Probabilistic Robotics Course

Dynamic Bayesian Networks (Filtering)

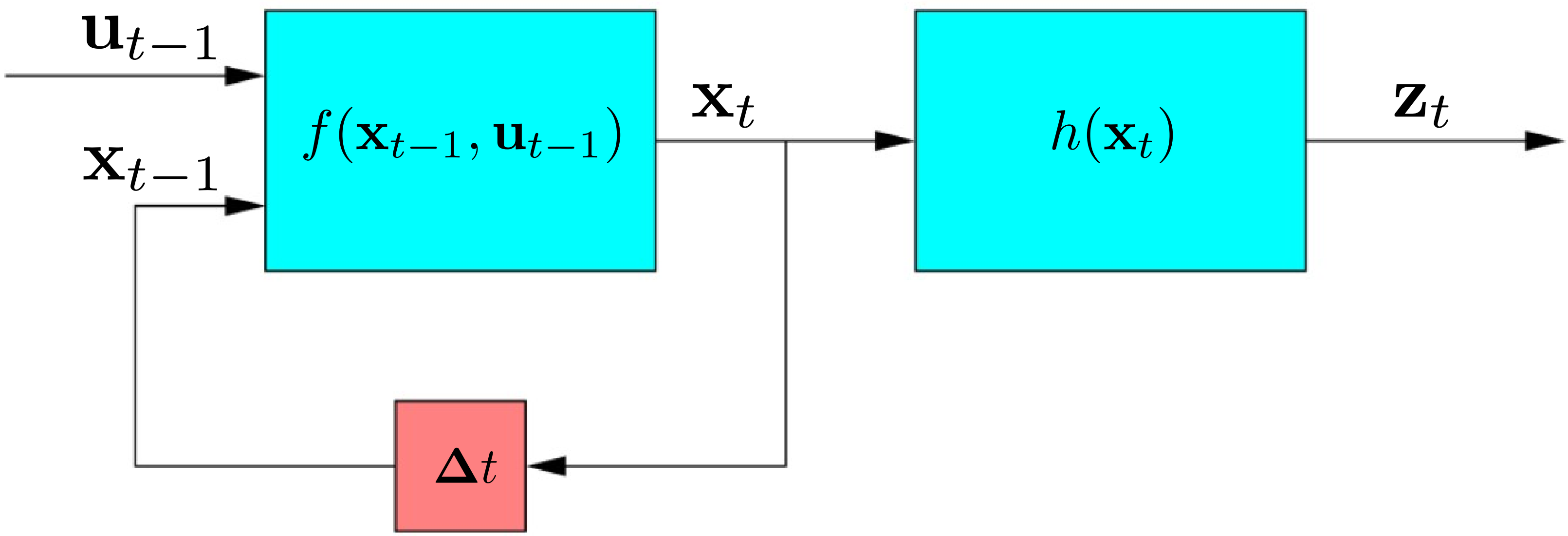
Giorgio Grisetti
grisetti@diag.uniroma1.it

Department of Computer Control and Management Engineering
Sapienza University of Rome

Overview

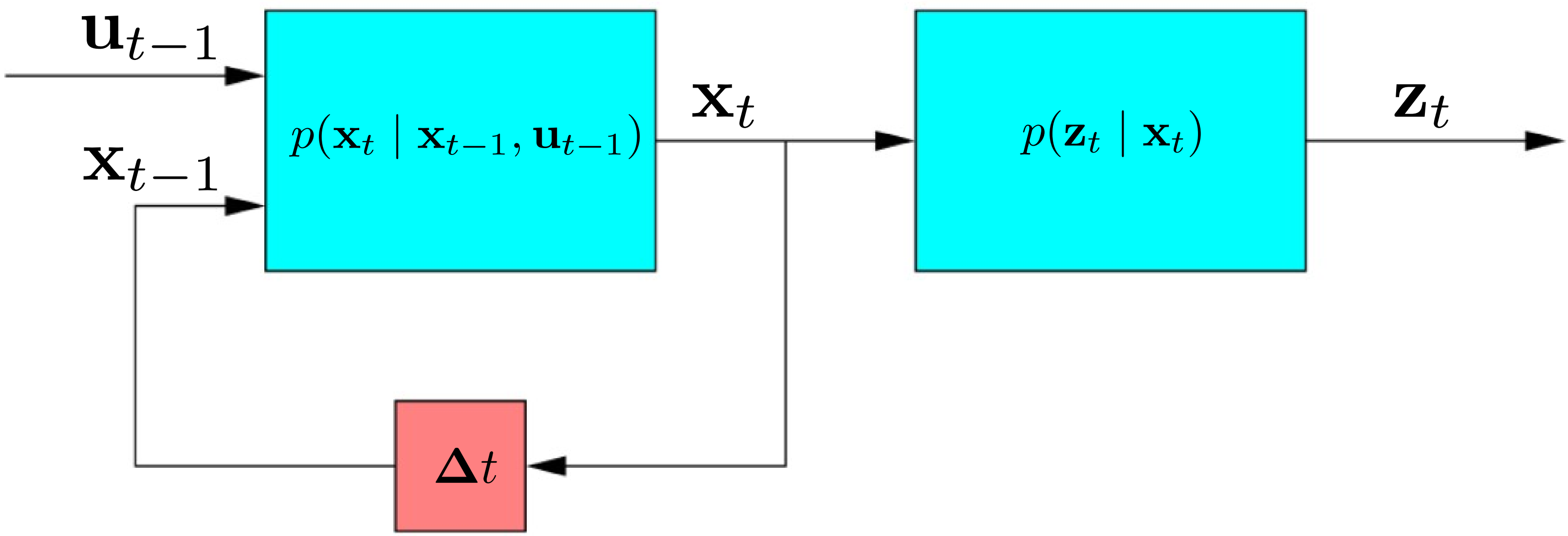
- Probabilistic Dynamic Systems
- Dynamic Bayesian Networks (DBN)
- Inference on DBN
- Recursive Bayes Equation

Dynamic System
Deterministic View



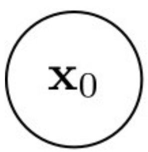
- $f(\mathbf{x}_{t-1}, \mathbf{u}_{t-1})$: transition function
- $h(\mathbf{x}_t)$: observation function
- $\mathbf{x}_{t-1}$: previous state
- $\mathbf{x}_t$: current state
- $\mathbf{u}_{t-1}$: previous control/action
- $\mathbf{z}_t$: current observation
- $\mathbf{\Delta}t$: delay

Dynamic System
Probabilistic View



- $p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{u}_{t-1})$: transition model
- $p(\mathbf{z}_t \mid \mathbf{x}_t)$: observation model
- $\mathbf{x}_{t-1}$: previous state
- $\mathbf{x}_t$: current state
- $\mathbf{u}_{t-1}$: previous control/action
- $\mathbf{z}_t$: current observation
- $\mathbf{\Delta} t$: delay

