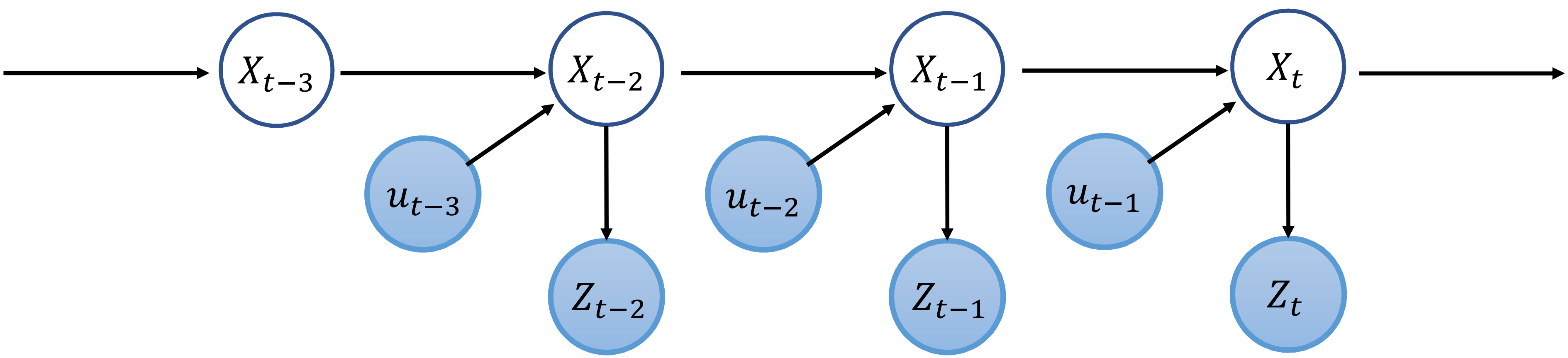
EECE 5550: Mobile Robotics

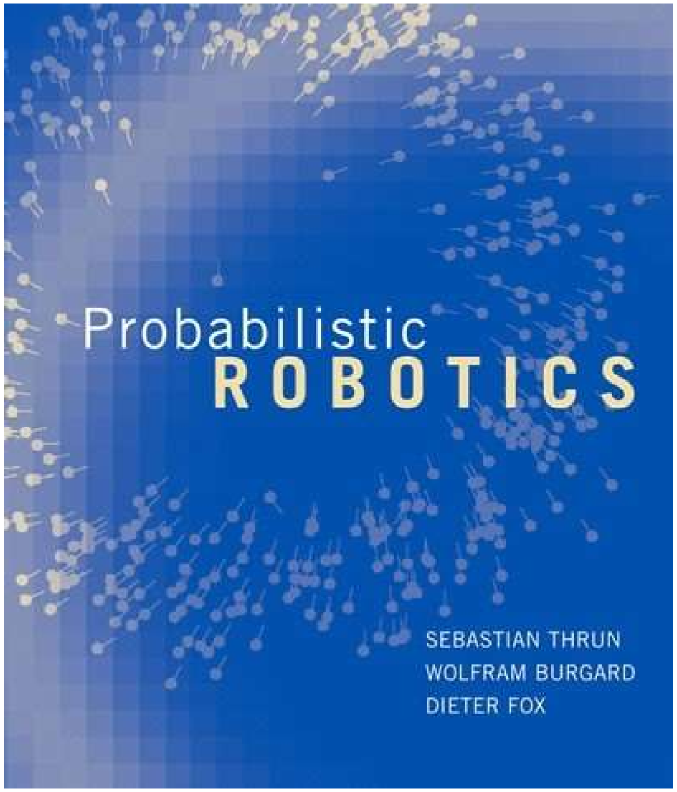


Lecture 9: Probabilistic Robotics

Plan of the day

- Uncertainty in robotics
- Probabilistic robotics
- Bayesian networks
- State estimation and dynamic Bayes nets
- Recursive Bayesian estimation & the Bayes Filter

References



Chapters 1 & 2 of "Probabilistic Robotics"

Motivation: The Problem of Uncertainty

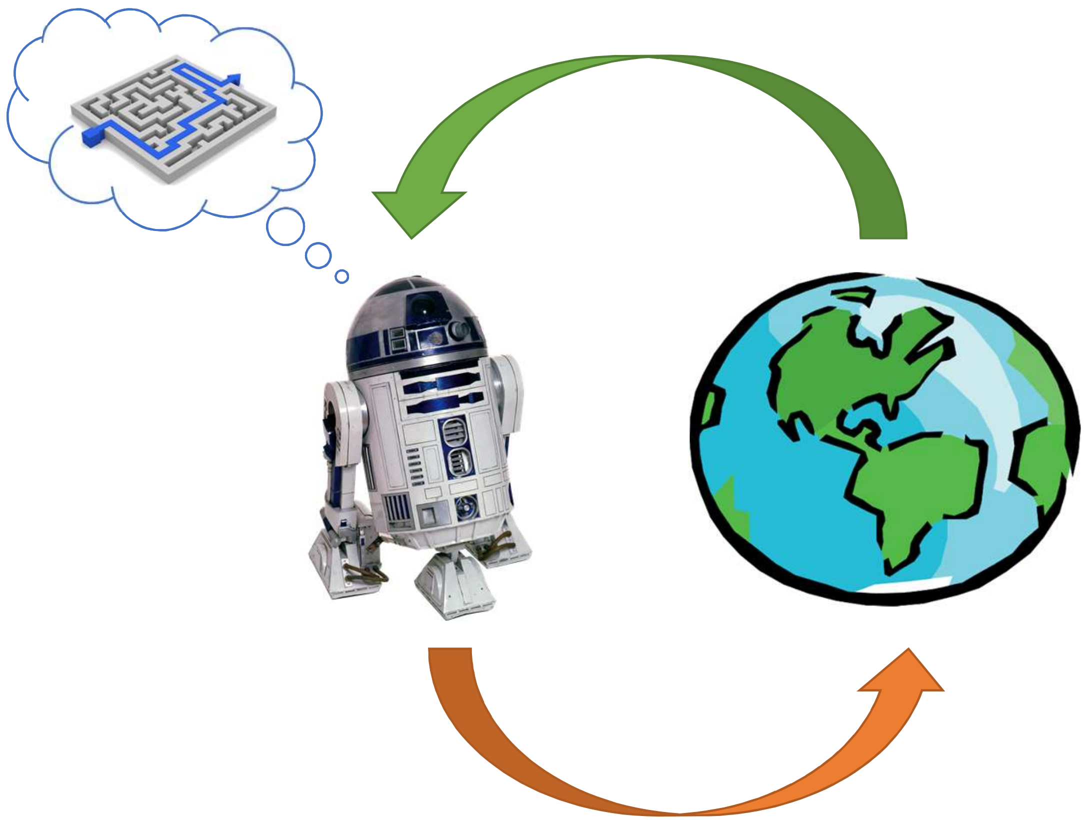
The Goal: Think -> Sense -> Act

The Problem: Uncertainty

What happens if:

- Our models of the world
- Our models of the robot

are *wrong*?



