

Probabilistic Robotics Course

Dynamic Bayesian Networks

Giorgio Grisetti

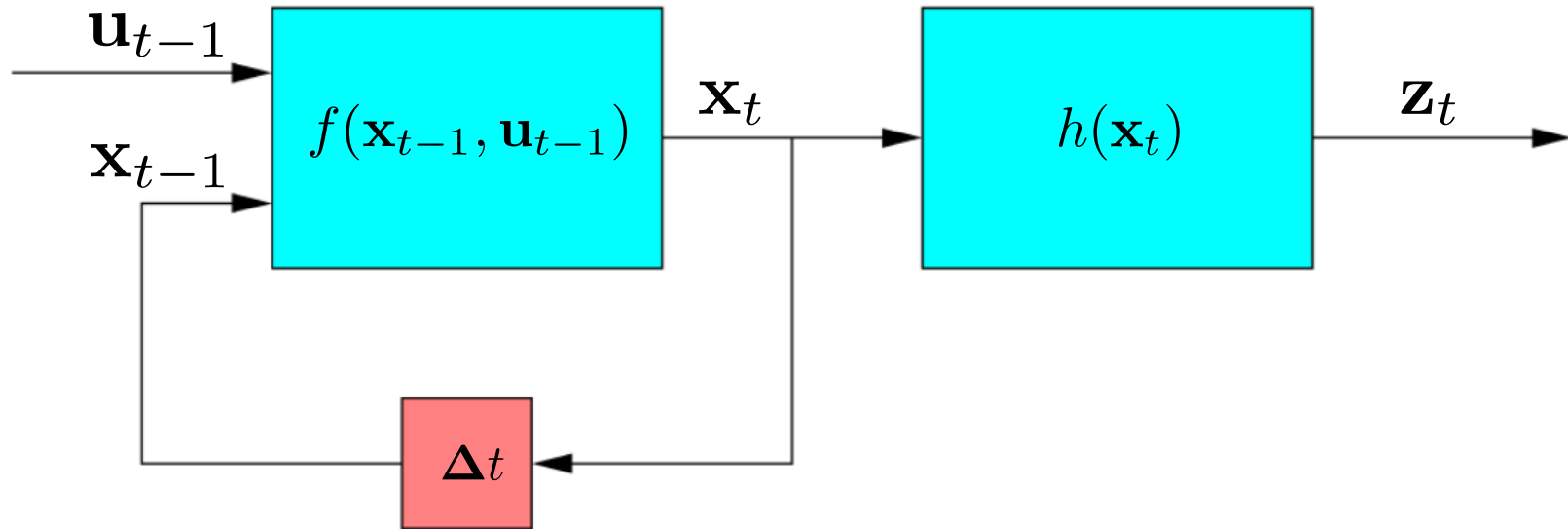
`grisetti@diag.uniroma1.it`

Dept of Computer Control and Management Engineering
Sapienza University of Rome

Overview

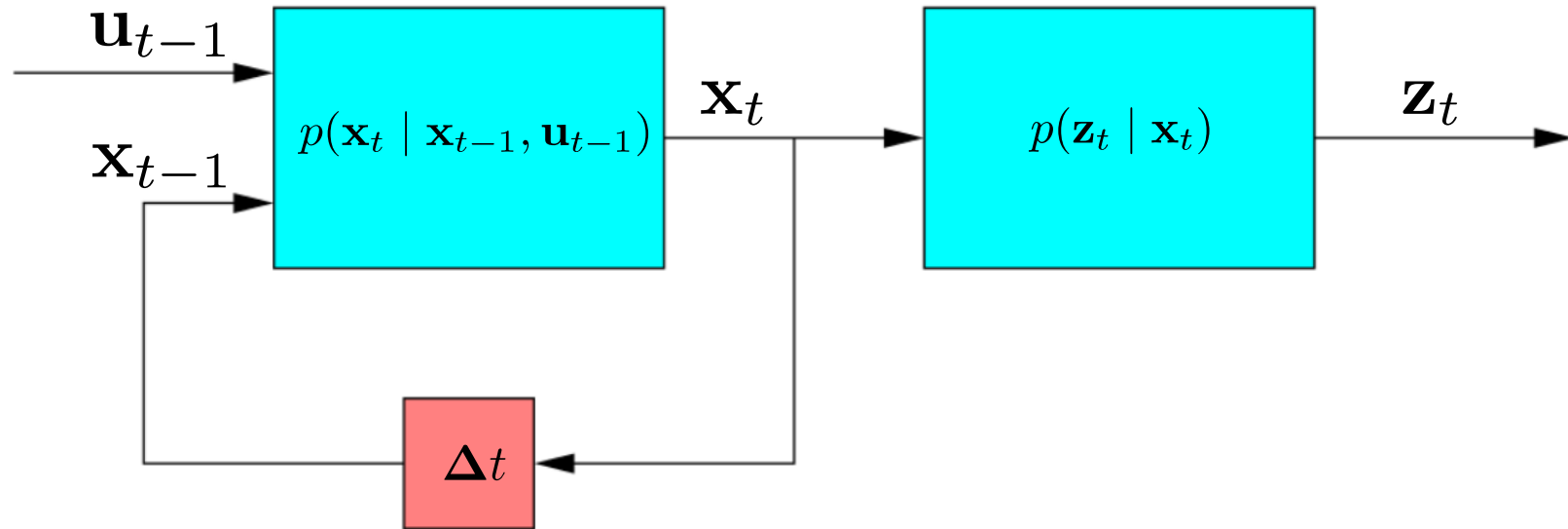
- Probabilistic Dynamic Systems
- Dynamic Bayesian Networks (DBN)
- Inference on DBN
- Recursive Bayes Equation