Evolution of a Dynamic System: State

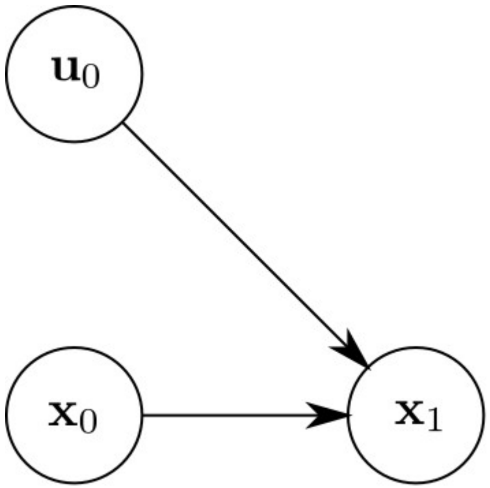


Let's start from a known initial state distribution $p(\mathbf{x}_0)$.

Evolution of a Dynamic System: Control

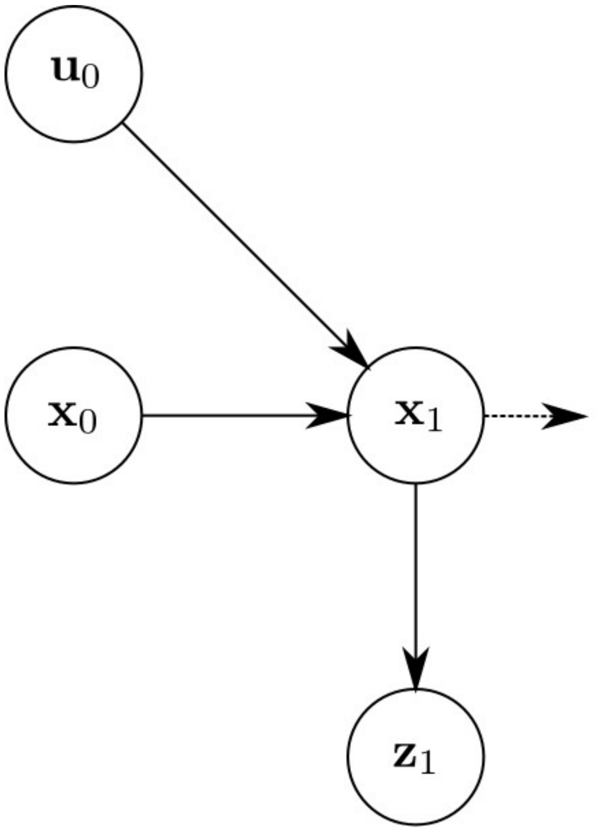
A control $\mathbf{u}_0$ becomes available.

Evolution of a Dynamic System: Transition



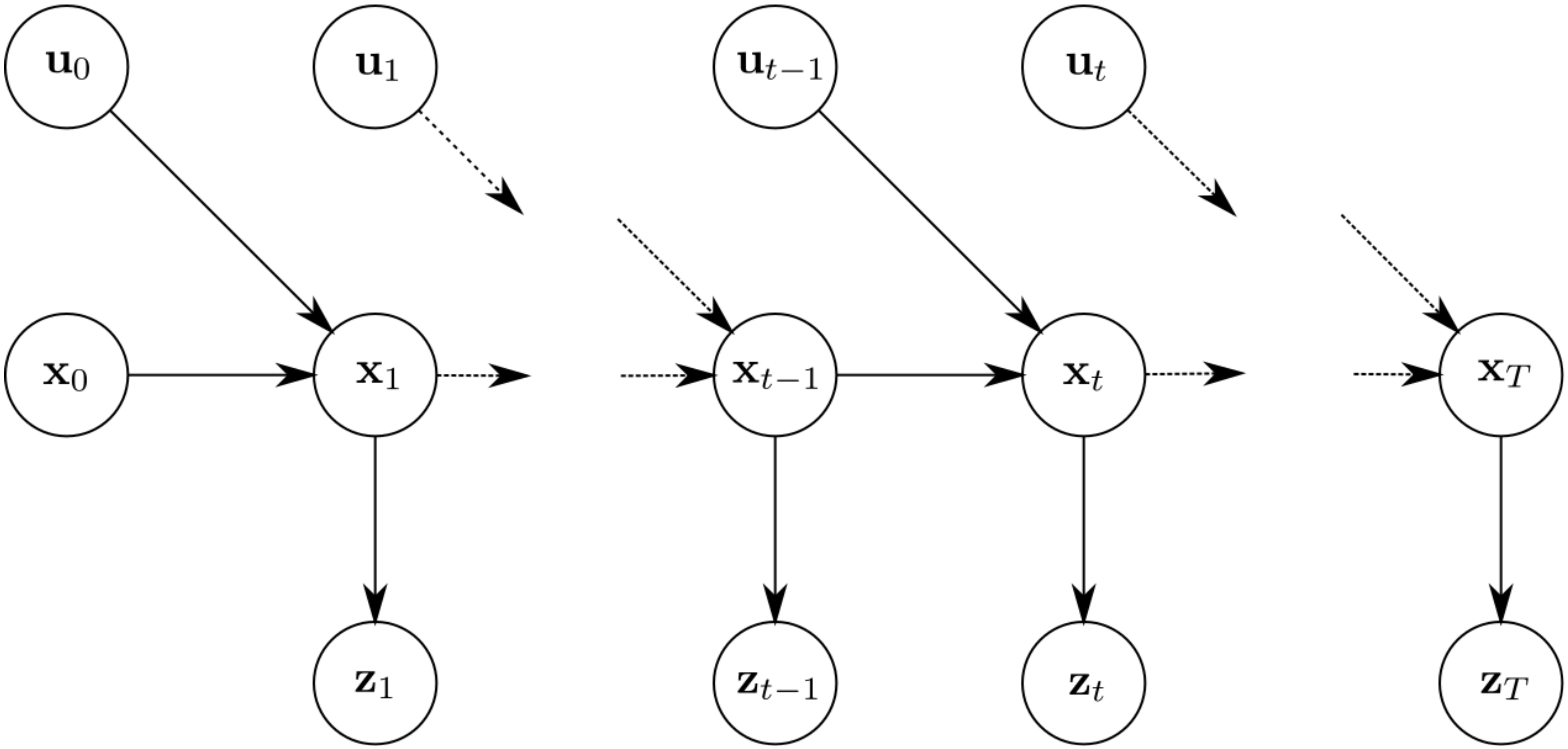
The transition model $p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{u}_{t-1})$ correlates the current state $\mathbf{x}_1$ with the previous control $\mathbf{u}_0$ and the previous state $\mathbf{x}_0$.