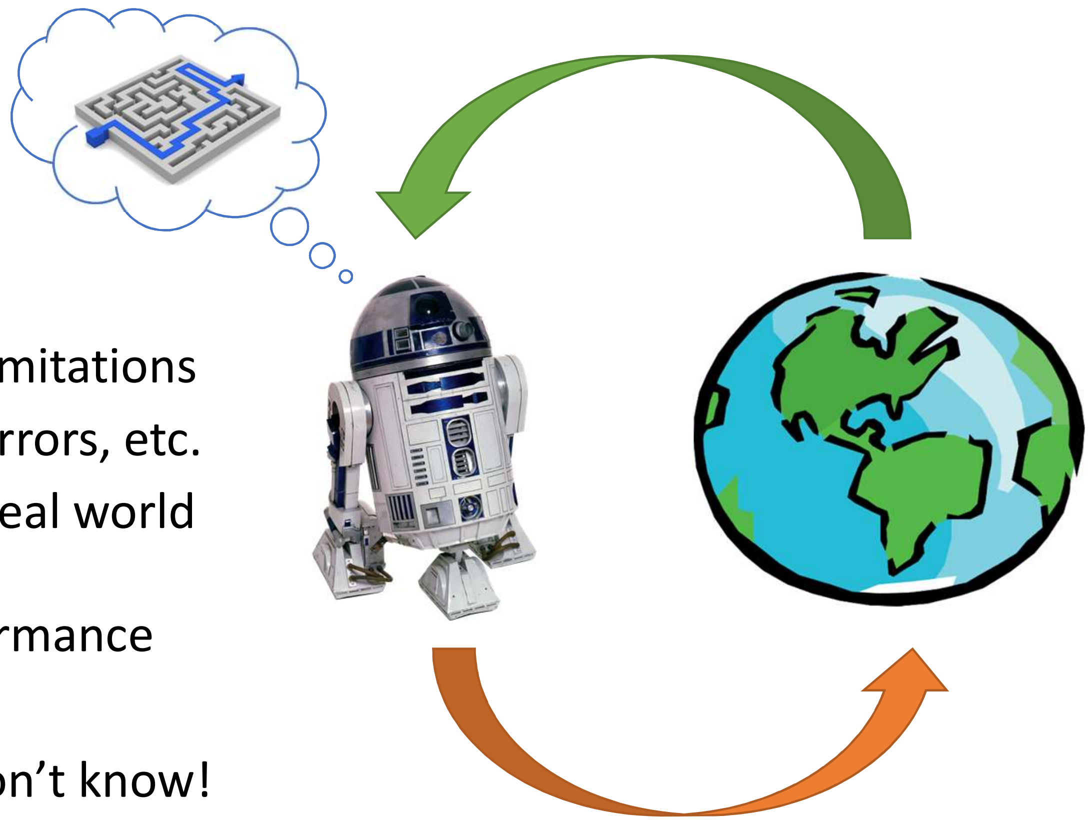
Motivation: The Problem of Uncertainty

The Goal: Think -> Sense -> Act

The Problem: Uncertainty

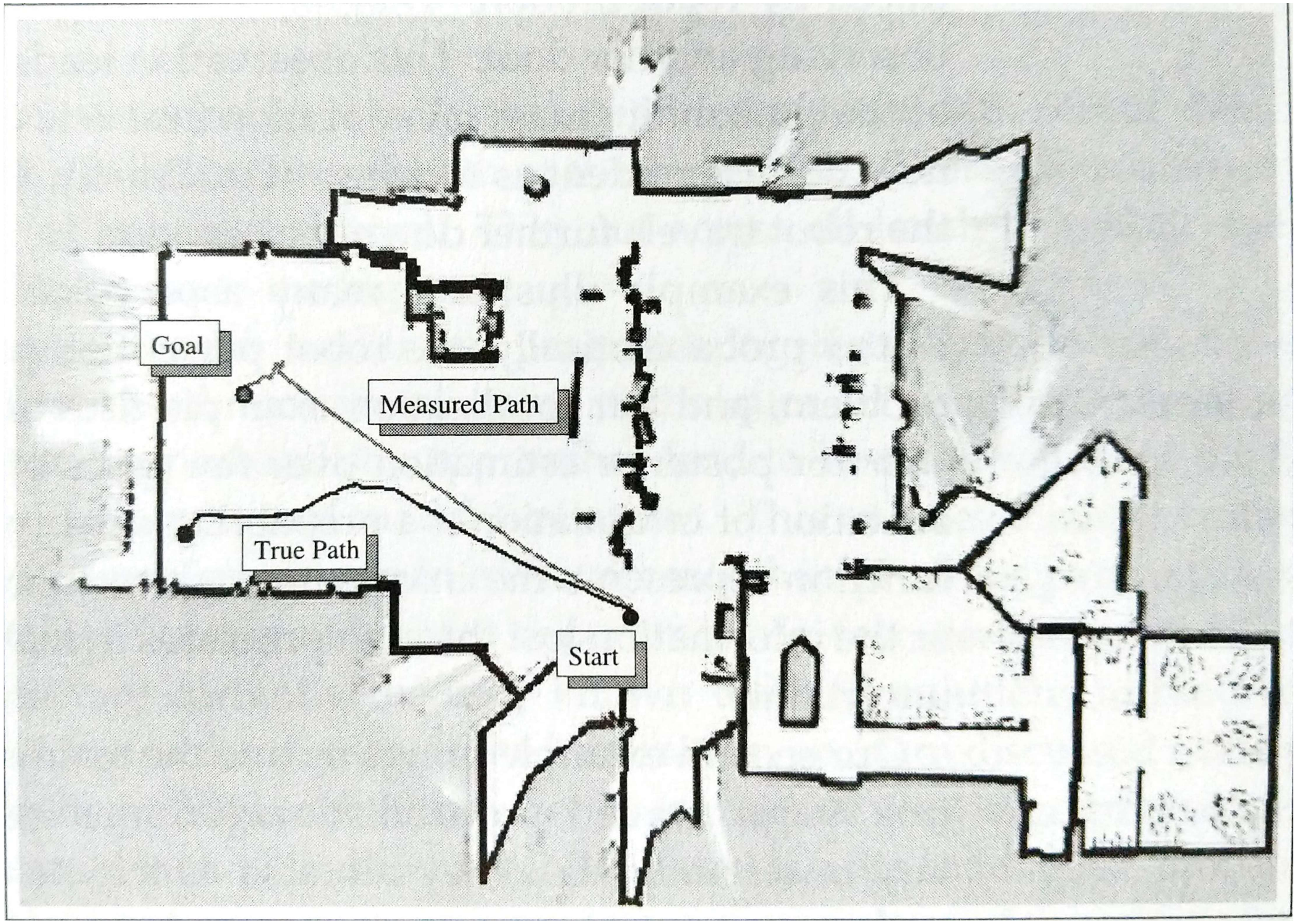
Sources of uncertainty:

- **Sensing**: noise, finite resolution, physical limitations
- **Actuation**: process noise, manufacturing errors, etc.
- **Modeling:** Models only *approximate* the real world
- **Algorithmics**: We often *use algorithmic approximations* to achieve real-time performance



Main takeaway: There is a **lot** of stuff we don't know!

