcomes; Ω is the sample space and n the value dimension.

Die example: X takes values in $\{1,2,3,4,5,6\}$.

Shorthand notation: $P(x) = P(X = x)$; $P(x, y) = P(X = x, Y = y)$.

Conditional probability: $P(x|y) = \frac{P(x,y)}{P(y)}$, for $P(y) > 0$.

Read “given y”: restrict attention to outcomes where y occurred.

For continuous states use a **density** $p(x)$; probabilities are areas over intervals.

Probability review: product rule and Bayes' rule

Product rule (the same joint event, two factorizations):

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

Divide by $P(y) > 0$ to get **Bayes' rule**:

$$P(x|y) = \frac{P(y|x)P(x)}{P(y)}$$

The denominator sums all ways to see y: $P(y) = \sum_i P(y|x_i)P(x_i)$.

x_i: mutually exclusive, exhaustive state values; i: their index.

Using measurements to update state estimates

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

X: robot state; Z: measurement; x, z: their values.

$P(x)$: **prior** belief (before this measurement).

$P(x|z)$: **posterior** belief (after this measurement).

$P(z|x)$: **likelihood / forward measurement model**.

$P(z) = \sum_i P(z|x_i)P(x_i)$: same normalizing denominator for every candidate x_i.

Example: Light Sensor Robot

Problem Setup:

A robot has a light sensor that detects if a light is 'on' or 'off'

The sensor is noisy and cannot be fully trusted

Goal: Estimate the true state of the light using noisy measurements

Sensor Model (Measurement Model)

Sensor model:

True light ON: sensor reports “on” 90% of the time.

- $P(z_t = \text{on} | x_t = \text{on}) = 0.9$

True light OFF: sensor reports “on” 40% of the time.

- $P(z_t = \text{on} | x_t = \text{off}) = 0.4$

Step 1: Initial Belief (Prior)

Before this measurement, the robot believes:

$$P(x_t = \text{on}) = 0.5$$

$$P(x_t = \text{off}) = 0.5$$

The two true states are **equally likely** under this prior.

Step 2: Sensor Reading

The sensor now reports:

$$z_t = \text{on}$$

Question: what should the robot now believe?

We need $P(x_t = \text{on} | z_t = \text{on})$.

Step 2: Sensor Reading

Apply **Bayes' rule**:

$$P(x_t = \text{on} | z_t = \text{on}) = \frac{P(z_t = \text{on} | x_t = \text{on}) P(x_t = \text{on})}{P(z_t = \text{on})}$$

Posterior

Likelihood

Normalizing denominator

Prior/Belief

Step 3: Calculate normalization constant

Compute $P(z_t = \text{on})$ by **total probability**:

$$P(z_t = \text{on})$$

$$= P(z_t = \text{on}|x_t = \text{on})P(x_t = \text{on})$$

$$+P(z_t = \text{on}|x_t = \text{off})P(x_t = \text{off})$$

$$= (0.9)(0.5) + (0.4)(0.5)$$

$$= 0.45 + 0.20 = 0.65$$

100 cases with a 50–50 prior

True state	Sensor "on"	Sensor "off"
ON (50)	**45**	5
OFF (50)	**20**	30

Among 65 "on" readings, 45 lights are ON.

Step 4: Calculate Posterior

Update belief using **Bayes' rule**:

$$P(x_t = \text{on} | z_t = \text{on})$$

$$= \frac{P(z_t = \text{on} | x_t = \text{on}) P(x_t = \text{on})}{P(z_t = \text{on})}$$

$$= \frac{(0.9)(0.5)}{0.65} = \frac{0.45}{0.65} = \frac{9}{13} \approx 0.692$$

The posterior probability of ON is **69.2%**.