Evolution of a Dynamic System: Observation



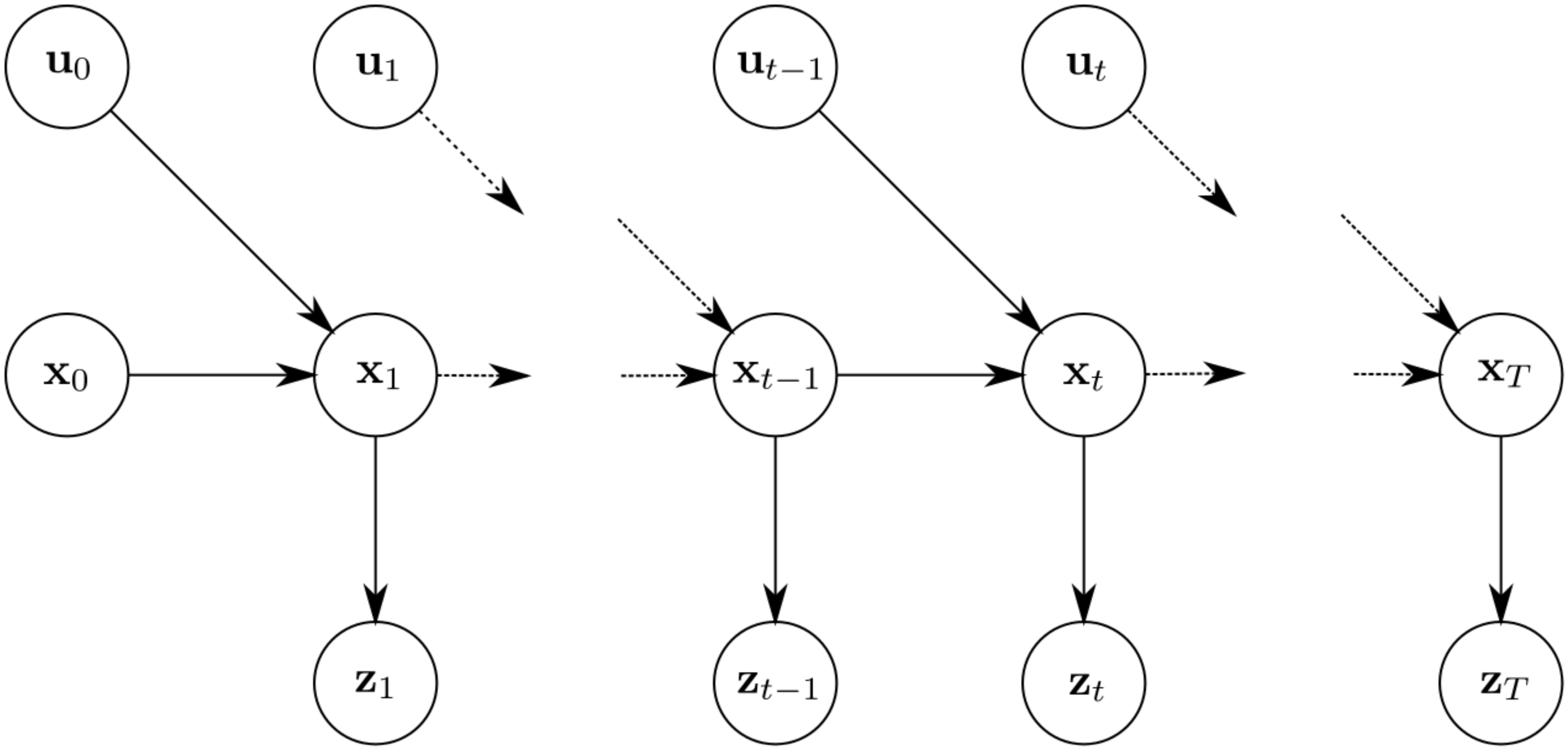
The observation model $p(\mathbf{z}_t \mid \mathbf{x}_t)$ correlates the observation $\mathbf{z}_1$ and the current state $\mathbf{x}_1$.

Evolution of a Dynamic System



This leads to a recurrent structure, that depends on the *time* t.

Dynamic Bayesian Networks (DBN)



- Graphical representations of stochastic dynamic processes
- Characterized by a recurrent structure

States in a DBN

The domain of the states $\mathbf{x}_t$, the controls $\mathbf{u}_t$ and the observations $\mathbf{z}_t$ are not restricted to be boolean or discrete.

Examples:

- Robot localization, with a laser range finder
  - States $\mathbf{x}_t \in SE(2)$, isometries on a plane
  - Observations $\mathbf{z}_t \in \Re^{\#beams}$, laser range measurements
  - Controls $\mathbf{u}_t \in \Re^2$, translational and rotational speed
- HMM (Hidden Markov Model)
  - States $\mathbf{x}_t \in [X_1, \ldots, X_{N_x}]$, finite states
  - Observations $\mathbf{z}_t \in [Z_1, \ldots, Z_{N_z}]$, finite observations
  - Controls $\mathbf{u}_t \in [U_1, \ldots, U_{N_u}]$, finite controls

Inference in a DBN requires to design a data structure that can represent a *distribution* over states.

Typical Inferences in a DBN

In a dynamic system, usually[1] we know:

- the observations $\mathbf{z}_{1:T}$ made by the system, because we *measure* them.
- the controls $\mathbf{u}_{0:T-1}$, because we *issue* them

Typical inferences in a DBN:

name	query	known
Filtering	$p(\mathbf{x}_T \mid \mathbf{u}_{0:T-1}, \mathbf{z}_{1:T})$	$\mathbf{u}_{0:T-1}, \mathbf{z}_{1:T}$
Smoothing	$p(\mathbf{x}_t \mid \mathbf{u}_{0:T-1}, \mathbf{z}_{1:T}),\ 0 < t < T$	$\mathbf{u}_{0:T-1}, \mathbf{z}_{1:T}$
Max a Posteriori	$\operatorname{argmax}_{\mathbf{x}_{0:T}}\ p(\mathbf{x}_{0:T} \mid \mathbf{u}_{0:T-1}, \mathbf{z}_{1:T})$	$\mathbf{u}_{0:T-1}, \mathbf{z}_{1:T}$

