DTU

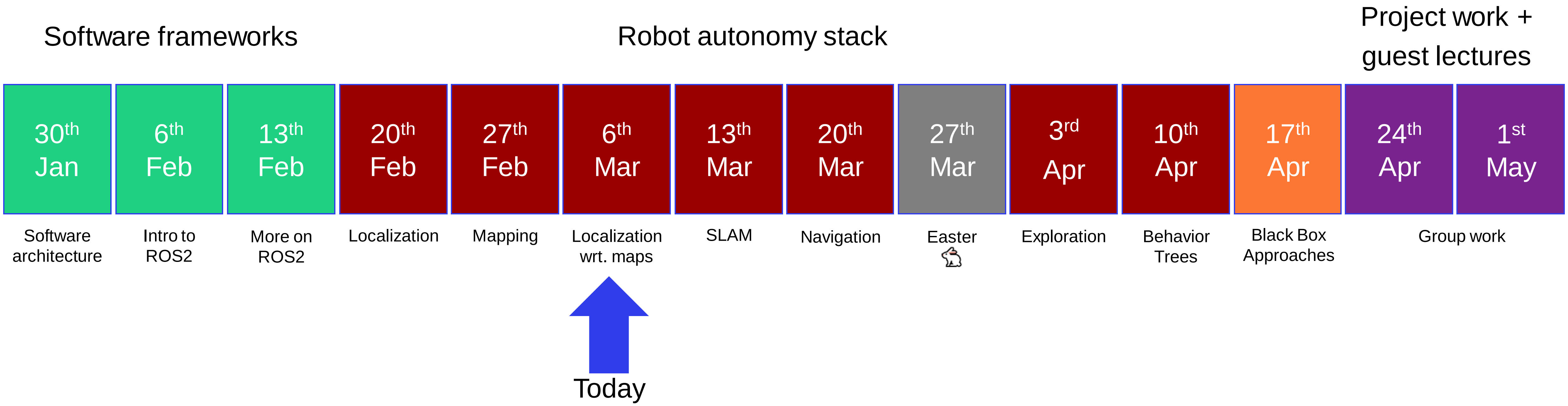
Rasmus Andersen

34761 – Robot Autonomy

Localization w.r.t. maps

Overview of 34761 – Robot Autonomy

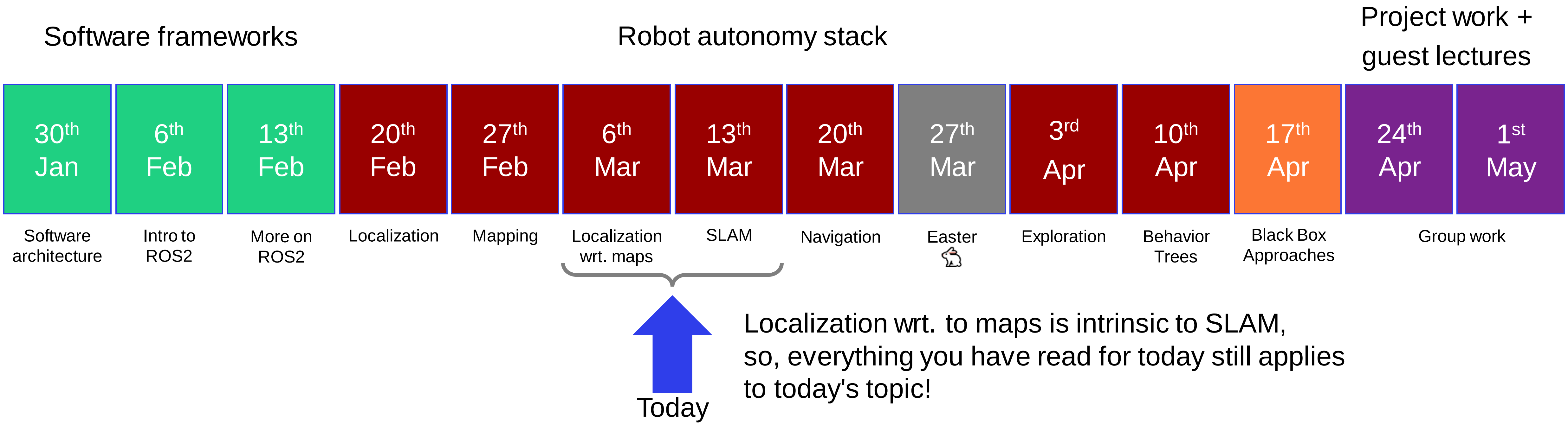
- 3 lectures on software frameworks
- 7 lectures on building your own autonomy stack for a mobile robot
- 1 lecture on DL/RL – an overview of black-box approaches to what you have done
- 2 lectures of project work before hand in + guest lectures



Software frameworks			Robot autonomy stack									Project work + guest lectures	
30th Jan	6th Feb	13th Feb	20th Feb	27th Feb	6th Mar	13th Mar	20th Mar	27th Mar	3rd Apr	10th Apr	17th Apr	24th Apr	1st May
Software architecture	Intro to ROS2	More on ROS2	Localization	Mapping	Localization wrt. maps	SLAM	Navigation	Easter 🐇	Exploration	Behavior Trees	Black Box Approaches	Group work	