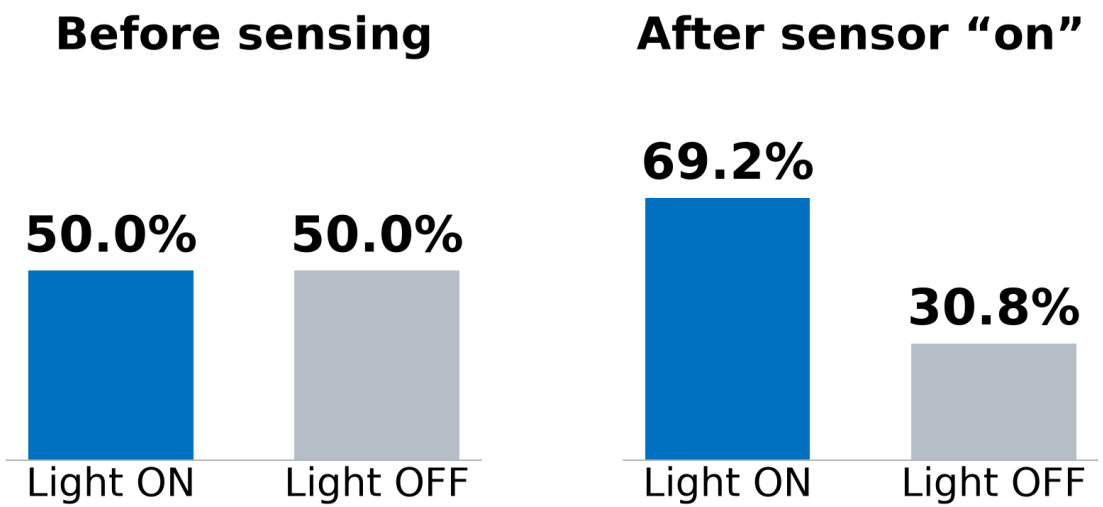
Result & Interpretation

Before sensing: **50%** probability the light is ON.

After sensor “on”: **69.2%** probability the light is ON.

The update combines:

- Prior belief (50% ON).
- Observed reading (“on”).
- Sensor model: 90% detection and 40% false positives.



Evolution: probabilistic Markov Chain models

A distribution $\pi \in P(Q)$ on finite states Q is a row vector in $\mathbb{R}^{|Q|}$: $\pi_i \geq 0$, $\sum_i \pi_i = 1$.

Deterministic one-successor transitions: $D: Q \to Q$.

Probabilistic transitions: $D: Q \to P(Q)$; $D(q)$ gives the next-state distribution from q.

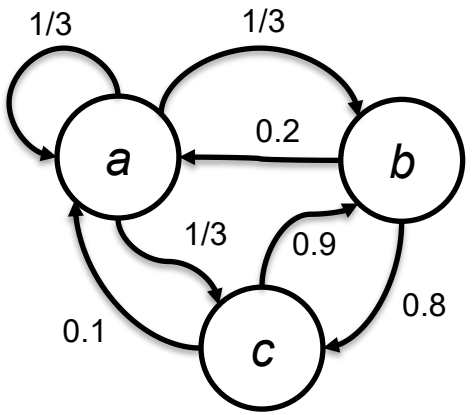
Example: $p_D(X_{t+1} = b | X_t = a) = \frac{1}{3}$.

$D(a) = [a{:}\frac{1}{3}, b{:}\frac{1}{3}, c{:}\frac{1}{3}]$; $D(b) = [a{:}\frac{1}{5}, b{:}0, c{:}\frac{4}{5}]$.

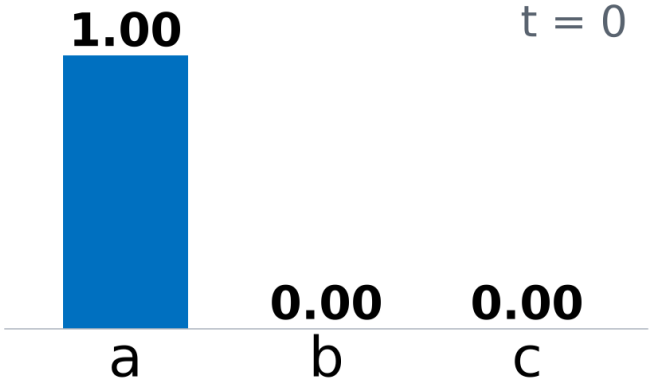
This model is a **Markov chain**.

D_{ij}: probability of state i going to j; every matrix **row sums to 1**.

Belief evolution: $\pi_{t+1} = \pi_t D$, starting with $\pi_0 \in P(Q)$.



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



Evolution: probabilistic MDP models

Controlled transitions: $D: Q \times U \to P(Q)$, where U is the action set.