[1]usually does not mean always

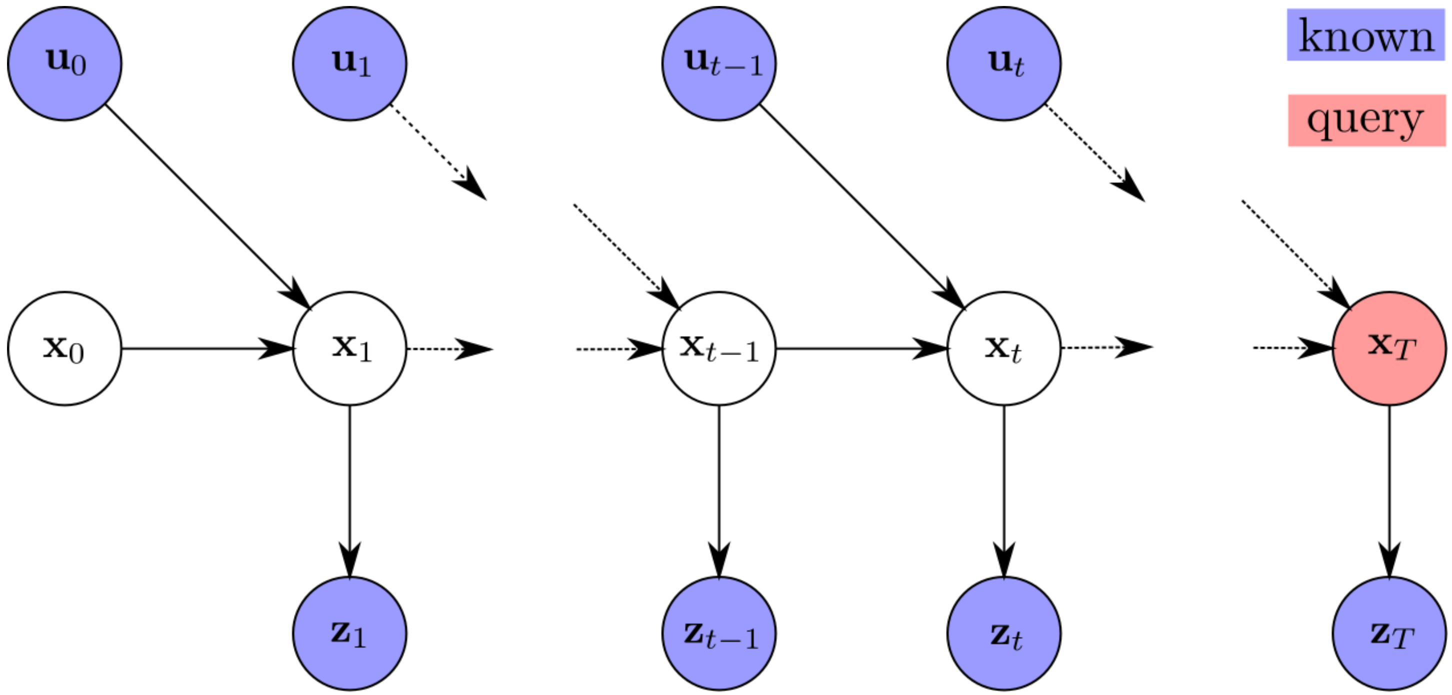
Typical Inferences in a DBN

Using the traditional tools for Bayes Networks is not a good idea:

- too many variables (potentially infinite) render the solution intractable
- the domains are not necessarily discrete

However, we can exploit the recurrent structure to design procedures that take advantage of it

DBN Inference: Filtering

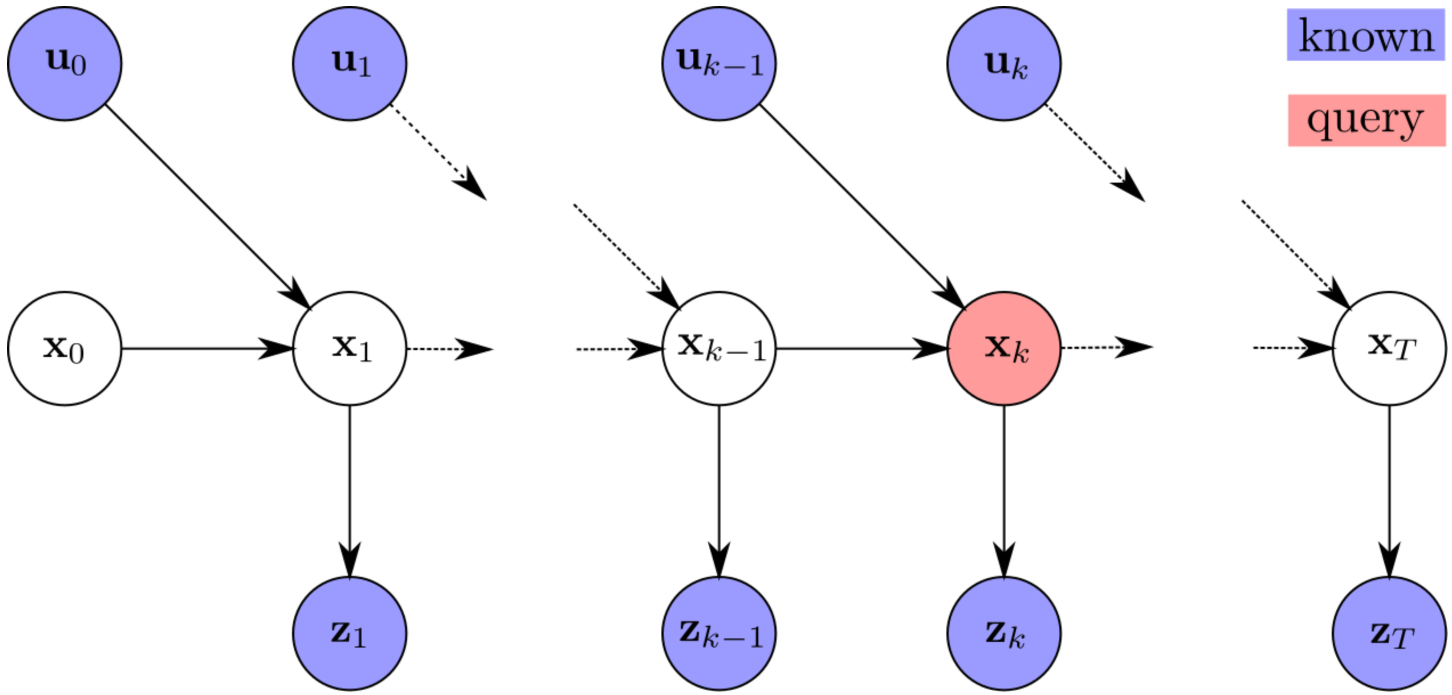


Given:

- the sequence of all observations $\mathbf{z}_{1:T}$ up to the current time T
- the sequence of all controls $\mathbf{u}_{0:T-1}$

we want to compute the distribution over the current state $p(\mathbf{x}_T | \mathbf{u}_{0:T-1}, \mathbf{z}_{1:T})$.