Dynamic System Deterministic View



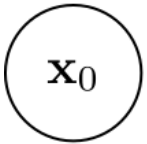
- $f(\mathbf{x}_{t-1}, \mathbf{u}_{t-1})$: transition function
- $h(\mathbf{x}_t)$: observation function
- \mathbf{x}_{t-1} : previous state
- \mathbf{x}_t : current state
- \mathbf{z}_t : current observation
- \mathbf{u}_{t-1} : previous control/action
- Δt : delay

Dynamic System Probabilistic View



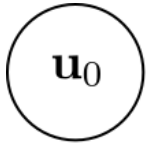
- $p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{u}_{t-1})$: transition model
- $p(\mathbf{z}_t \mid \mathbf{x}_t)$: observation model
- \mathbf{x}_{t-1} : previous state
- \mathbf{x}_t : current state
- \mathbf{z}_t : current observation
- \mathbf{u}_{t-1} : previous control/action
- Δt : delay

Evolution of a Dynamic System

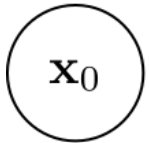


Let's start from a known initial state distribution $p(\mathbf{x}_0)$.

Evolution of a Dynamic System



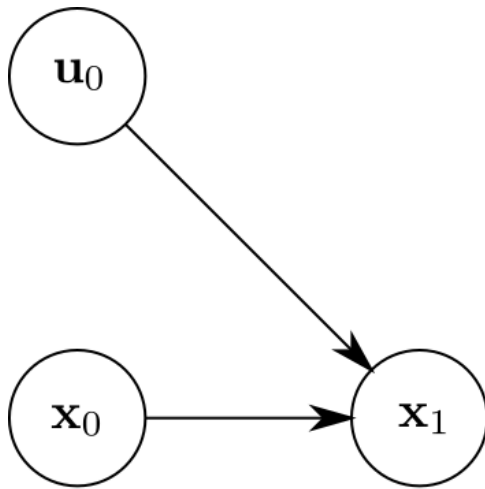
\mathbf{u}_0



\mathbf{x}_0

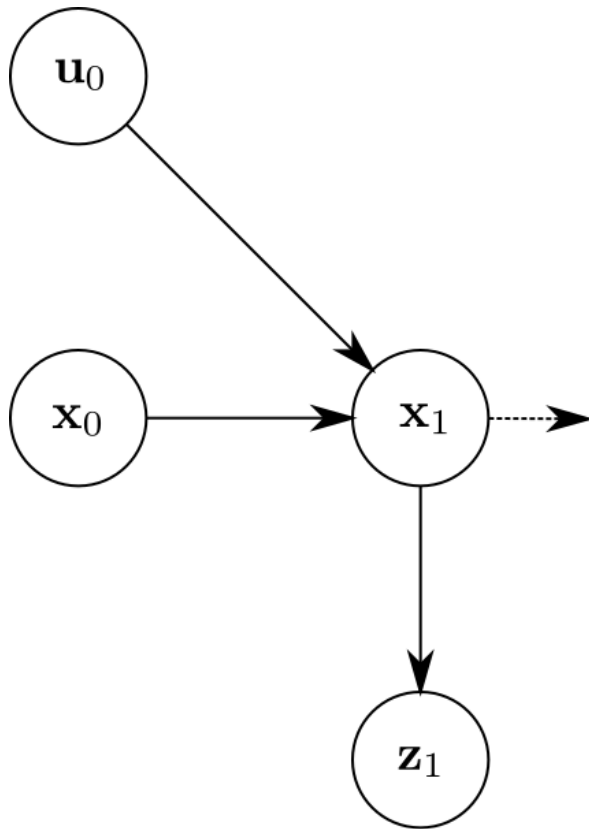
A control \mathbf{u}_0 becomes available.