Example: $p_D(X_{t+1} = b | X_t = a, u_t = \text{red}) = \frac{1}{3}$.

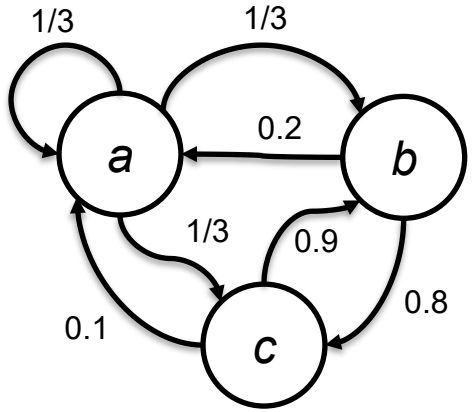
A **controlled Markov chain** is the transition model of an **MDP** (Markov decision process).

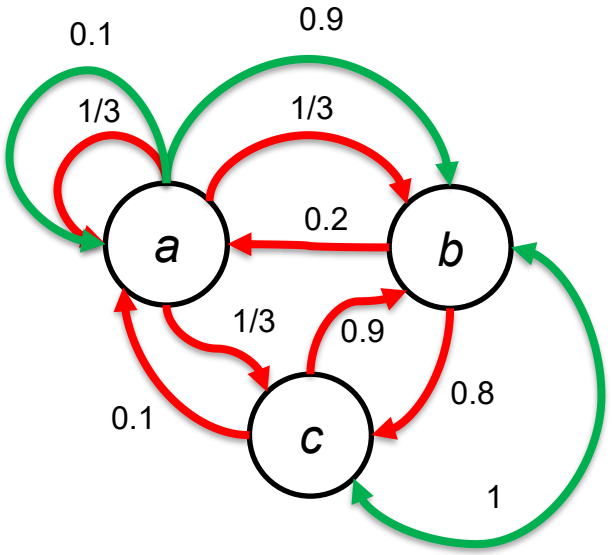
A full MDP also specifies rewards/costs and an objective.

Each action u selects a matrix $D(u) \in \mathbb{R}^{|Q| \times |Q|}$; $D_{ij}(u)$ is the probability $i \to j$.

For time-invariant transitions: $p_D(x' | x, u)$.

x': next state; x: current state; u: applied action.





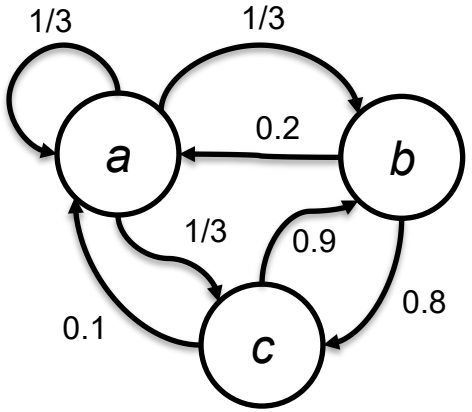
Evolution and measurement: probabilistic models

General motion can depend on the state, observations and input history:

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

A **complete** state summarizes what the motion model needs.

But in practice, we often use a simpler Markovian model.

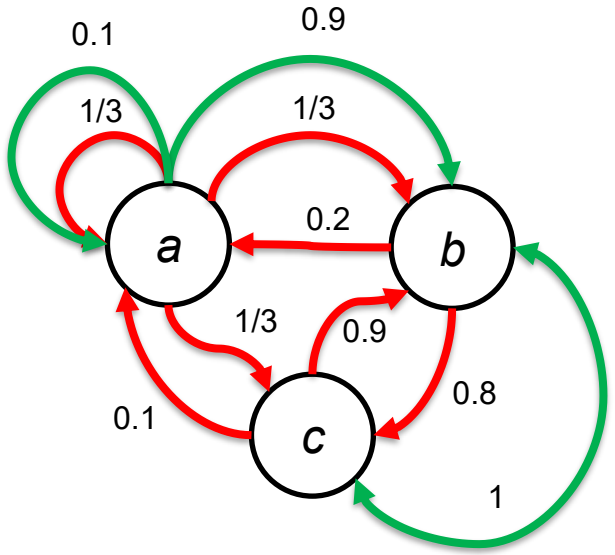


Controlled Markov assumption:

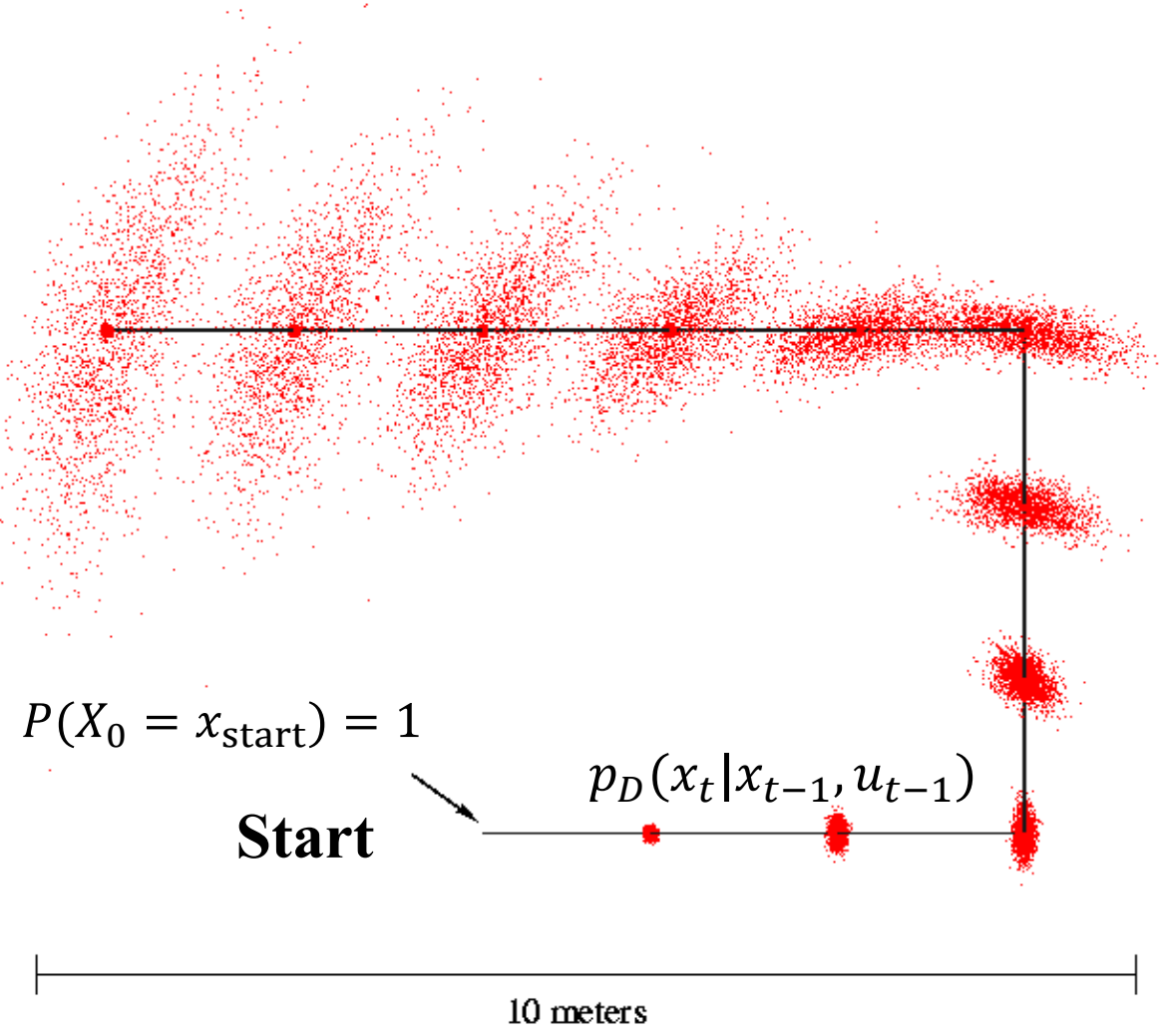
$p_D(x_t|x_{0:t-1}, z_{1:t-1}, u_{0:t-1}) = p_D(x_t|x_{t-1}, u_{t-1})$.

If also time-invariant, write $p_D(x'|x, u)$.

x': next-state value; x: current-state value; u: applied input.