DBN Inference: Smoothing



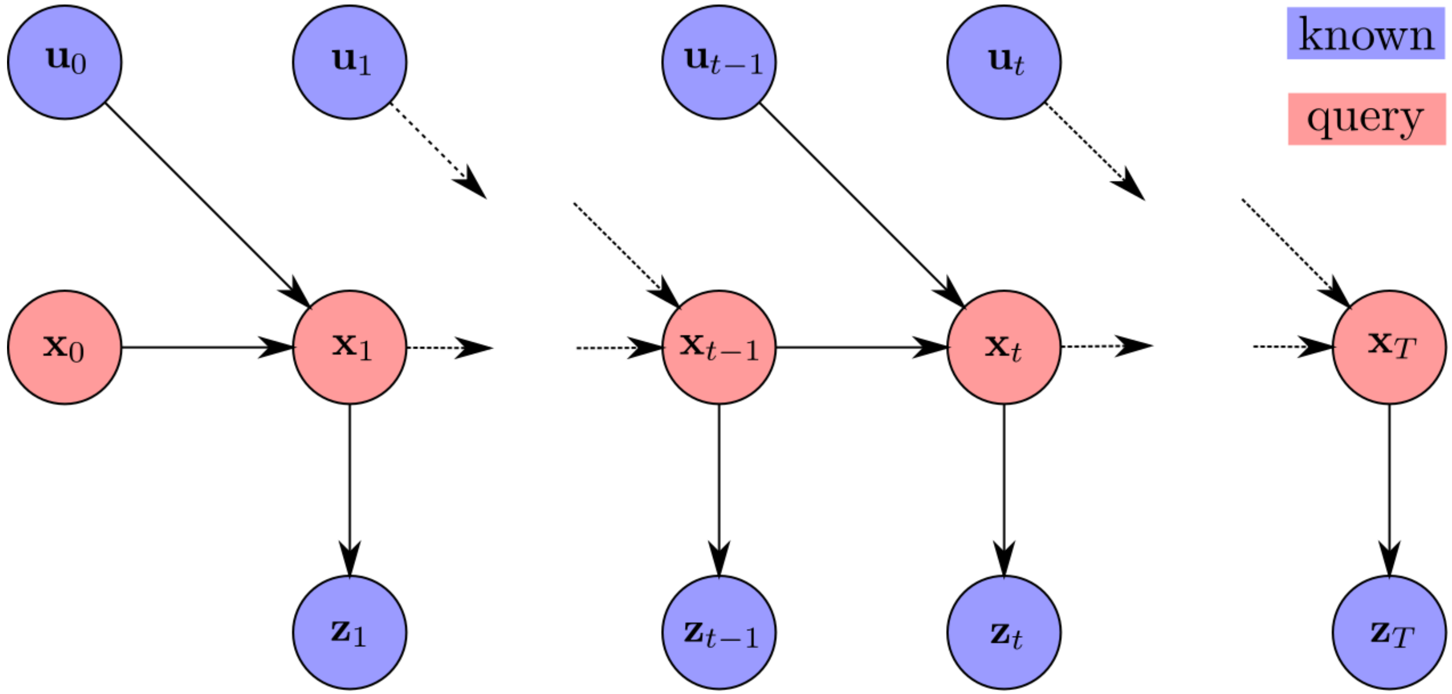
Given:

- the sequence of all observations $\mathbf{z}_{1:T}$ up to the current time T
- the sequence of all controls $\mathbf{u}_{0:T-1}$

we want to compute the distribution over a past state $p(\mathbf{x}_k | \mathbf{u}_{0:T-1}, \mathbf{z}_{1:T})$.

Knowing also the controls $\mathbf{u}_{0:T-1}$ and the observations $\mathbf{z}_{1:T}$ *after* time k, leads to more accurate estimates than pure filtering.

DBN Inference: Maximum a Posteriori



Given:

- the sequence of all observations $\mathbf{z}_{1:T}$ up to the current time T
- the sequence of all controls $\mathbf{u}_{0:T-1}$

we want to find the most likely *trajectory* of states $\mathbf{x}_{0:T}$.

In this case we are not seeking for a distribution.
Just the most likely *sequence*.

DBN Inference: Belief

- Algorithms for performing inference on a DBN keep track of the *estimate* of a distribution of states.
- This distribution should be stored in an appropriate data structure.
- The structure depends on:
  - the knowledge of the characteristics of the distribution (e.g. Gaussian)
  - the domain of the state variables (e.g. continuous vs discrete)

When we write $b(\mathbf{x}_t)$ we mean our current belief of $p(\mathbf{x}_t|...)$

The algorithms for performing inference on a DBN work by updating a belief.

DBN Inference: Belief

- In the simple case of a system with discrete state $\mathbf{x} \in \{X_{1:n}\}$, the belief can be represented through an array $\mathbf{x}$ of float values. Each cell of the array $\mathbf{x}[i] = p(\mathbf{x} = X_i)$ contains the probability of that state

- If our system has a continuous state and we know it is distributed according to a Gaussian, we can represent the belief through its parameters (mean and covariance matrix)

- If the state is continuous but the distribution is unknown, we can use some approximate representation (e.g. weighed samples of state values).

Filtering: Bayes Recursion

We want to compute: $p\left(\mathbf{x}_T \middle| \mathbf{u}_{0:T-1}, \mathbf{z}_{1:T}\right)$

We know:

- the observations $\mathbf{z}_{1:T}$
- the controls $\mathbf{u}_{0:T-1}$
- $p\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{u}_{t-1}\right)$: the transition model. It is a function that, given the previous state $\mathbf{x}_{t-1}$ and control $\mathbf{u}_{t-1}$, tells us how likely it is to land in state $\mathbf{x}_t$.
- $p\left(\mathbf{z}_t \mid \mathbf{x}_t\right)$: the observation model. It is a function, that given the current state $\mathbf{x}_t$, tells us how likely it is to observe $\mathbf{z}_t$.
- $b\left(\mathbf{x}_{t-1}\right)$, which is our belief about the previous state $p\left(\mathbf{x}_{t-1} \mid \mathbf{u}_{0:t-2}, \mathbf{z}_{1:t-1}\right)$

Filtering: Bayes Rule

$$p(\mathbf{x}_T | \mathbf{u}_{0:T-1}, \mathbf{z}_{1:T}) = \tag{1}$$

- splitting z_t:

$$= p(\underbrace{\mathbf{x}_t}_{A} \mid \underbrace{\mathbf{z}_t}_{B}, \underbrace{\mathbf{u}_{0:t-1}, \mathbf{z}_{1:t-1}}_{C}) \tag{2}$$

- recall the conditional Bayes rule $p(A|B,C) = \frac{p(B|A,C)p(A|C)}{p(B|C)}$

$$= \frac{p(\overbrace{\mathbf{z}_t}^{B} \mid \overbrace{\mathbf{x}_t}^{A}, \overbrace{\mathbf{u}_{0:t-1}, \mathbf{z}_{1:t-1}}^{C}) p(\overbrace{\mathbf{x}_t}^{A} | \overbrace{\mathbf{u}_{0:t-1}, \mathbf{z}_{1:t-1}}^{C})}{p(\underbrace{\mathbf{z}_t}_{B} | \underbrace{\mathbf{u}_{0:t-1}, \mathbf{z}_{1:t-1}}_{C})} \tag{3}$$

Filtering: Denominator

- let the denominator

$$\eta_t = 1/p(\mathbf{z}_t \mid \mathbf{u}_{0:t-1}, \mathbf{z}_{1:t-1}) \tag{4}$$

Note that η_t does not depend on the state $\mathbf{x}$, thus to the extent of our computation is just a normalizing constant.