Evolution of a Dynamic System



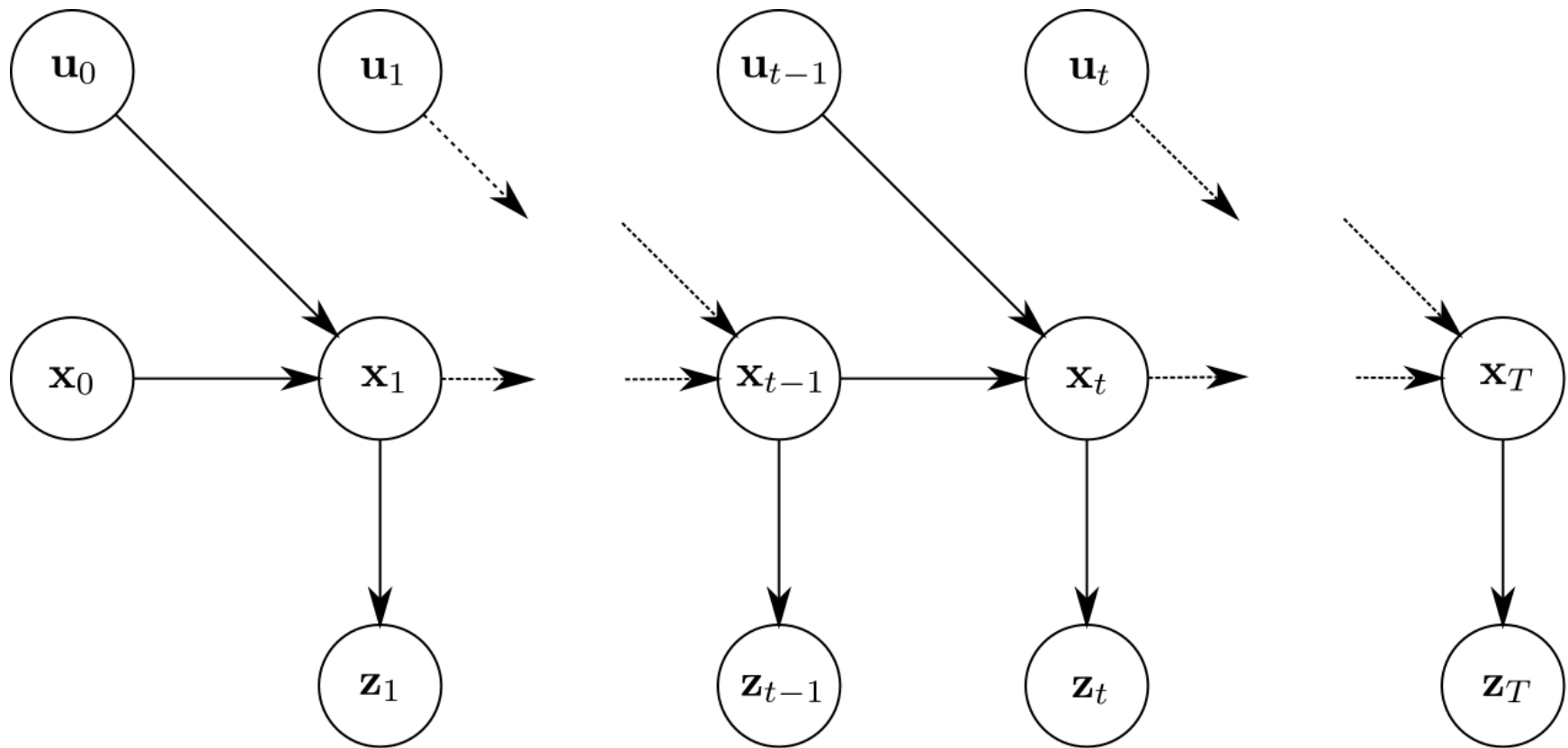
The transition model $p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{u}_{t-1})$ correlates the current state \mathbf{x}_1 with the previous control \mathbf{u}_0 and the previous state \mathbf{x}_0 .