Example: The effects of noise in robot navigation

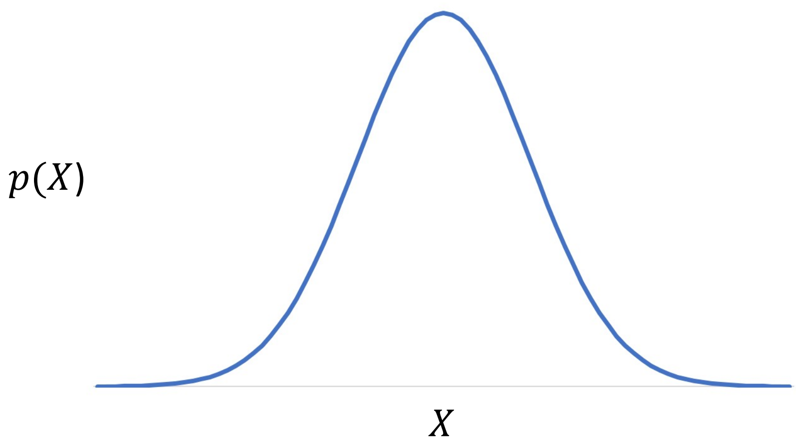


Probabilistic robotics to the rescue!

Main idea: *Explicitly represent* our degree of (un)certainty about the world using the tools of (Bayesian) *probability*

Central object of study: *Belief*

⇒A probability distribution that models our uncertainty over possible states *X* of the world



Rev. Thomas Bayes

A simple example

Suppose I roll a fair die, but don't tell you the result *X*

Q1: What should your belief be about *X*?

Q2: Suppose that now I look at *X*, and tell you that its value is even. What should your belief be now?

A simple example

Suppose I roll a fair die, but don't tell you the result *X*

Q1: What should your belief be about *X*?

A1: Prior: p(X = x) = 1/6 for all x ϵ {1, …, 6}

Q2: Suppose that now I look at *X*, and tell you that its value is even. What should your belief be now?

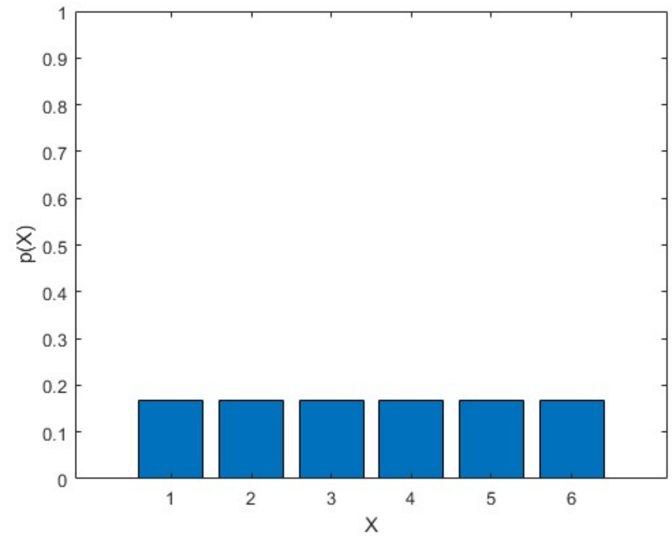
A2: Posterior (conditional):

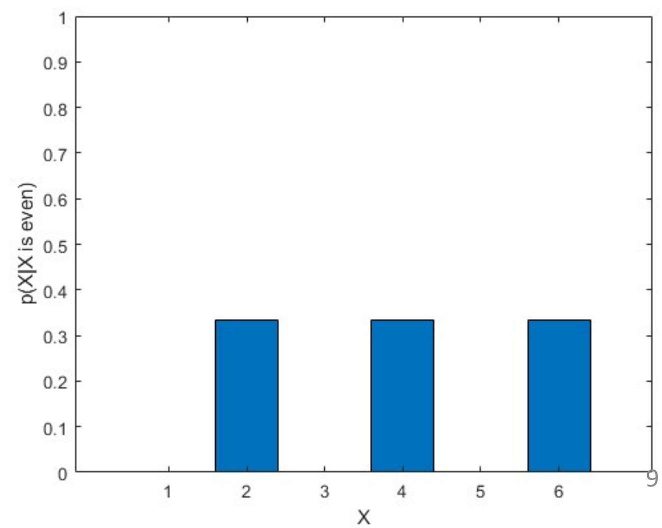
$$p(X = x \mid X \in \{2,4,6\}) = \frac{p(X = x)}{p(\{2,4,6\})} = 1/3$$

for x ϵ {2, 4, 6}

Key point: In this example, it is *not the world* that is changing, but rather *our information about* the world!

Remember: Beliefs model our *state of knowledge* of the world





Why probabilistic robotics?

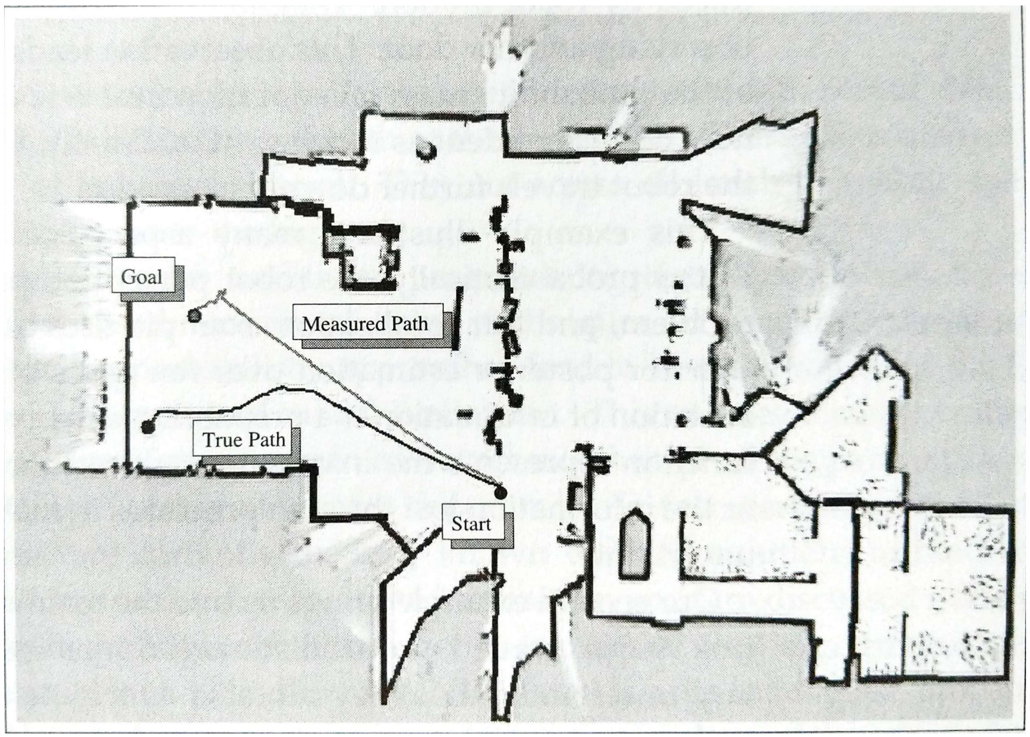
Three main reasons:

- **Correctness:** Accounts for the *actual uncertainty* in sensing, actuation, etc.
  ⇒Probabilistic algorithms degrade gracefully in the real world

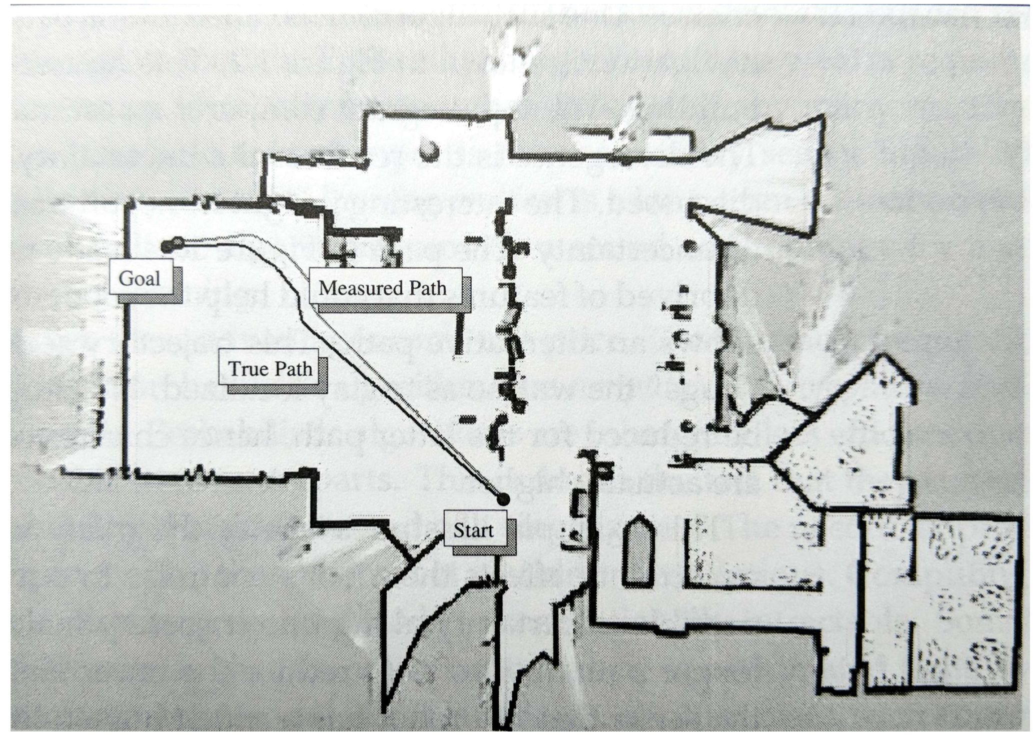
- **Optimality**: Laws of probability tell us exactly how to *optimally combine information* to obtain the *best possible* estimate of the true state of the world

- **Introspection**: Bayesian probabilistic models let us *reason about what we do or don't know*
  ⇒ We can plan actions that will *reduce uncertainty*, if necessary

Example robotics application: localization



Dead reckoning



Coastal navigation

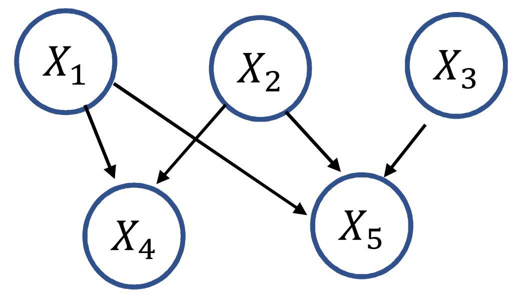
Bayesian networks

A Bayesian network is a directed acyclic graph (DAG) that models the factorization of a probability distribution $p(X_1, \dots, X_n)$ as a product of *conditional* distributions

Definition: A DAG $G = (V, E)$ is a Bayesian network associated to a probability distribution $p(X_1, \dots, X_n)$ if *p* admits a factorization of the form:

$$p(X_1, \dots, X_n) = \prod_{i=1}^{n} p(X_i | pa(X_i))$$

Example



$$p(X) = p(X_1)p(X_2)p(X_3)p(X_4|X_1, X_2)p(X_5|X_1, X_2, X_3)$$

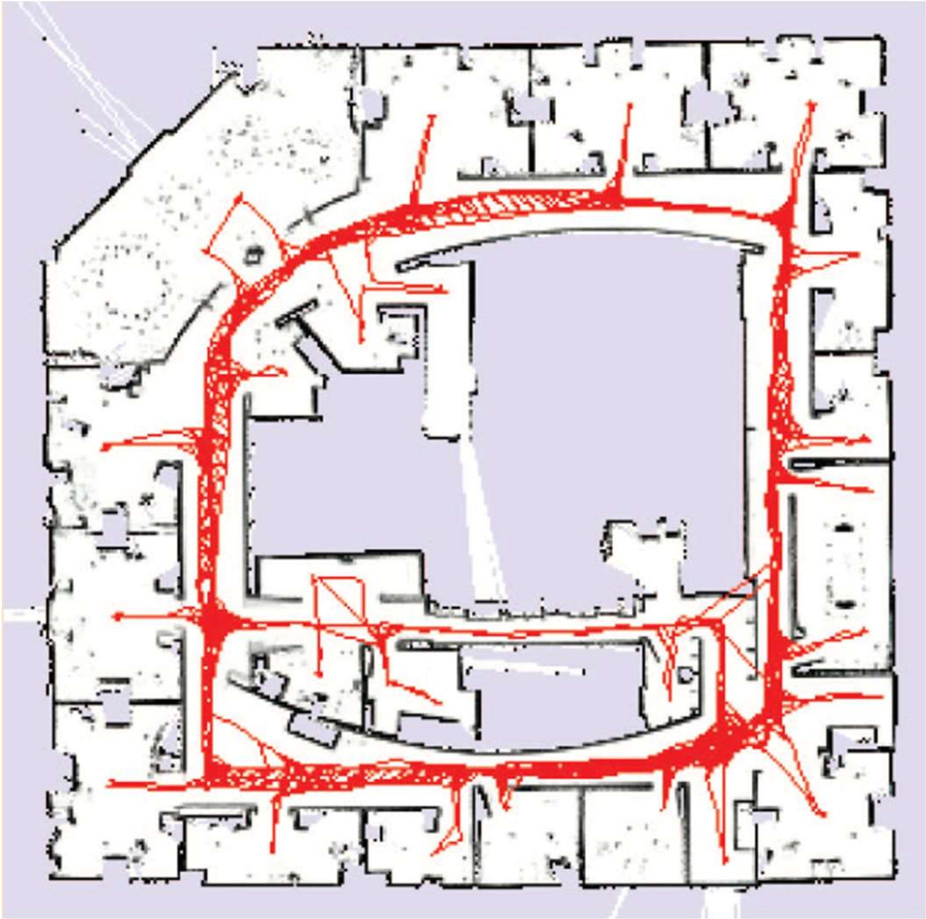
Payoffs:

- **Modeling:** Bayes nets are a convenient language for building complex joint distributions from simple parts
- **Local Markov property:** Bayes nets directly reveal conditional independence among the variables:

$$X_i \perp\!\!\!\perp X_{V - de(X_i)} \mid pa(X_i)$$

- **Computation**:
  - Factorization $\Rightarrow$ more compact storage
  - Exploiting conditional independence is essential for fast inference

Example: Robotic mapping

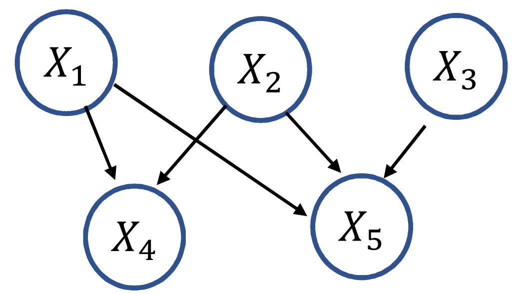


Map constructed in the **Intel Research Lab**, Seattle. This map contains 1,728 individual robot poses / scans
⇒5184-dimensional joint distribution over robot poses

Bayesian networks as generative models

Bayesian networks provide a convenient *generative* description of a probability distribution *p*: that is, they tell us how to *draw samples from it*

Since $G = (V, E)$ is a DAG, its vertices admit a *topological ordering* π
⇒ We can order the vertices such that for each edge $i \to j$,
vertex *i* precedes *j* in the ordering



Example for the graph at right: X_3, X_1, X_2, X_5, X_4

Given this ordering, we can generate a sample $X = (X_1, \dots, X_n)$ from the *joint* distribution by sampling *each element* in topological order

$$p(X_1, \dots, X_n) = \prod_{i=1}^{n} p(X_i | pa(X_i))$$

Algorithm: For each *i* = 1, …. *n*:

$$X_{\pi(i)} \sim p(X_{\pi(i)} | pa(X_{\pi(i)})$$

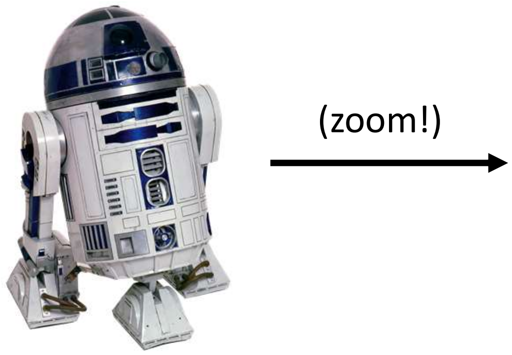
This algorithm is called *ancestral sampling*

Punchline: Bayesian networks are convenient descriptions for simulation

State estimation and dynamic Bayes nets

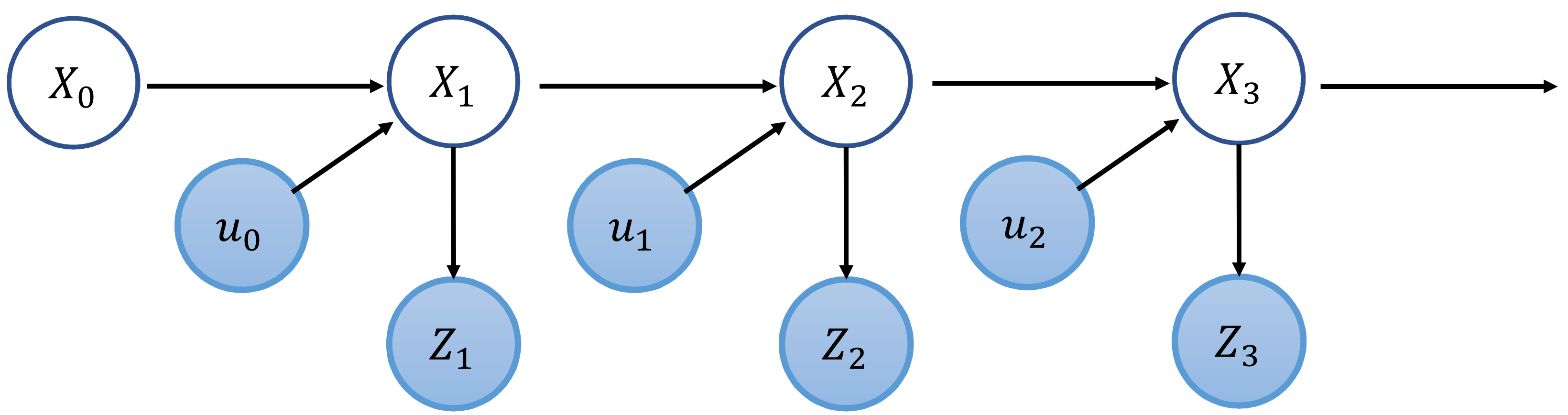
Consider a navigating robot. Starting at some initial position X_0, the robot repeatedly:

- Collects an observation Z_t about its current position X_t
- Applies a control u_t to move to its next position

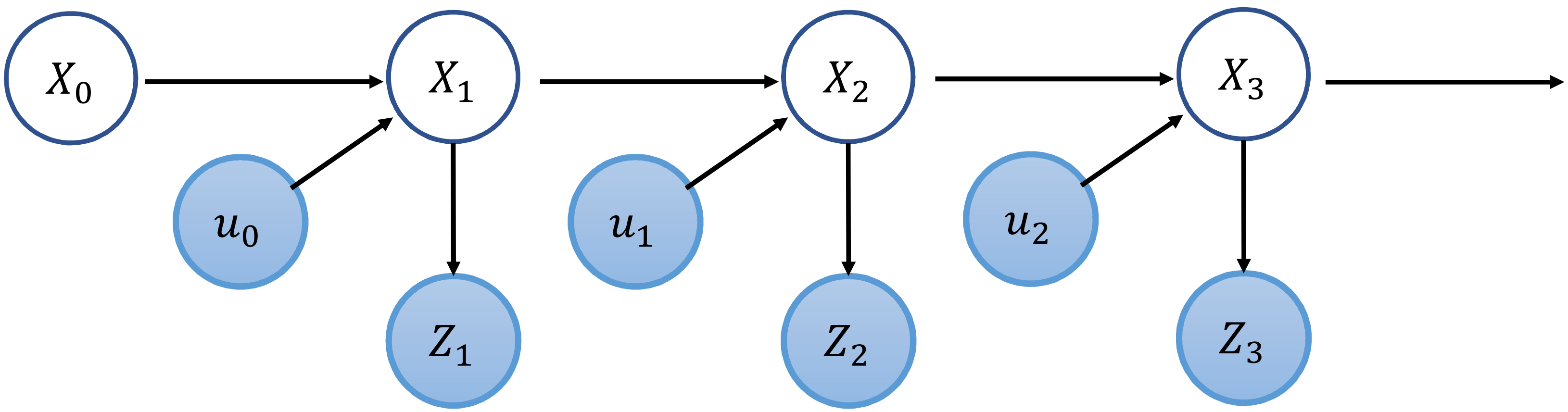


Goal: Estimate a belief $p(X_t | u_{0:t-1}, Z_{1:t})$ over the robot's current position X_t, given all previous controls $u_{0:t-1}$ and observations $Z_{1:t}$