Example Motion Model without measurements



One **example** of a noisy state model:

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

$\varepsilon_t \sim N(0, Q_t)$: zero-mean **Gaussian** noise vector.

Q_t: covariance (spread of motion errors).

Without measurements, motion uncertainty accumulates.

Probabilistic measurements

A **measurement model** gives output probabilities for a candidate state.

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

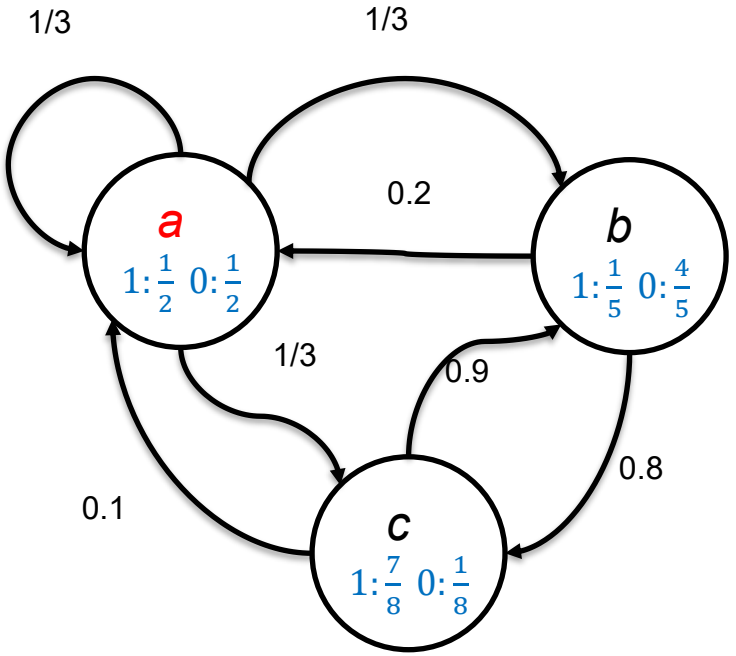
General history: $p_M(z_t | x_{0:t}, z_{1:t-1}, u_{0:t-1})$.

Memoryless measurement assumption:

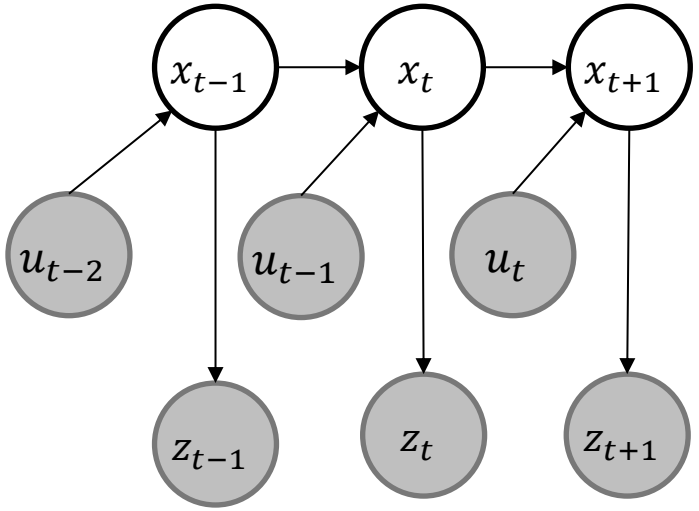
$p_M(z_t | x_{0:t}, z_{1:t-1}, u_{0:t-1}) = p_M(z_t | x_t)$.

$p_M(z_t | x_t)$: measurement probability (given map m understood).

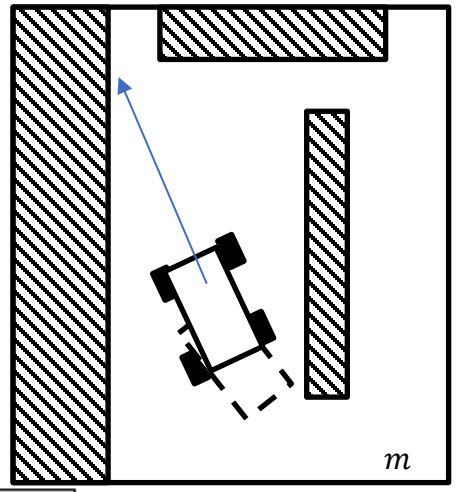
$p_M(z | x)$ if the model is time-invariant.