Evolution of a Dynamic System



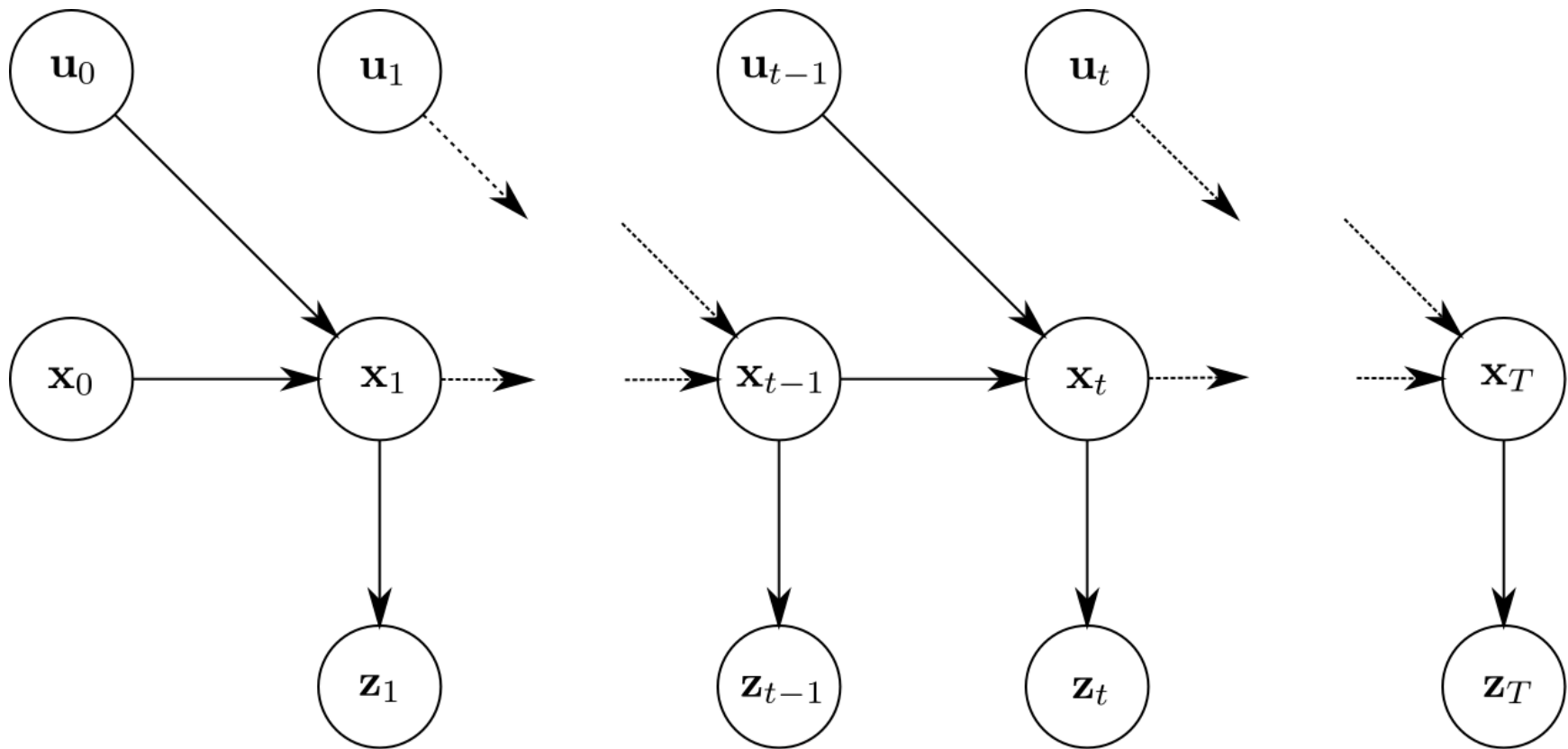
The observation model $p(\mathbf{z}_t \mid \mathbf{x}_t)$ correlates the observation \mathbf{z}_1 and the current state \mathbf{x}_1 .

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 $\mathbf{z}_t \in \mathcal{R}^{\#beams}$, laser range measurements
 - Controls $\mathbf{u}_t \in \mathcal{R}^2$, translational and rotational speed
- HMM
 - States $\mathbf{x}_t \in X_1, \dots, X_{N_x}$, finite states
 - Observations $\mathbf{u}_t \in U_1, \dots, U_{N_u}$, finite observations
 - Controls $\mathbf{z}_t \in Z_1, \dots, Z_{N_z}$, 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, usually¹ 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	$p(\mathbf{x}_T \mathbf{u}_{0:T-1}, \mathbf{z}_{1:T})$	$\mathbf{u}_{0:T-1}, \mathbf{z}_{1:T}$
Smoothing	$p(\mathbf{x}_k \mathbf{u}_{0:T-1}, \mathbf{z}_{1:T}), 0 < k < T$	$\mathbf{u}_{0:T-1}, \mathbf{z}_{1:T}$
Max a Posteriori	$\operatorname{argmax}_{\mathbf{x}_{0:T}} p(\mathbf{x}_{0:T} \mathbf{u}_{0:T-1}, \mathbf{z}_{1:T})$	$\mathbf{u}_{0:T-1}, \mathbf{z}_{1:T}$