Overview of 34761 – Robot Autonomy

- 3 lectures on software frameworks
- 7 lectures on building your own autonomy stack for a mobile robot
- 1 lecture on DL/RL – an overview of black-box approaches to what you have done
- 2 lectures of project work before hand in + guest lectures



	Date	Topic
Software frameworks	30th Jan	Software architecture
	6th Feb	Intro to ROS2
	13th Feb	More on ROS2
Robot autonomy stack	20th Feb	Localization
	27th Feb	Mapping
	6th Mar	Localization wrt. maps
	13th Mar	SLAM
	20th Mar	Navigation
	27th Mar	Easter 🐇
	3rd Apr	Exploration
	10th Apr	Behavior Trees
	17th Apr	Black Box Approaches
Project work + guest lectures	24th Apr	Group work
	1st May	

Today

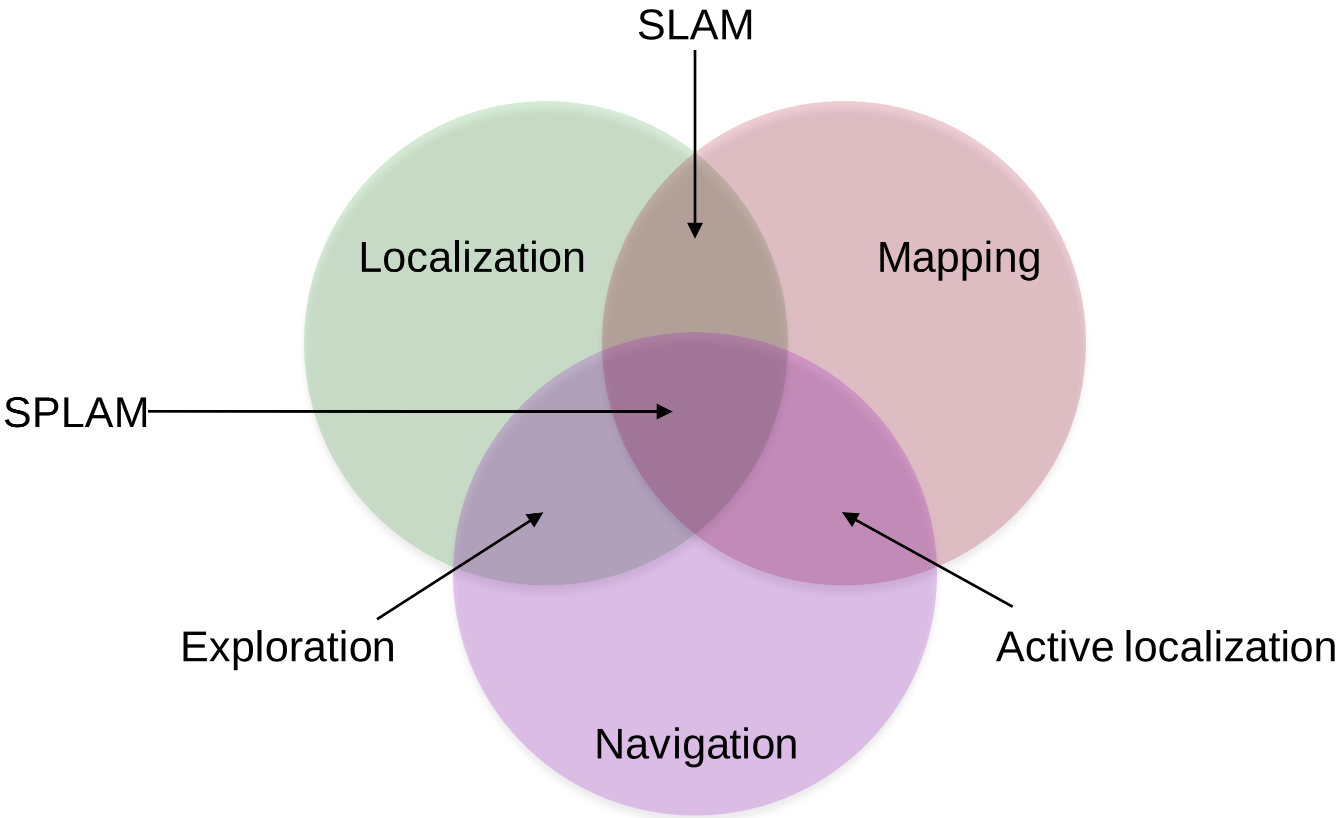
Localization wrt. to maps is intrinsic to SLAM, so, everything you have read for today still applies to today's topic!