We can model this scenario using a dynamic Bayes net



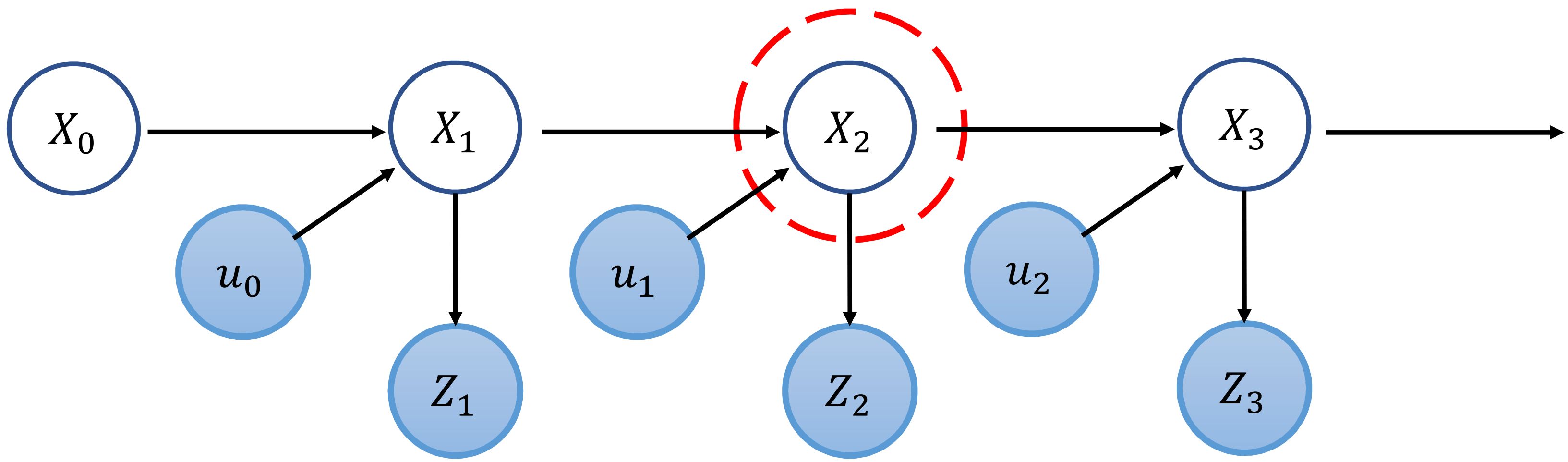
State estimation and dynamic Bayes nets



Problem: The size of this Bayes net is increasing as the robot explores. This is a problem for:

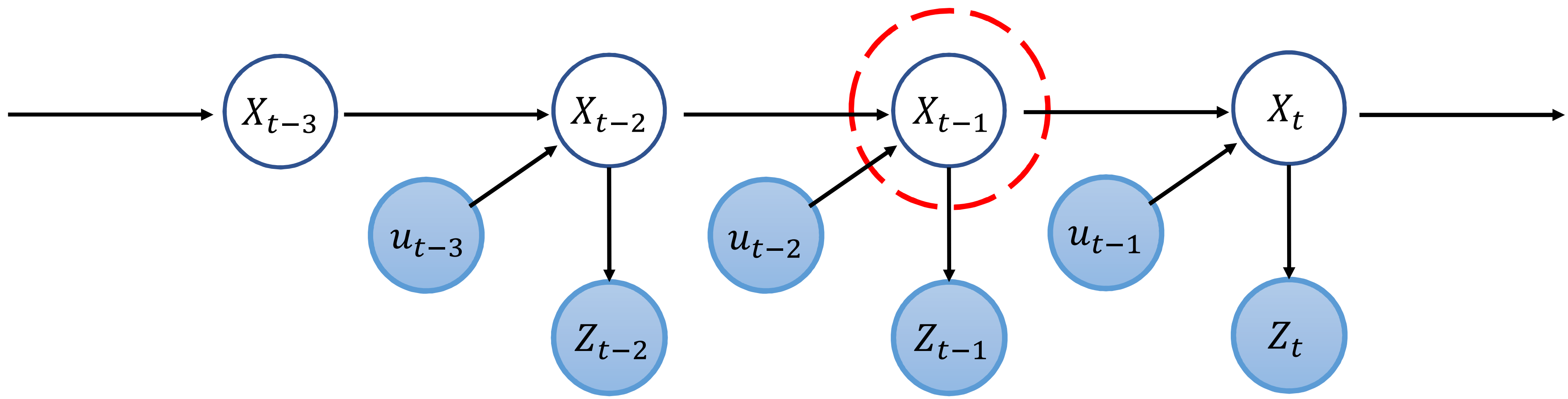
- **Storage**: Need to keep track of more variables, measurements, motor commands
- **Compute**: Need to solve a larger problem to estimate current state

State estimation and dynamic Bayes nets



But: Notice that by the local Markov property, X_3 is conditionally independent of X_0 and X_1 given X_2

State estimation and dynamic Bayes nets



More generally: X_t is conditionally independent of $X_{0:t-2}$ given X_{t-1}.

This suggests that we need only know $p(X_{t-1}|u_{0:t-2}, Z_{1:t-1})$ – the belief over the *previous* position – to compute $p(X_t|u_{0:t-1}, Z_{1:t})$ – the belief at the *current* position

If so, we could discard information about all of the earlier states $X_{1:t-2}$! (This would be a *huge* savings!)

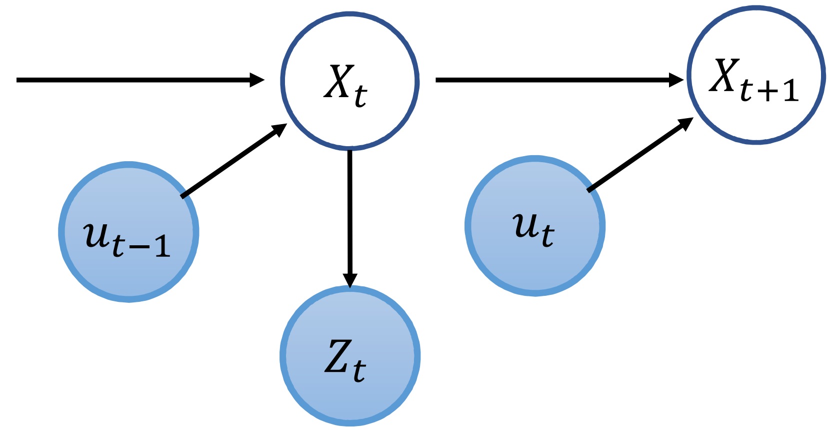
Let's see if we can make this precise …

Recursive Bayesian estimation

Suppose that we *already have* $p(X_t|u_{0:t-1}, Z_{1:t})$, the belief for the robot's current position X_t given all previous controls $u_{0:t-1}$ and measurements $Z_{1:t}$

Now we apply the command u_t to move to the next position.

Q: What is $p(X_{t+1}|u_{0:t}, Z_{1:t})$, the belief for the *next* position X_{t+1} given the control u_t?