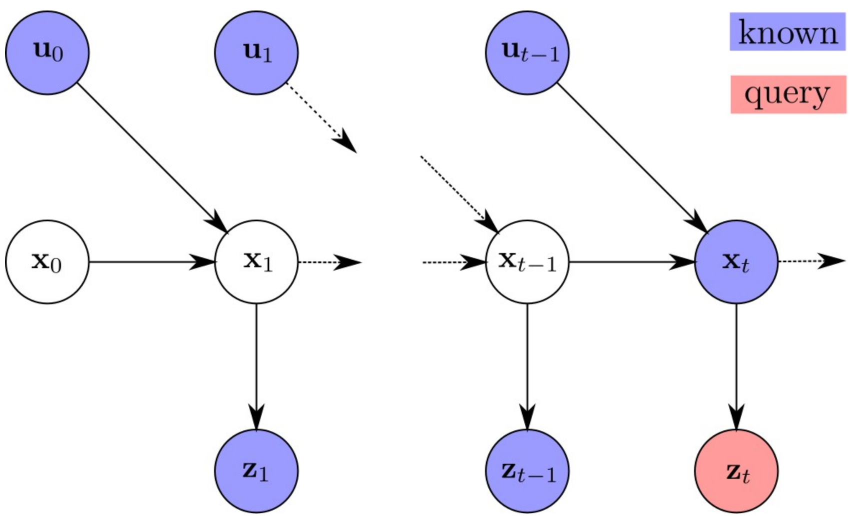
We will come back to the denominator later.

Filtering: Observation model

- our filtering equation becomes:

$$\eta_t p(\mathbf{z}_t \mid \mathbf{x}_t, \mathbf{u}_{0:t-1}, \mathbf{z}_{1:t-1}) p(\mathbf{x}_t \mid \mathbf{u}_{0:t-1}, \mathbf{z}_{1:t-1}) \quad (5)$$

Note that $p(\mathbf{z}_t \mid \mathbf{x}_t, \mathbf{u}_{0:t-1}, \mathbf{z}_{1:t-1})$ means this:



- if we know $\mathbf{x}_t$, we do not need to know $\mathbf{u}_{0:t-1}, \mathbf{z}_{1:t-1}$ to predict $\mathbf{z}_t$, since the state $\mathbf{x}_t$ encodes all the knowledge about the past (Markov assumption):

$$p(\mathbf{z}_t \mid \mathbf{x}_t, \mathbf{u}_{0:t-1}, \mathbf{z}_{1:t-1}) = p(\mathbf{z}_t \mid \mathbf{x}_t) \quad (6)$$

Filtering: Transition Model

- thus, our current equation is:

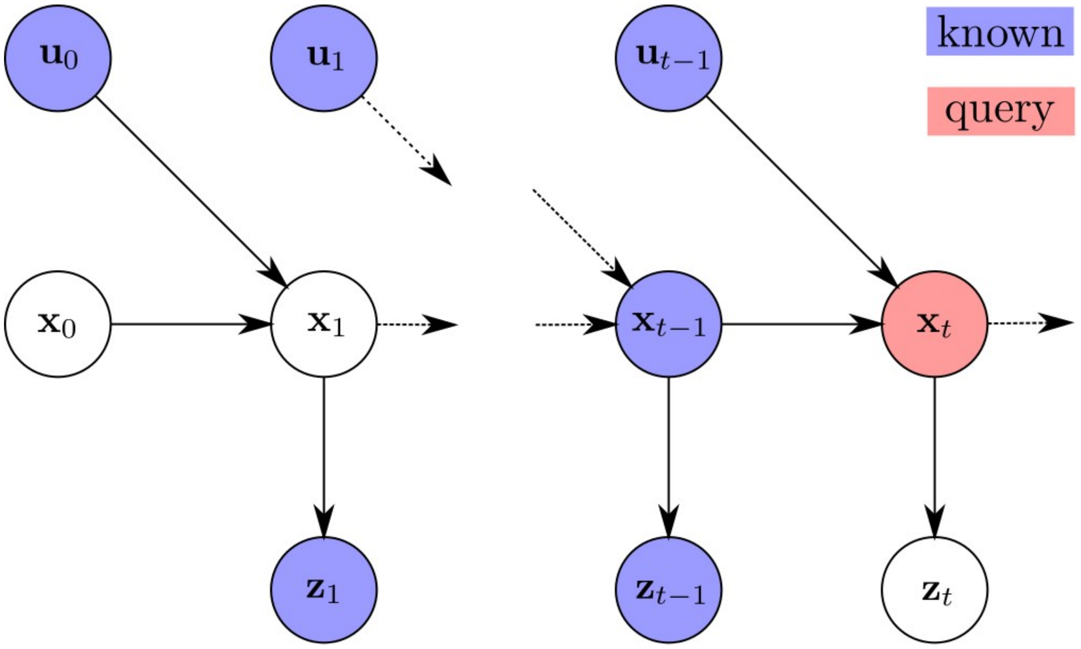
$$p(\mathbf{x}_t \mid \mathbf{u}_{0:t-1}, \mathbf{z}_{1:t}) = \eta_t p(\mathbf{z}_t \mid \mathbf{x}_t) p(\mathbf{x}_t \mid \mathbf{u}_{0:t-1}, \mathbf{z}_{1:t-1}) \quad (7)$$

Still the second part of the equation is obscure.

Our task is to manipulate it, to get something that matches our preconditions.