State a produces output 1 and 0 each with probability 0.5

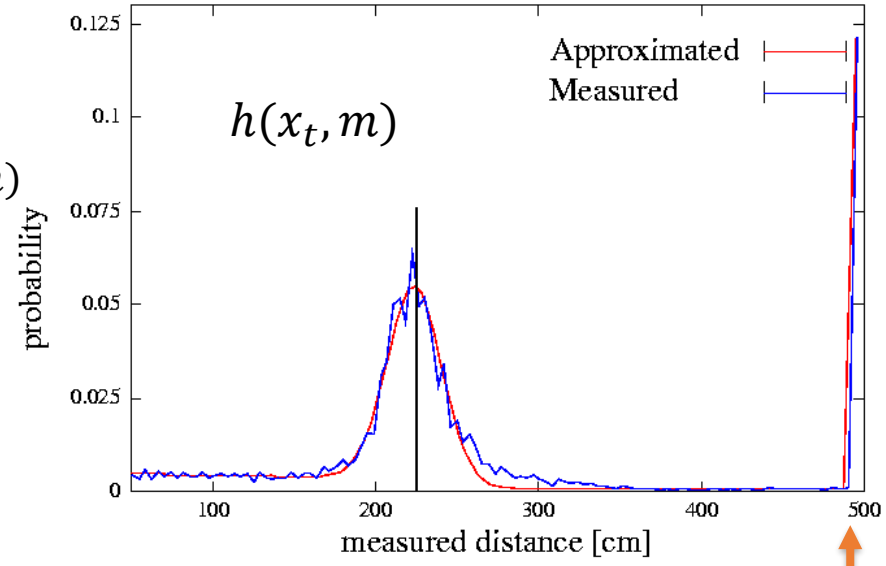


Example Proximity Sensor Measurement Models



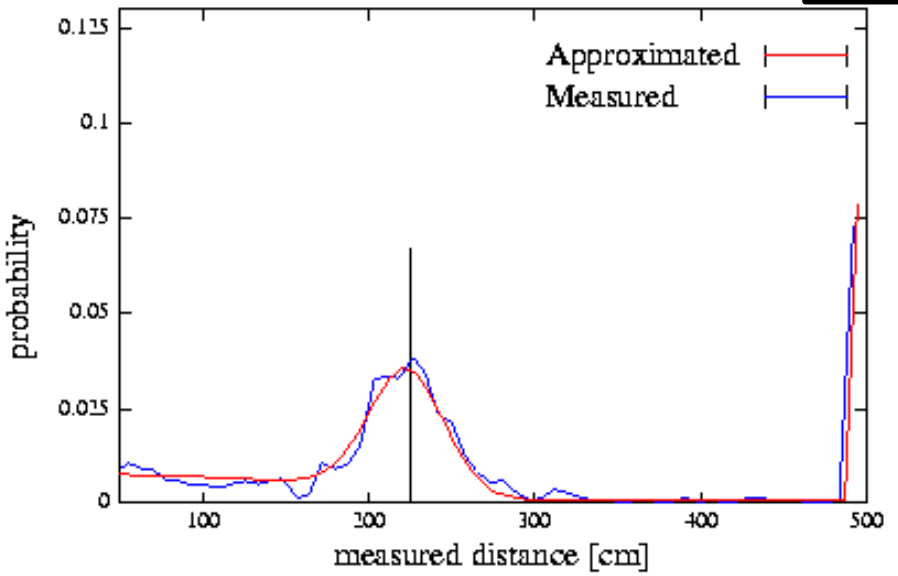
Peak: predicted range with noise. Short returns: nearer objects. Max-range spike: no valid return.

$p_M(z_t|x_t, m)$



Laser sensor

max-range spike
(sensor failure)



Sonar sensor

Summary so far: Evolution and measurement

Motion: $p_D(x_t|x_{0:t-1}, z_{1:t-1}, u_{0:t-1})$.

Controlled Markov assumption:

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

Time-invariant form: $p_D(x'|x, u)$.

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

Memoryless measurement assumption:

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

Time-invariant form: $p_M(z|x)$; given map m understood.

Next lecture: combine motion, measurement and the previous belief.