Outline for the next 7 weeks

- Our own autonomy stack:

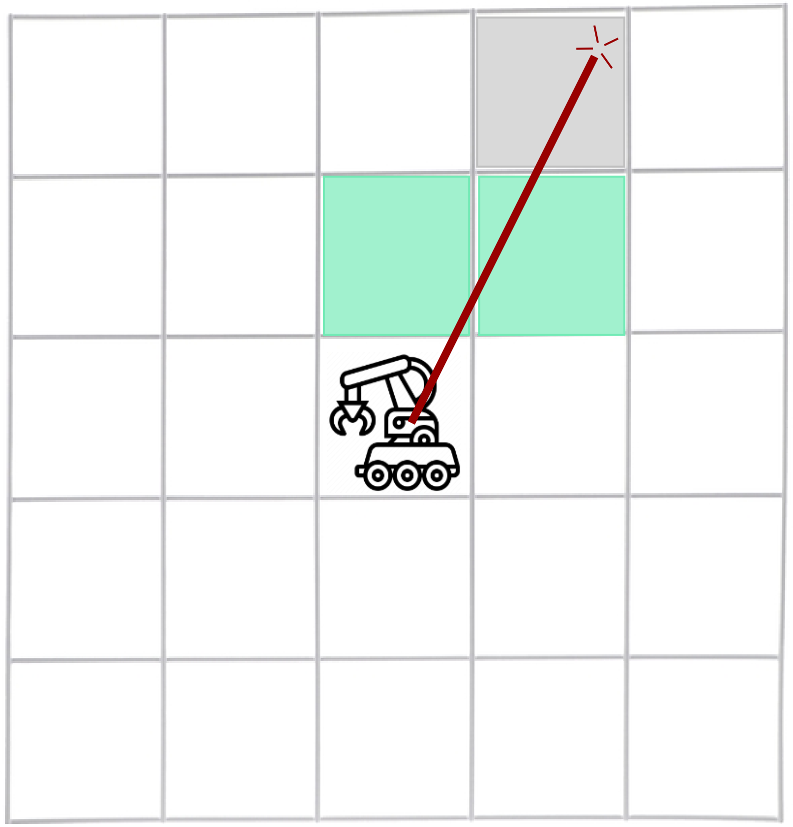
Topic of today

1. Localization
2. Mapping
3. Navigation
4. Exploration
5. Behaviour trees



Recall from last lecture

- What can we use a map for?
- What are the challenges with a map?
- Map types?
- How can we represent a map?



Exercises from last lecture

- Plotting your sensor data using a ROS interface?
  - Occupied vs free cells

- New forum on LEARN to help answer facilitate questions

Localization w.r.t. a map

- We have a map; we want to know where in the map we are
  - – Estimates the location and orientation of the robot in the environment as it moves

- How do we get the initial position?
  - – Bayes filtering
  - – Particle filter / monte carlo localization

Bayes filtering

- Performs state estimation in a recursive fashion to estimate the current state/location of a system
- From time step t to timestep t+1 using only the current observation
- Our belief about the current state

$$Bel(x_t) = P(x_t | z_1, \dots, z_t)$$

- Using bayes rule

$$Bayes = \eta \boxed{P(z_t | x_t, z_1, \dots, z_{t-1})} \boxed{P(x_t | z_1, \dots, z_{t-1})}$$

Likelihood (what’s the likelihood of getting z_t)

Prior (our prediction of the state we are in, and therefore, it doesn’t depend on z_t)

Bayes filtering

- Using bayes rule

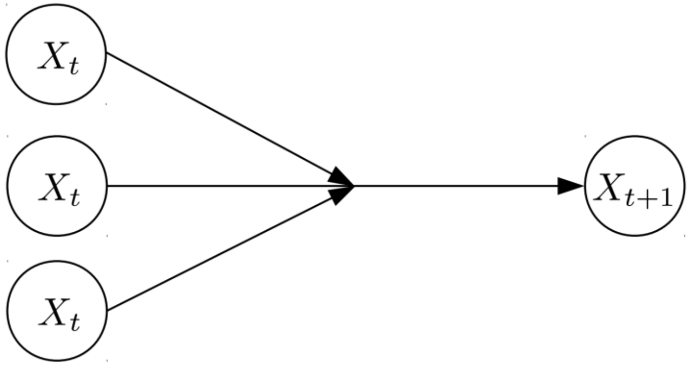
$$Bayes = \eta \boxed{P(z_t|x_t, z_1, \dots, z_{t-1})} \boxed{P(x_t|z_1, \dots, z_{t-1})}$$

Likelihood (what's the likelihood of getting z_t)

Prior (our prediction of the state we are in, and therefore, it doesn't depend on z_t)

- Sensor independence
  - Our current sensor input doesn't depend on previous sensor inputs
  - So the likelihood can be simplified to: $P(z_t|x_t)$
  - The prior do depend on previous states, so we can expand the prior:

$$P(x_t|z_1, \dots, z_{t-1}) = \int P(x_t|z_1, \dots, z_{t-1}, x_{t-1}) \cdot P(x_{t-1}|z_1, \dots, z_{t-1}) dx_{t-1}$$



Bayes filtering

- Using bayes rule

$$Bayes = \eta \boxed{P(z_t|x_t, z_1, \ldots, z_{t-1})}\boxed{P(x_t|z_1, \ldots, z_{t-1})}$$