¹usually does not mean always

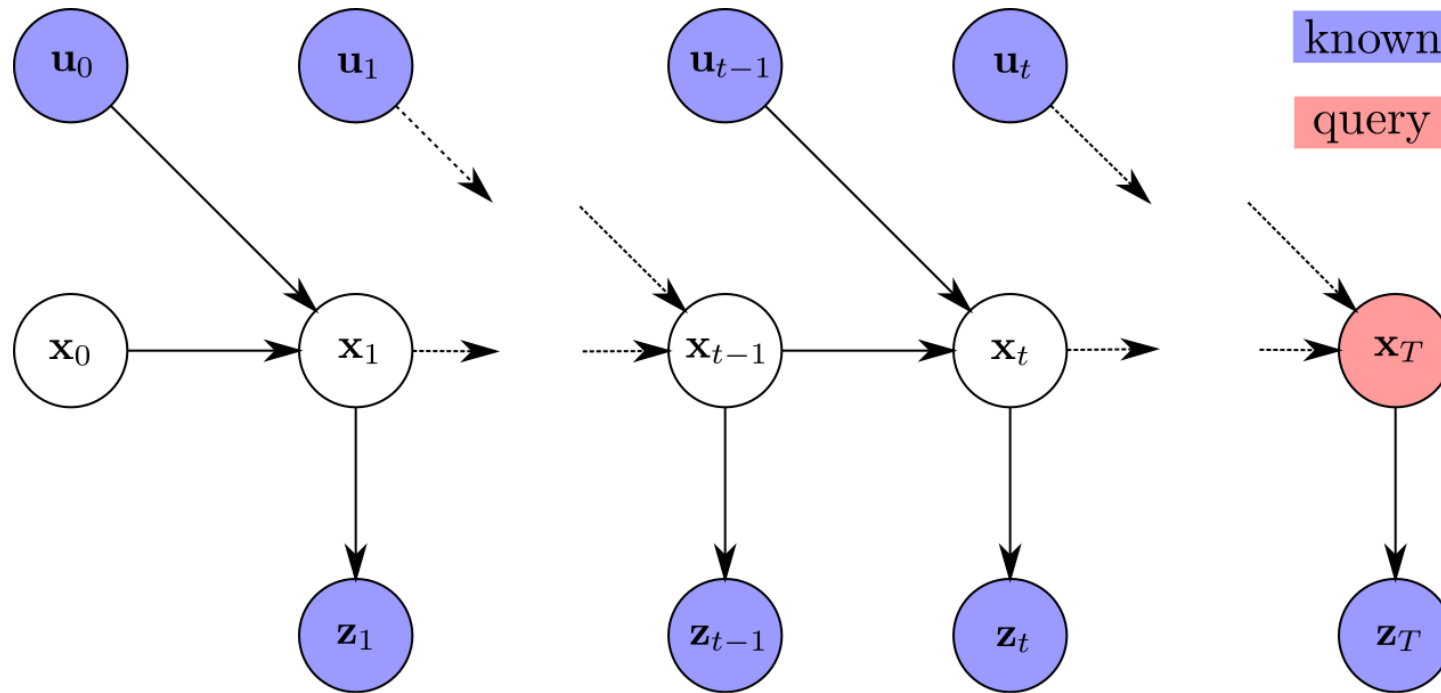
Typical Inferences in a DBN

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