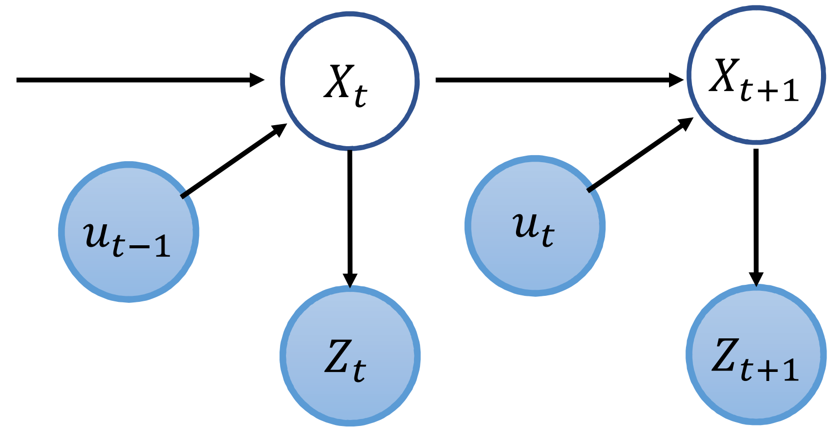


$$\begin{aligned} p(X_{t+1}, X_t|u_{0:t}, Z_{1:t}) &= p(X_{t+1}|X_t, u_{0:t}, Z_{1:t}) \cdot p(X_t|u_{0:t}, Z_{1:t}) \quad &\text{(Chain Rule of probability)} \\ &= \underbrace{p(X_{t+1}|X_t, u_t)}_{\text{Motion model}} \cdot \underbrace{p(X_t|u_{0:t-1}, Z_{1:t})}_{\text{Belief over prior state } X_{t-1}} \quad &\text{(Local Markov property)} \end{aligned}$$

Now we can marginalize over X_t!

$$\begin{aligned} p(X_{t+1}|u_{0:t}, Z_{1:t}) &= \int p(X_{t+1}, X_t|u_{0:t}, Z_{1:t}) \; dX_t \\ &\boxed{= \int p(X_{t+1}|X_t, u_t) \cdot p(X_t|u_{0:t-1}, Z_{1:t}) \; dX_t} \end{aligned}$$

Recursive Bayesian estimation



Now suppose that we take another measurement Z_{t+1} at position X_{t+1}.

Q: What is $p(X_{t+1}|u_{0:t}, Z_{1:t+1})$, the *updated* belief for X_{t+1} after incorporating the latest measurement Z_{t+1}?

$$p(X_{t+1}|u_{0:t}, Z_{1:t+1}) = \frac{p(Z_{t+1}|X_{t+1}, u_{0:t}, Z_{1:t})p(X_{t+1}|u_{0:t}, Z_{1:t})}{p(Z_{t+1}|\, u_{0:t}, Z_{1:t})}$$ (Conditional Bayes' Rule)

Measurement model

Predicted belief for X_{t+1}

$$= \frac{p(Z_{t+1}|X_{t+1})p(X_{t+1}|u_{0:t}, Z_{1:t})}{p(Z_{t+1}|\, u_{0:t}, Z_{1:t})}$$ (Local Markov property)

Evidence calculation:

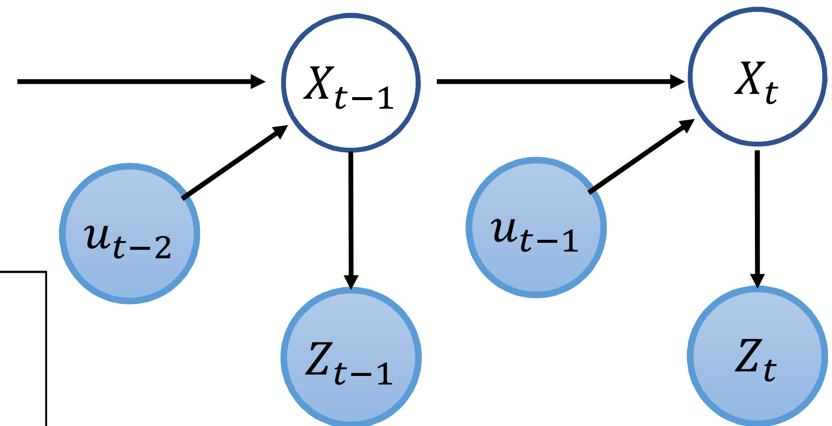
$$p(Z_{t+1}|\, u_{0:t}, Z_{1:t}) = \int p(Z_{t+1}|X_{t+1})p(X_{t+1}|u_{0:t}, Z_{1:t})\ dX_{t+1}$$

The Bayes Filter

The Bayes Filter is a simple and efficient algorithm for recursive Bayesian estimation in dynamic Bayes nets

Given:
- Prior $p(X_0)$ for the initial state X_0
- Sequence of controls $u_{0:t-1}$ and sensor observations $Z_{1:t}$

Find: $p(X_t|u_{0:t-1}, Z_{1:t})$, the posterior belief over the *current* state X_t



Bayes Filter: For t = 1, 2 … repeat the following operations:

- **Predict** belief for current state X_t given previous control u_{t-1}:

$$p(X_t|u_{0:t-1}, Z_{1:t-1}) = \int p(X_t|X_{t-1}, u_{t-1}) \cdot p(X_{t-1}|u_{0:t-2}, Z_{1:t-1})\ dX_t$$

- **Update** belief after incorporating measurement Z_t at current state X_t:

$$p(X_t|u_{0:t-1}, Z_{1:t}) = \frac{p(Z_t|X_t)p(X_t|u_{0:t-1}, Z_{1:t-1})}{\int p(Z_t|X_t)p(X_t|u_{0:t-1}, Z_{1:t-1})\ dX_t}$$

Why the Bayes Filter is Super Cool

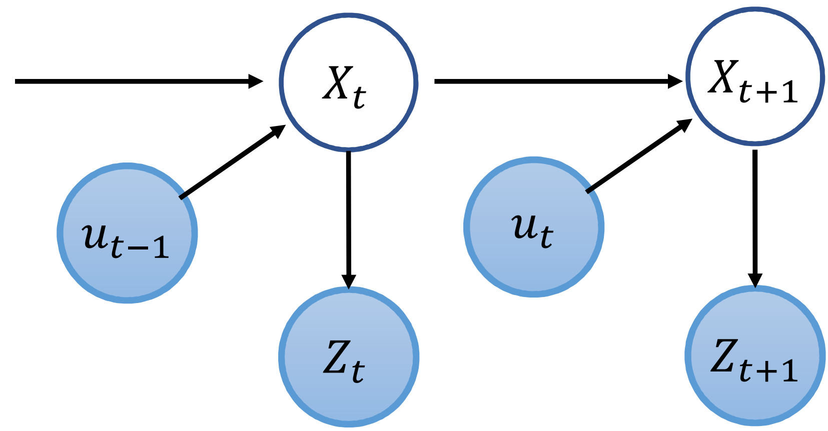
Bayes Filter: For t = 1, 2 … repeat the following operations:

- **Predict** belief for current state X_t given previous control u_{t-1}:

$$p(X_t|u_{0:t-1}, Z_{1:t-1}) = \int p(X_t|X_{t-1}, u_{t-1}) \cdot p(X_{t-1}|u_{0:t-2}, Z_{1:t-1})\ dX_t$$