Likelihood (what's the likelihood of getting z_t)

Prior (our prediction of the state we are in, and therefore, it doesn't depend on z_t)

- Sensor independence
  - Our current sensor input doesn't depend on previous sensor inputs
  - So the likelihood can be simplified to: $P(z_t|x_t)$
  - The prior do depend on previous states, so we can expand the prior:

$$P(x_t|z_1, \ldots, z_{t-t}) = \int P(x_t|z_1, \ldots, z_{t-1}, x_{t-1}) \cdot P(x_{t-1}|z_1, \ldots, z_{t-1}) dx_{t-1}$$

  - The markovian property (i.e. the current state can be explained through only the previous state):

$$\eta P(z_t|x_t) \int \boxed{P(x_t|\cancel{z_1, \ldots, z_{t-1},} x_{t-1})} \cdot P(x_{t-1}|z_1, \ldots, z_{t-1}) dx_{t-1}$$

Effectively, this becomes our environment dynamics

(given x_{t-1}, how likely am I to transition to x_t)

Bayes filtering

- Using bayes rule

$$Bayes = \eta \boxed{P(z_t|x_t, z_1, \ldots, z_{t-1})}\boxed{P(x_t|z_1, \ldots, z_{t-1})}$$

Likelihood (what's the likelihood of getting z_t)

Prior (our prediction of the state we are in, and therefore, it doesn't depend on z_t)

- Sensor independence
  - Our current sensor input doesn't depend on previous sensor inputs
  - So the likelihood can be simplified to: $P(z_t|x_t)$
  - The prior do depend on previous states, so we can expand the prior:

$$P(x_t|z_1, \ldots, z_{t-t}) = \int P(x_t|z_1, \ldots, z_{t-1}, x_{t-1}) \cdot P(x_{t-1}|z_1, \ldots, z_{t-1}) dx_{t-1}$$

  - The markovian property (i.e. the current state can be explained through only the previous state):

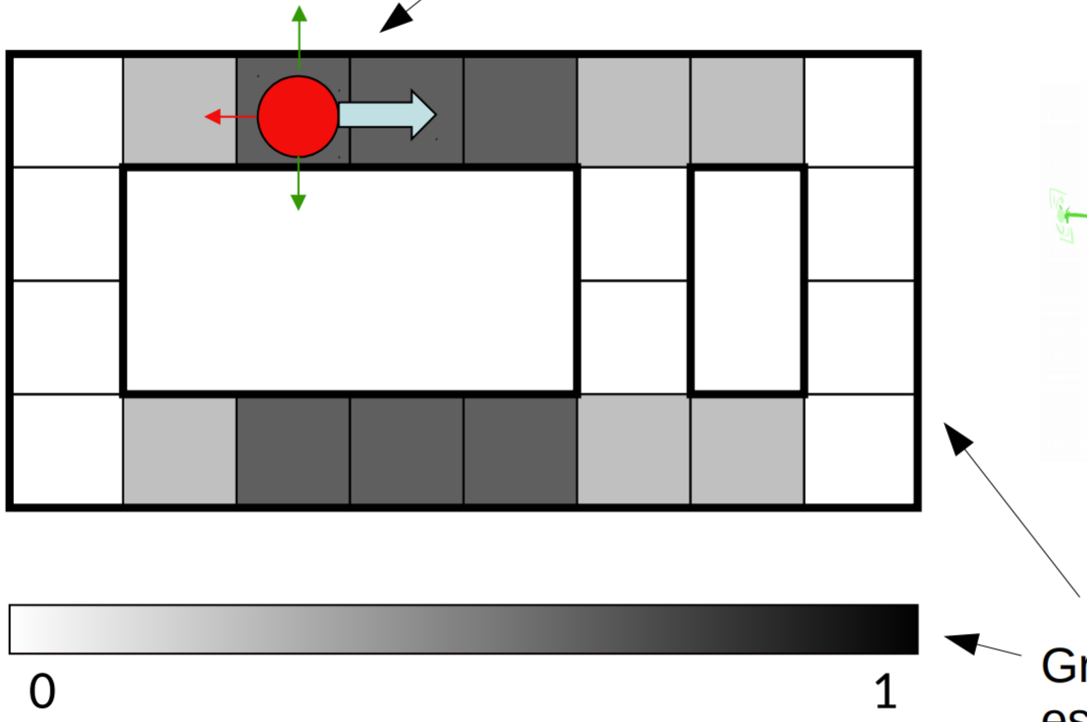
$$\eta P(z_t|x_t) \int P(x_t|\cancel{z_1, \ldots, z_{t-1},} x_{t-1}) \cdot P(x_{t-1}|z_1, \ldots, z_{t-1}) dx_{t-1}$$