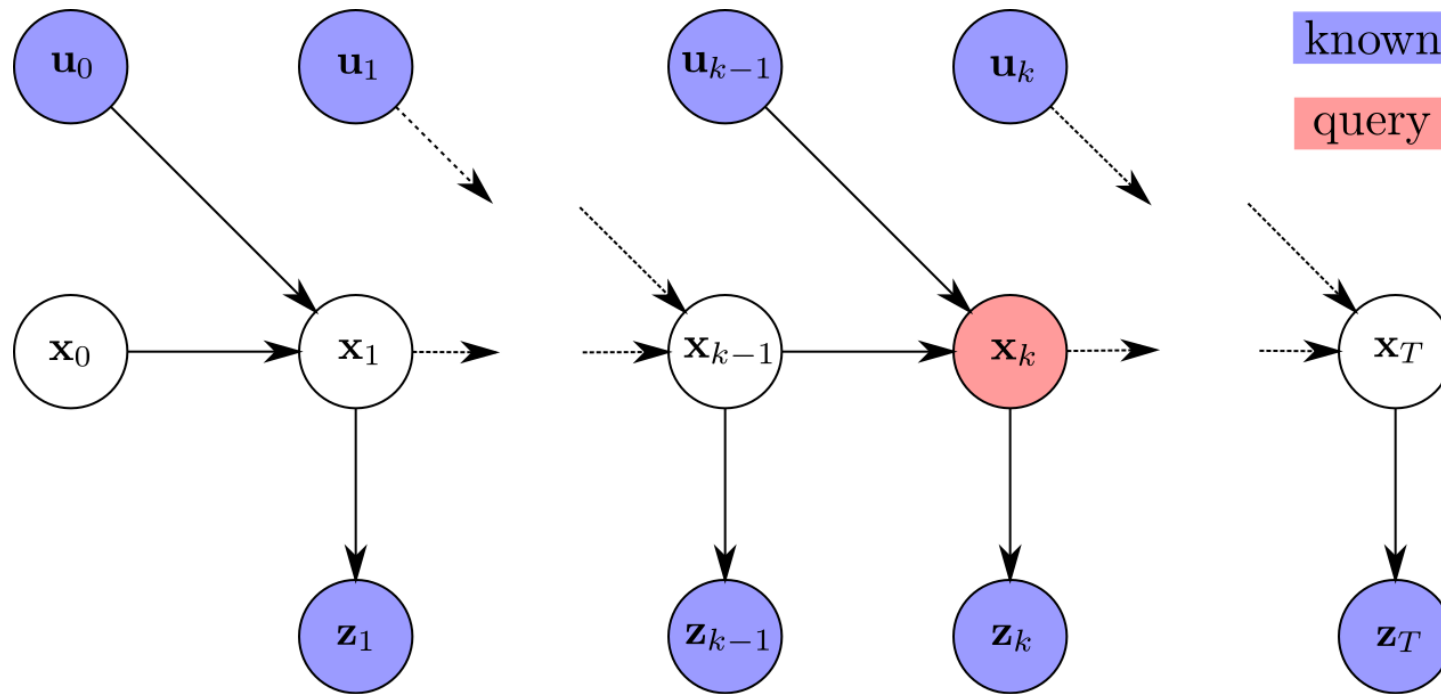


Given:

- 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



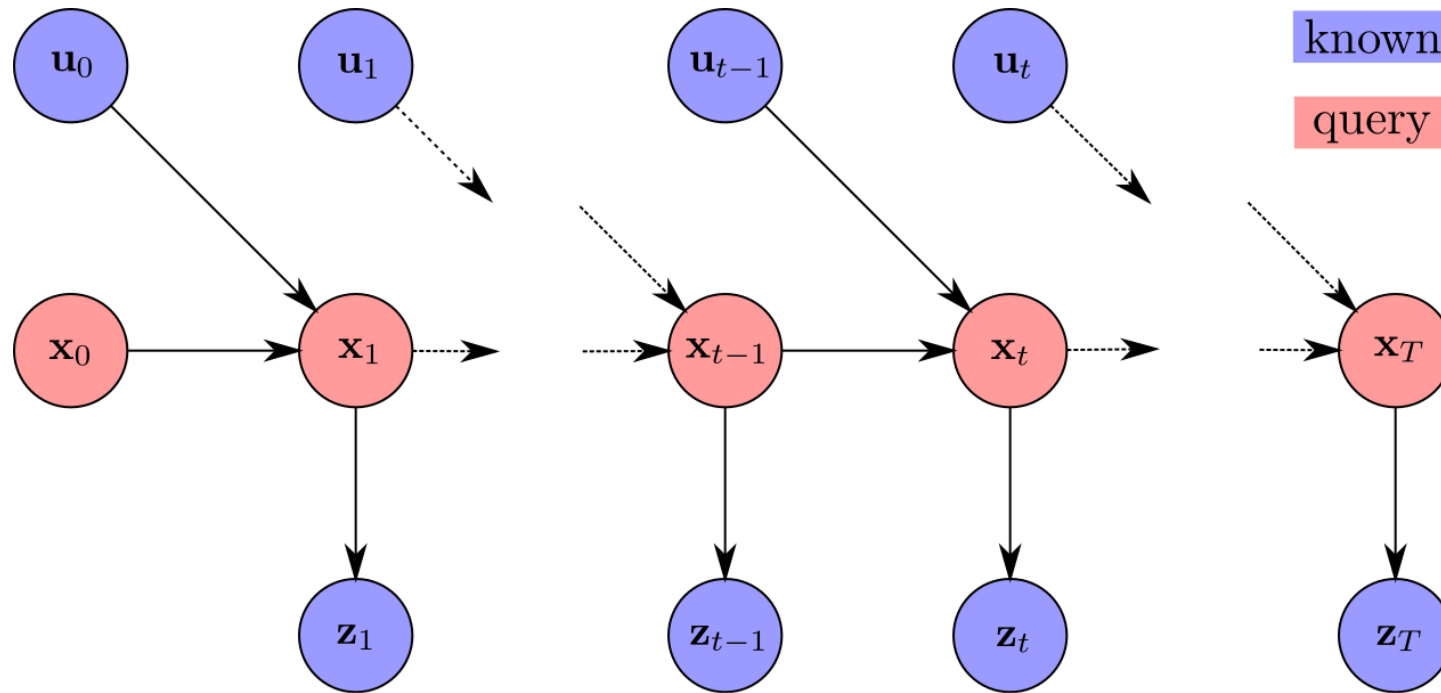
Given:

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

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

We know:

- the observations $\mathbf{z}_{1:T}$
- the controls $\mathbf{u}_{0:T-1}$
- $p(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{u}_{t-1})$: 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 .
- $p(\mathbf{z}_t | \mathbf{x}_t)$: 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 .
- $b(\mathbf{x}_{t-1})$, which is our previous belief about the previous state $p(\mathbf{x}_{t-1} | \mathbf{u}_{0:t-2}, \mathbf{z}_{1:t-1})$.

Filtering: Bayes Rule

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

- splitting \mathbf{z}_t :

$$= p(\underbrace{\mathbf{x}_t}_A | \underbrace{\mathbf{z}_t}_B, \underbrace{\mathbf{u}_{0:t-1}, \mathbf{z}_{1:t-1}}_C) \quad (2)$$

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

$$= \frac{p(\underbrace{\mathbf{z}_t}_B | \underbrace{\mathbf{x}_t}_A, \underbrace{\mathbf{u}_{0:t-1}, \mathbf{z}_{1:t-1}}_C) p(\underbrace{\mathbf{x}_t}_A | \underbrace{\mathbf{u}_{0:t-1}, \mathbf{z}_{1:t-1}}_C)}{p(\underbrace{\mathbf{z}_t}_B | \underbrace{\mathbf{u}_{0:t-1}, \mathbf{z}_{1:t-1}}_C)} \quad (3)$$

Filtering: Denominator

- let the denominator

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

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

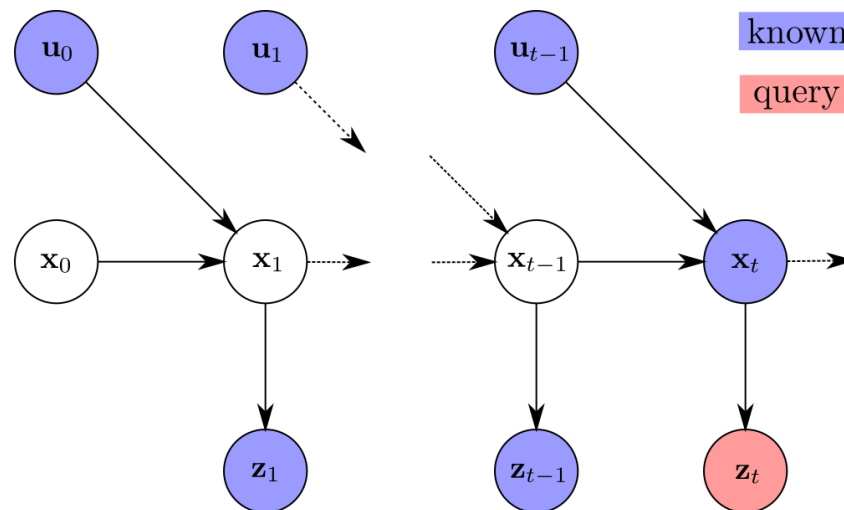
We will come back to the denominator later.

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:

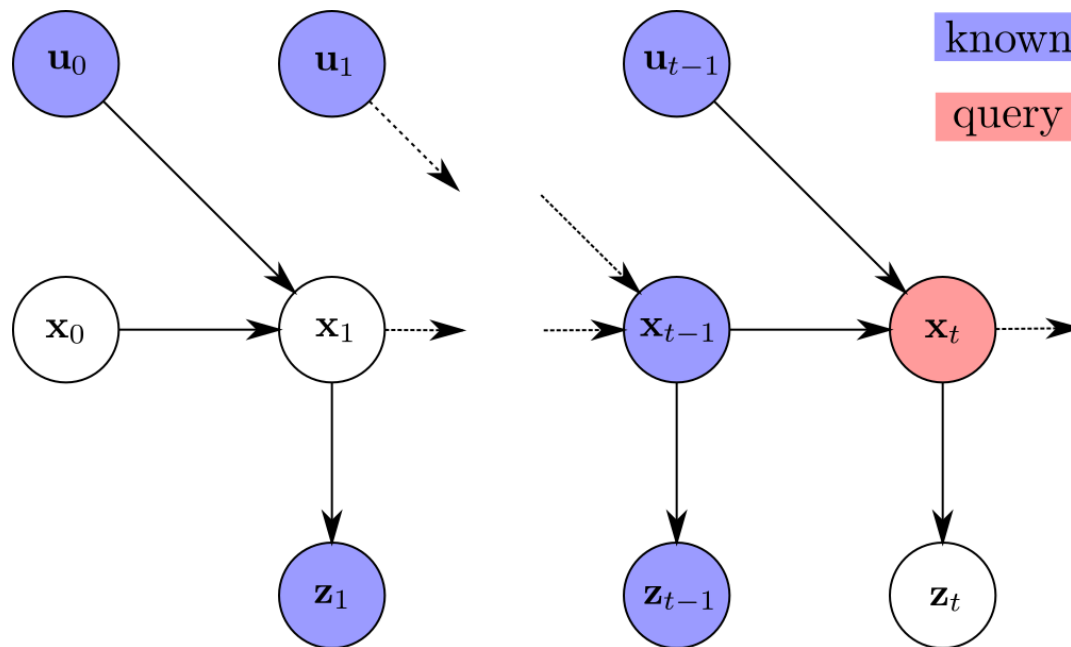
$$p(\mathbf{x}_t \mid \mathbf{u}_{0:t-1}, \mathbf{z}_{1:t}) = \eta_t p(\mathbf{z}_t \mid \mathbf{x}_t) p(\mathbf{x}_t \mid \mathbf{u}_{0:t-1}, \mathbf{z}_{1:t-1}) \quad (7)$$

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

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



■ thus:

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

Filtering: Transition Model

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

- recalling the probability identities:

$$p(A|C) = \sum_b p(A, B|C) \quad (9)$$

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

- 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

$$\begin{aligned} & p(\underbrace{\mathbf{x}_t}_A \mid \underbrace{\mathbf{u}_{0:t-1}, \mathbf{z}_{1:t-1}}_C) = \\ & \sum_{\mathbf{x}_{t-1}} p(\underbrace{\mathbf{x}_t}_A \mid \underbrace{\mathbf{x}_{t-1}}_B, \underbrace{\mathbf{u}_{0:t-1}, \mathbf{z}_{1:t-1}}_C) p(\underbrace{\mathbf{x}_{t-1}}_B \mid \underbrace{\mathbf{u}_{0:t-1}, \mathbf{z}_{1:t-1}}_C) \end{aligned}$$

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

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

Filtering: Wrapup

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

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

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

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:

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

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:

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

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

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:

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

Normalizer: η_t

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

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

Filtering: Discrete case

```
60
61 #----- FILTERING -----
62
63 #retrieve new robot position according to our transition model
64 robot_position = getRobotPosition(map, robot_position, control_input);
65
66 #obtain current observations according to our observation model
67 observations = getObservations(map, robot_position);
68
69 #INITIALIZE robot position belief
70 belief_matrix_previous = belief_matrix;
71 belief_matrix = zeros(map_rows, map_cols);
72
73 #PREDICT robot position belief
74 for row = 1:map_rows
75     for col = 1:map_cols
76         belief_matrix += transitionModel(map, row, col, control_input)*belief_matrix_previous(row, col);
77     endfor
78 endfor
79
80 #UPDATE robot position belief and COMPUTE the normalizer
81 inverse_normalizer = 0;
82 for row = 1:map_rows
83     for col = 1:map_cols
84         belief_matrix(row, col) *= observationModel(map, row, col, observations);
85         inverse_normalizer += belief_matrix(row, col);
86     endfor
87 endfor
88
89 #NORMALIZE the belief probabilities to [0, 1]
90 normalizer = 1./inverse_normalizer;
91 belief_matrix *= normalizer;
92
93 #----- FILTERING -----
94
```

Filtering: Alternative Formulation

Predict: incorporate in the last belief b_{t-1} , the most recent observation.

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

$$p(\mathbf{x}_t, \mathbf{x}_{t-1} | \mathbf{z}_{1:t-1}, \mathbf{u}_{1:t-1}) = p(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{u}_{t-1}) \underbrace{p(\mathbf{x}_t, \mathbf{x}_{t-1} | \mathbf{z}_{1:t-1}, \mathbf{u}_{1:t-2})}_{b_{t-1}}$$

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

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

Filtering: Alternative Formulation

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

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

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

- Incorporate the current observation through *conditioning* on the actual measurement:

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

Note: since we already know the value of \mathbf{z}_t , we do not need to compute the joint distribution for all possible values of $\mathbf{z} \in \mathcal{Z}$, but just for the current measurement.