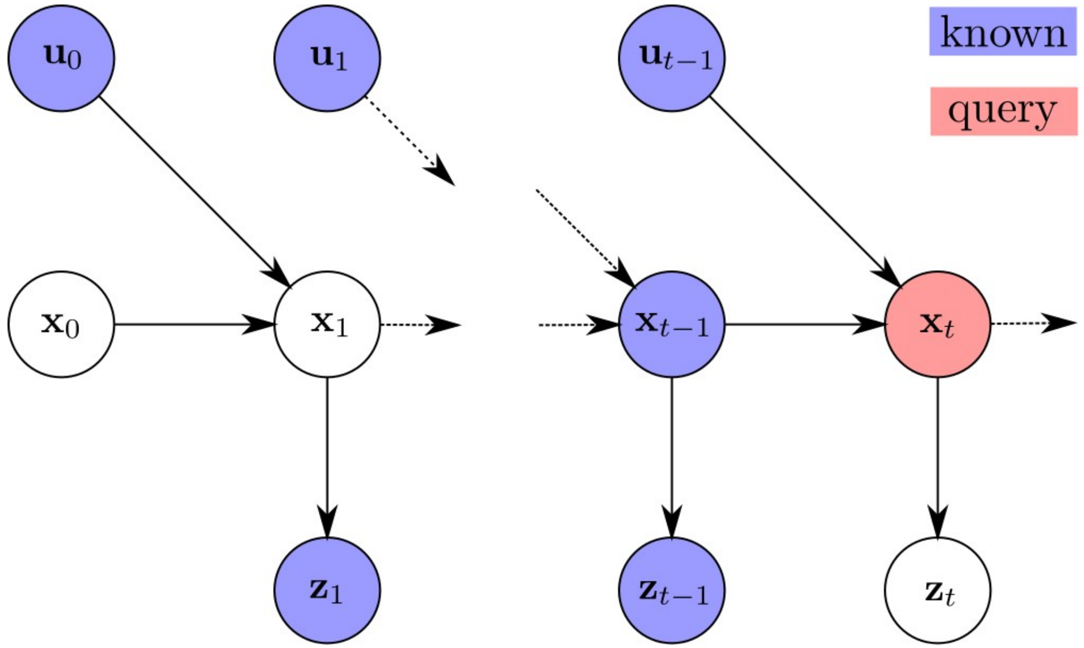
Filtering: Transition Model

Knowing $\mathbf{x}_{t-1}$ would make our life much easier, as we could repeat the trick done for the observation model:



Filtering: Transition Model

Knowing $\mathbf{x}_{t-1}$ would make our life much easier, as we could repeat the trick done for the observation model:



- thus:

$$p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{u}_{0:t-1}, \mathbf{z}_{1:t-1}) = p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{u}_{t-1}) \qquad (8)$$

Filtering: Transition Model

The sad truth is that we do not have $\mathbf{x}_{t-1}$, however:

- recalling the probability identities:

marginalization: $$p(A|C) = \sum_B p(A,B|C) \quad (9)$$

chain rule: $$p(A,B|C) = p(A|B,C)p(B|C) \quad (10)$$

- by combining the two above we obtain:

$$p(A|C) = \sum_B p(A|B,C)p(B|C) \quad (11)$$

Filtering: Transition Model

- let's look again at our problematic equation, and put some letters

$$p(\underbrace{\mathbf{x}_t}_{A} \mid \underbrace{\mathbf{u}_{0:t-1}, \mathbf{z}_{1:t-1}}_{C}) =$$

$$\sum_{\mathbf{x}_{t-1}} p(\underbrace{\mathbf{x}_t}_{A} \mid \underbrace{\mathbf{x}_{t-1}}_{B}, \underbrace{\mathbf{u}_{0:t-1}, \mathbf{z}_{1:t-1}}_{C}) p(\underbrace{\mathbf{x}_{t-1}}_{B} \mid \underbrace{\mathbf{u}_{0:t-1}, \mathbf{z}_{1:t-1}}_{C})$$

- putting in the result of Eq. (8), we highlight the transition model as:

$$= \sum_{\mathbf{x}_{t-1}} p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{u}_{t-1}) p(\mathbf{x}_{t-1} \mid \mathbf{u}_{0:t-1}, \mathbf{z}_{1:t-1}) \quad (12)$$

$$p(A|C) = \sum_B p(A|B,C)p(B|C)$$

Filtering: Wrapup

- after our efforts, we figure out that the recursive filtering equation is the following:

$$p(\mathbf{x}_t \mid \mathbf{u}_{0:t-1}, \mathbf{z}_{1:t}) = \tag{13}$$

$$\eta_t p(\mathbf{z}_t \mid \mathbf{x}_t) \sum_{\mathbf{x}_{t-1}} p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{u}_{t-1}) p(\mathbf{x}_{t-1} \mid \mathbf{u}_{0:t-1}, \mathbf{z}_{1:t-1})$$

Yet, if in the last term of the product in the summation, we would not have a dependency from $\mathbf{u}_{t-1}$, we would have a *recursive* equation.

Luckily we have:

$$p(\mathbf{x}_{t-1} \mid \mathbf{u}_{0:t-1}, \mathbf{z}_{1:t-1}) = p(\mathbf{x}_{t-1} \mid \mathbf{u}_{0:t-2}, \mathbf{z}_{1:t-1}) \tag{14}$$

Since the last control has no influence on $\mathbf{x}_{t-1}$, if we don't know $\mathbf{x}_t$.

Filtering: Wrapup

- we can finally write the recursive equation of filtering as:

$$\overbrace{p(\mathbf{x}_t \mid \mathbf{u}_{0:t-1}, \mathbf{z}_{1:t})}^{b(\mathbf{x}_t)} = \tag{15}$$

$$\eta_t p(\mathbf{z}_t \mid \mathbf{x}_t) \sum_{\mathbf{x}_{t-1}} p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{u}_{t-1}) \underbrace{p(\mathbf{x}_{t-1} \mid \mathbf{u}_{0:t-2}, \mathbf{z}_{1:t-1})}_{b(\mathbf{x}_{t-1})}$$

During the estimation, we do not have the true distribution, but rather the beliefs *estimate*.