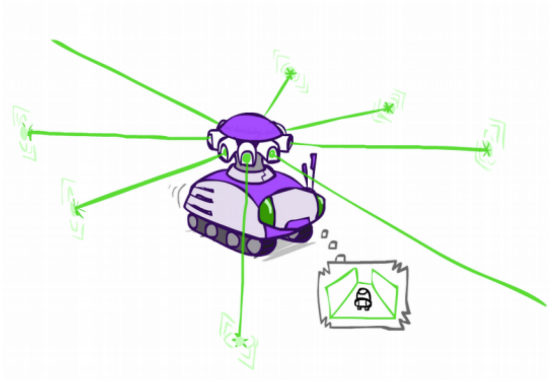
  - $P(x_{t-1}|z_1, \ldots, z_{t-1})$ is actually just our belief from the previous timestep: $Bel(x_{t-1})$

$$Bel(x_t) = \eta P(z_t|x_t) \int P(x_t|x_{t-1}) \cdot Bel(x_{t-1}) dx_{t-1}$$

Example

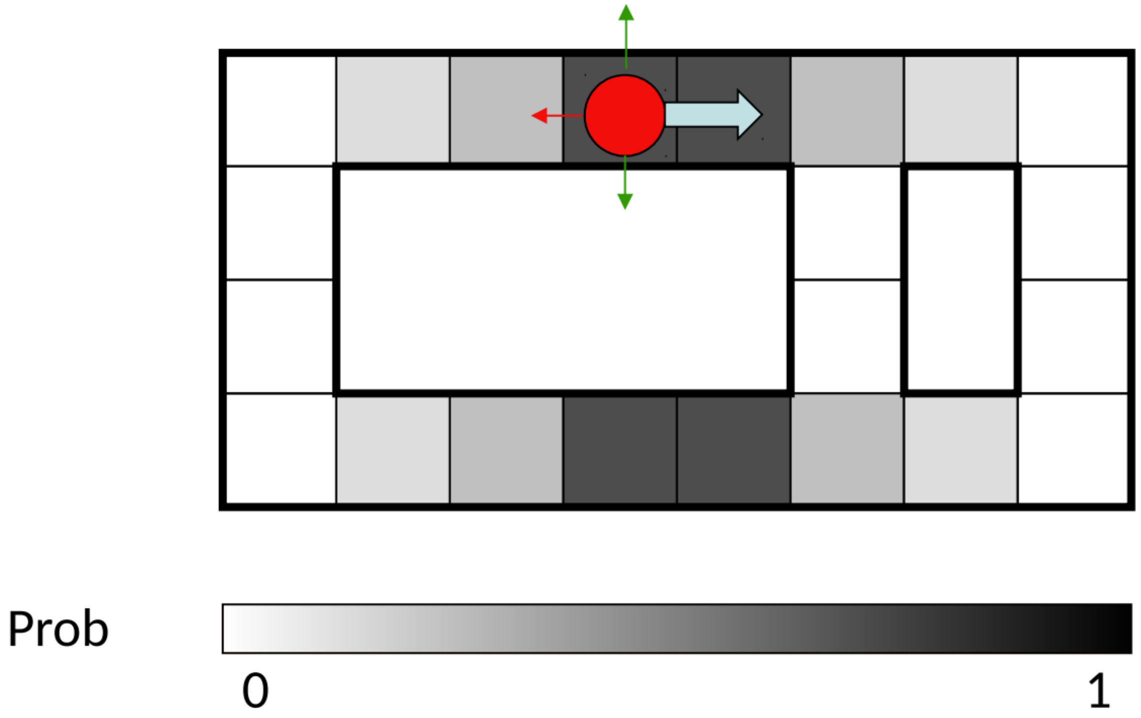
Robot perceives that there are walls above and below, but no walls either left or right