- **Update** belief after incorporating measurement Z_t at current state X_t:

$$p(X_t|u_{0:t-1}, Z_{1:t}) = \frac{p(Z_t|X_t)p(X_t|u_{0:t-1}, Z_{1:t-1})}{\int p(Z_t|X_t)p(X_t|u_{0:t-1}, Z_{1:t-1})\ dX_t}$$

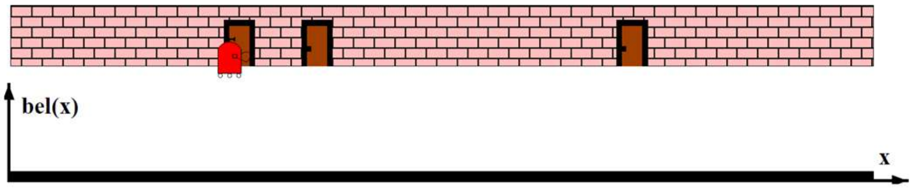


Computational efficiency: While the total number of states, controls, and measurements in the dynamic Bayes net *grows linearly* in time, the Bayes Filter is

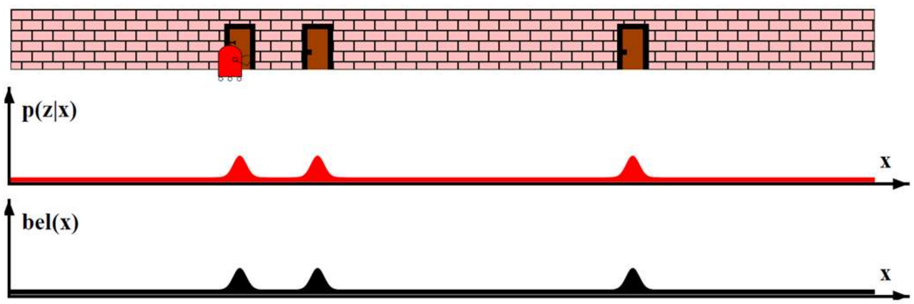
- **Constant space:**
  - The Bayes Filter only maintains a belief over the current state X_t – this is a distribution of a *fixed size*
  - The control u_t and measurement Z_t are only used for prediction and updating in timestep *t* (i.e. a *single step*)
    ⇒We don't need to remember these after they're applied!

- **Constant time:** Each prediction and update step involves computing integrals over distributions of a *fixed size and type*
  ⇒ These are *constant-time* operations. This is super important for (real-time!) robotics applications

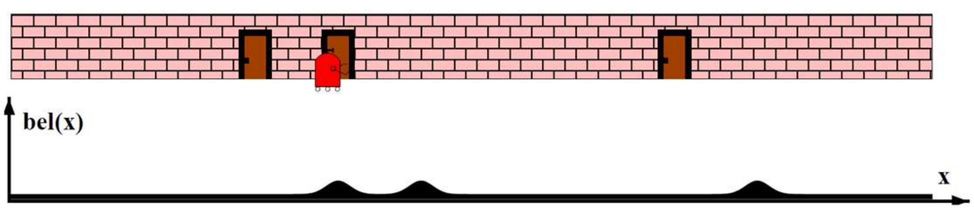
Example application: Robot localization



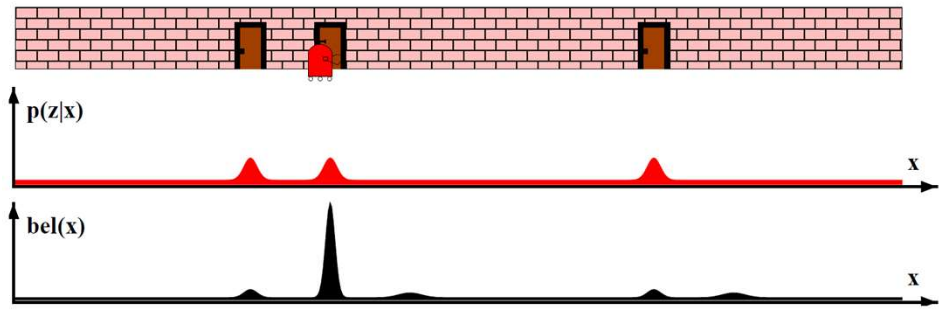
Initial belief



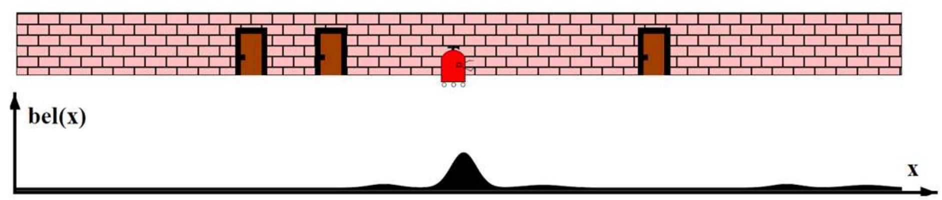
Measurement



Move right



Measurement



Move right

Summary

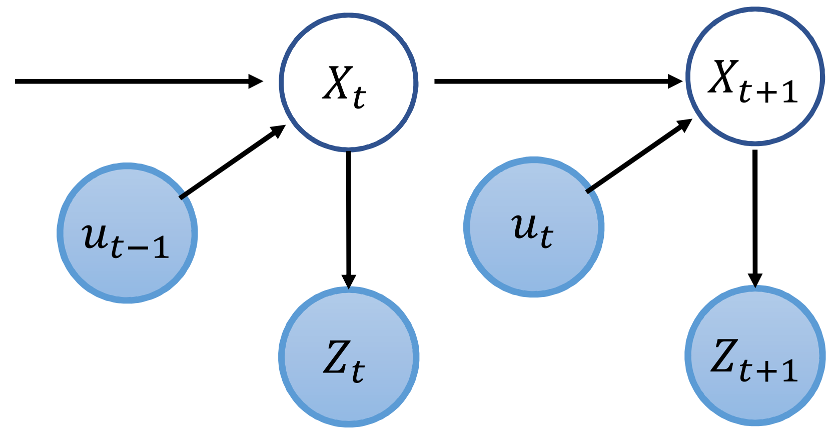
Bayes Filter: For t = 1, 2 … repeat the following operations:

- **Predict** belief for current state X_t given previous control u_{t-1}:

$$p(X_t|u_{0:t-1}, Z_{1:t-1}) = \int p(X_t|X_{t-1}, u_{t-1}) \cdot p(X_{t-1}|u_{0:t-2}, Z_{1:t-1}) \; dX_t$$

- **Update** belief after incorporating measurement Z_t at current state X_t:

$$p(X_t|u_{0:t-1}, Z_{1:t}) = \frac{p(Z_t|X_t)p(X_t|u_{0:t-1}, Z_{1:t-1})}{\int p(Z_t|X_t)p(X_t|u_{0:t-1}, Z_{1:t-1}) \; dX_t}$$



This time:

- Probabilistic robotics
- Bayesian networks
- Recursive Bayesian estimation and the Bayes Filter

Next time: Three specific ways to implement the Bayes Filter