- Eq. (16) tells us how to update a current belief once new observations/controls become available:

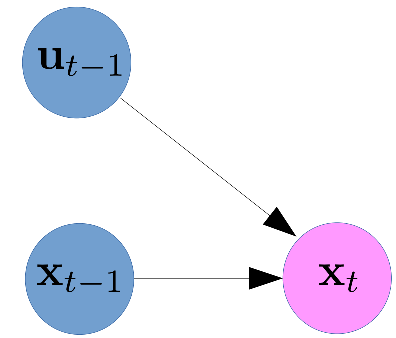
$$b(\mathbf{x}_t) = \eta_t p(\mathbf{z}_t \mid \mathbf{x}_t) \sum_{\mathbf{x}_{t-1}} p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{u}_{t-1}) b(\mathbf{x}_{t-1}) \tag{16}$$

Normalizer: η_t

The *normalizer* η_t is just a constant ensuring that $b(\mathbf{x}_t)$ is still a probability distribution:

$$\eta_t = \frac{1}{\sum_{\mathbf{x}_t} p(\mathbf{z}_t \mid \mathbf{x}_t) \sum_{\mathbf{x}_{t-1}} p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{u}_{t-1}) b(\mathbf{x}_{t-1})} \quad (17)$$

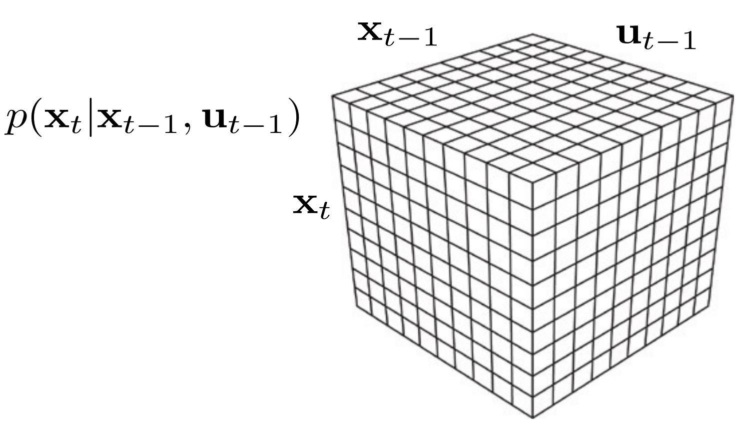
Filtering: Alternative Formulation



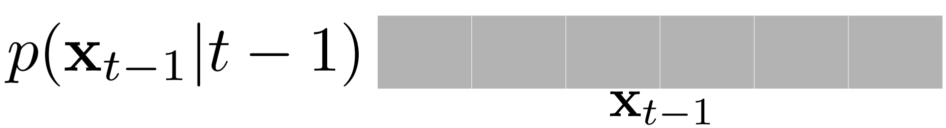
Predict: incorporate in the last belief $b_{t-1|t-1}$ the most recent control $\mathbf{u}_{t-1}$.

Ingredients:

- Transition model

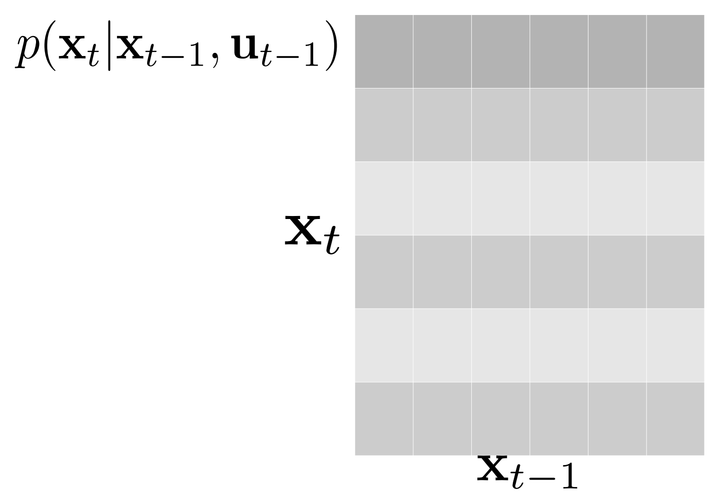


- Prior belief



- Control $\mathbf{u}_{t-1}$

The control is known, so we can work with a "2D" distribution selected according to the current $\mathbf{u}_{t-1}$.



Filtering: Alternative Formulation

Predict:

- From the transition model and the last state, compute the following joint distribution through *chain rule*:

$$p(\mathbf{x}_t, \mathbf{x}_{t-1} | t-1) = p(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{u}_{t-1}) \underbrace{p(\mathbf{x}_{t-1} | t-1)}_{b_{t-1|t-1}}$$

- From the joint, remove $\mathbf{x}_{t-1}$ through *marginalization:*

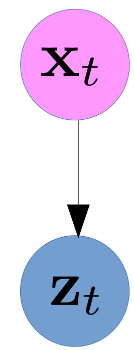
$$\underbrace{p(\mathbf{x}_t | t-1)}_{b_{t|t-1}} = \sum_{\mathbf{x}_{t-1}} p(\mathbf{x}_t, \mathbf{x}_{t-1} | t-1)$$

- Programmatically (discrete case)

```
BeliefType b_pred = BeliefType::Zero;
for (x_i : X)
  for (x_j: X)
     b_pred[x_j] += b[x_i]*transitionModel(x_j,x_i,u);
```

Filtering: Alternative Formulation

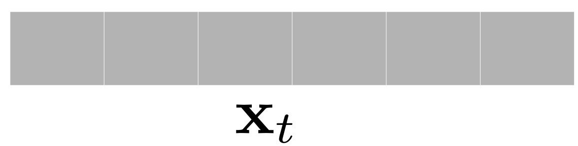
Update: incorporate in the predicted belief $b_{t|t-1}$ the new measurement $\mathbf{z}_t$



Ingredients

- Predicted belief

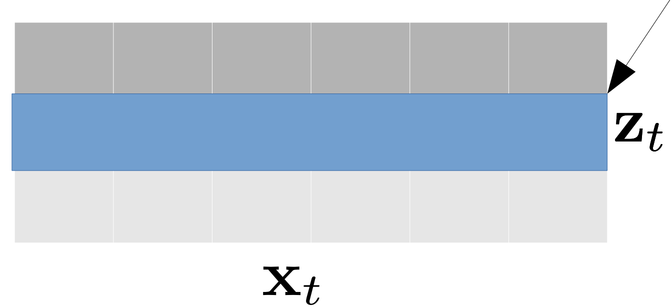
$p(\mathbf{x}_t|t-1)$



- Observation model

$p(\mathbf{z}_t|\mathbf{x}_t)$

we focus on the known measurement, the rest is irrelevant



- Known measurement $\mathbf{z}_t$

Filtering: Alternative Formulation

Update: from the predicted belief $b_{t|t-1}$, compute the joint distribution that predicts the observation.

- Joint over state and measurement (*chain rule)*:

$$p(\mathbf{x}_t, \mathbf{z}_t | t) = p(\mathbf{z}_t | \mathbf{x}_t) p(\mathbf{x}_t, | t-1)$$

- *Condition* on the actual measurement:

$$\underbrace{p(\mathbf{x}_t | t)}_{b_{t|t}} = \frac{p(\mathbf{x}_t, \mathbf{z}_t | t)}{p(\mathbf{z}_t | t)}$$

- Programmatically (discrete case)

```
float normalizer=0;
for (x_i : X) {
    b[x_i] = b_pred[x_i] * observationModel(z,x_i);
    normalizer += b[x_i];
}
b *= 1./normalizer;
```