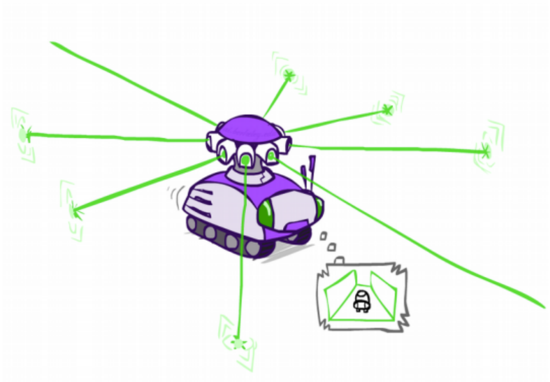




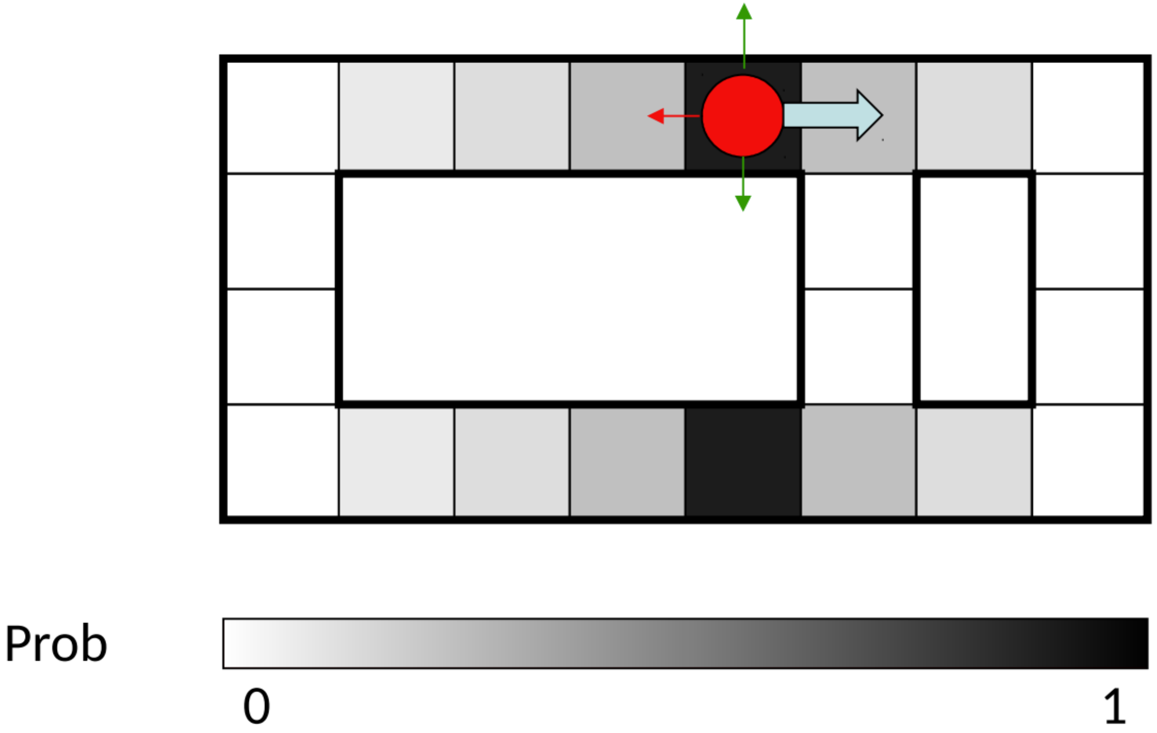
Gray level denotes estimated probability that robot is in that square

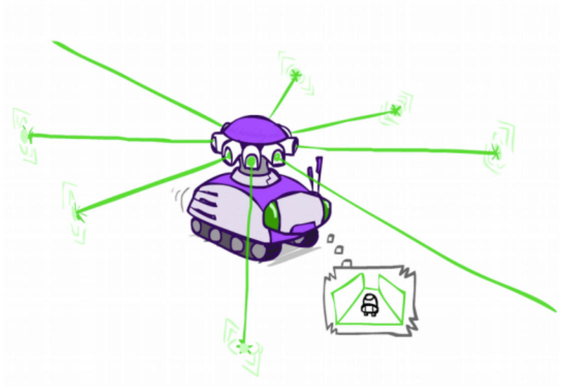
Example



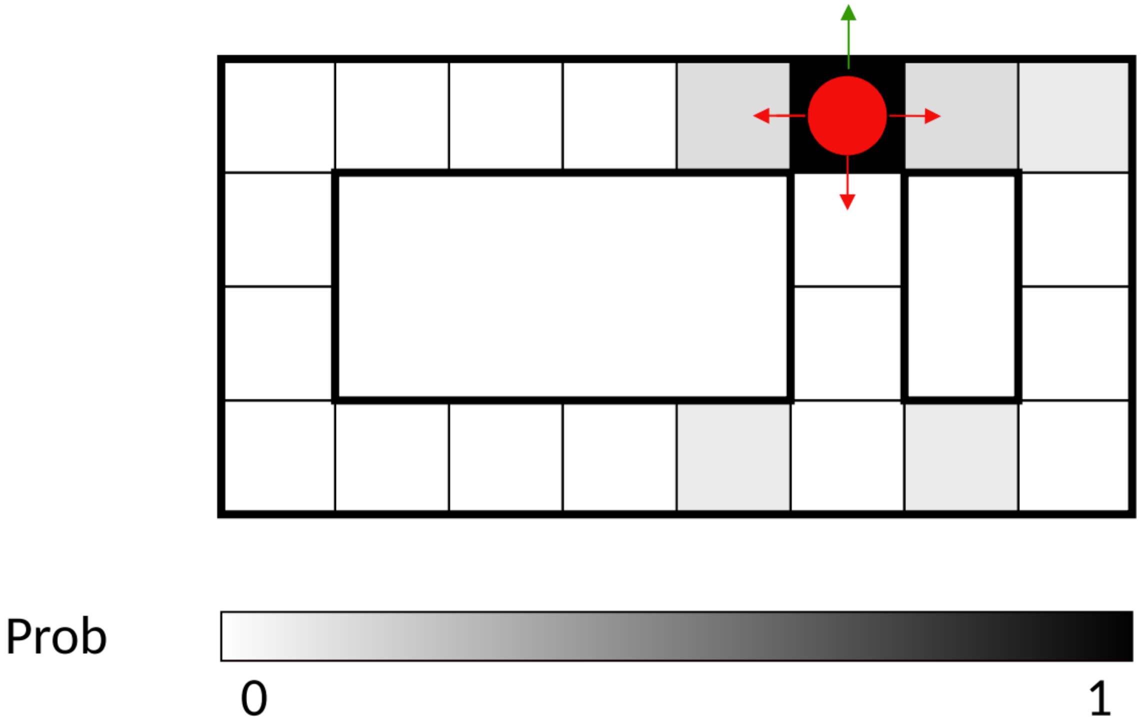


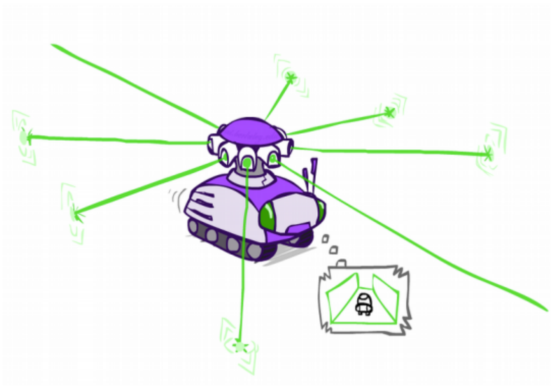
Example

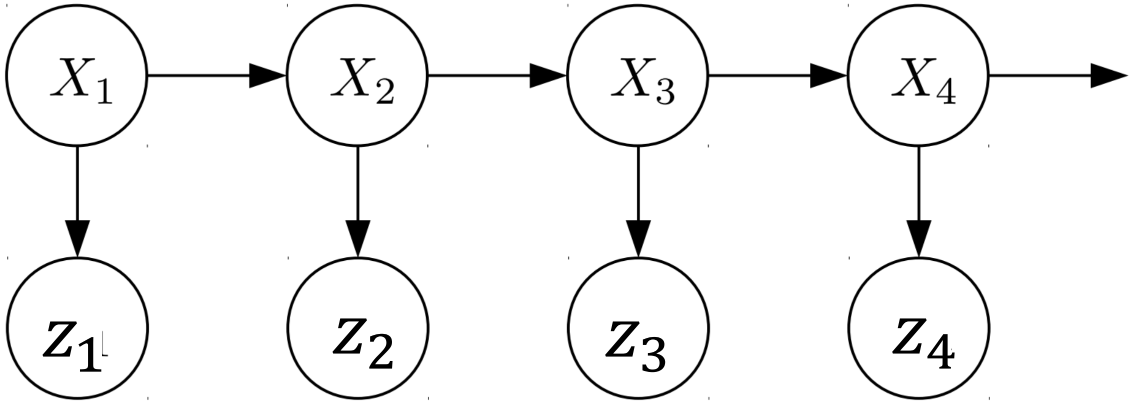




Example

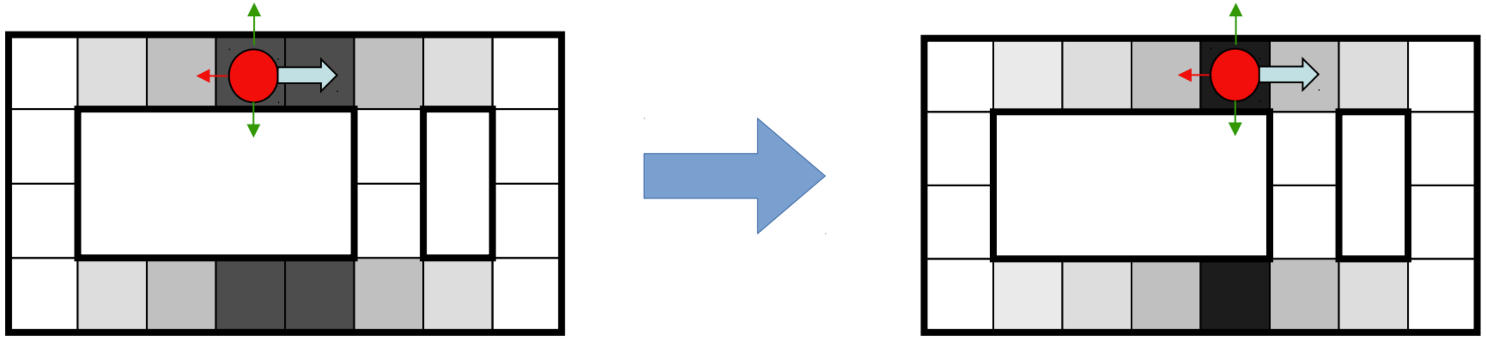


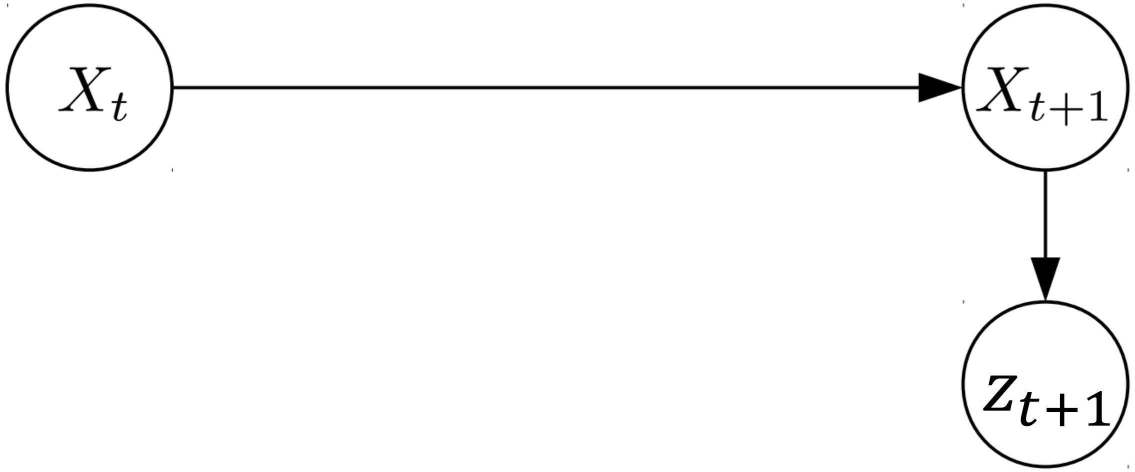


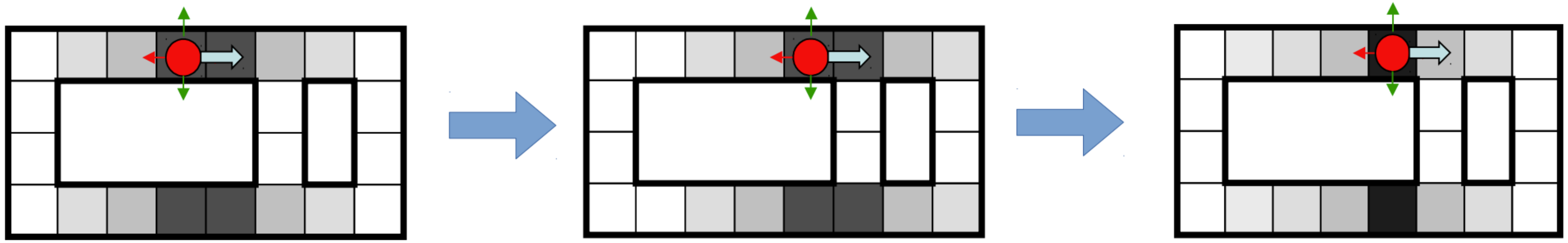


- The state X_t is unobserved and the one we would like to estimate
- z_t is the observation we can make with our sensors
- From the previous slides we have the prior (observation dynamics) and

$$P(z_t|x_t) \rightarrow \text{Observation dynamics}$$
$$P(x_t|x_{t-1}) \rightarrow \text{Environment dynamics}$$




X_t
X_{t+1}
z_{t+1}



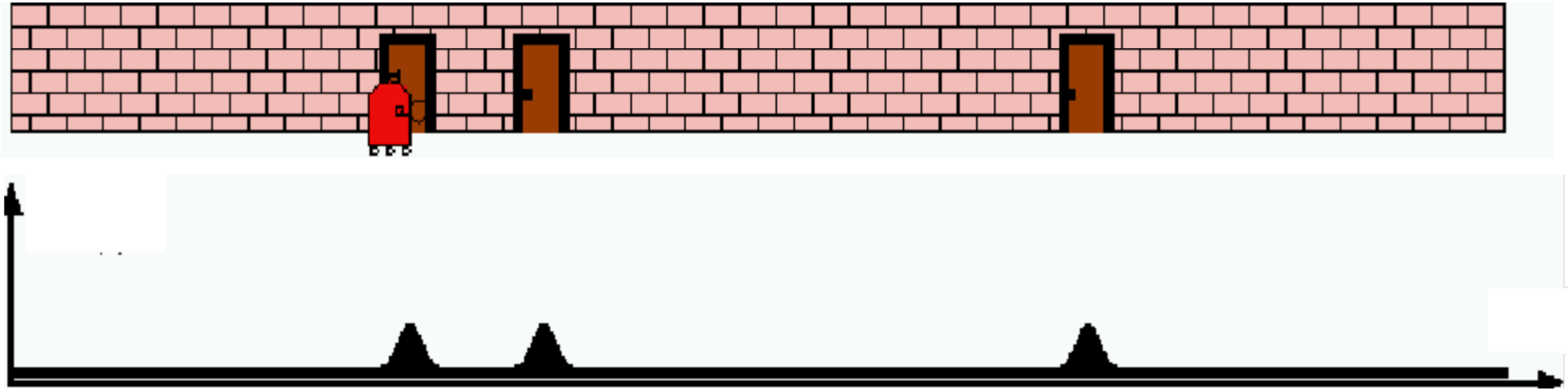
$$\begin{tikzcd} B(X_t) & B'(X_t) & B(X_{t+1}) \\ \boxed{P(X_t|z_{1:t})} \arrow[r] & \boxed{P(X_{t+1}|z_{1:t})} \arrow[r] & \boxed{P(X_{t+1}|z_{1:t+1})} \end{tikzcd}$$

Process update

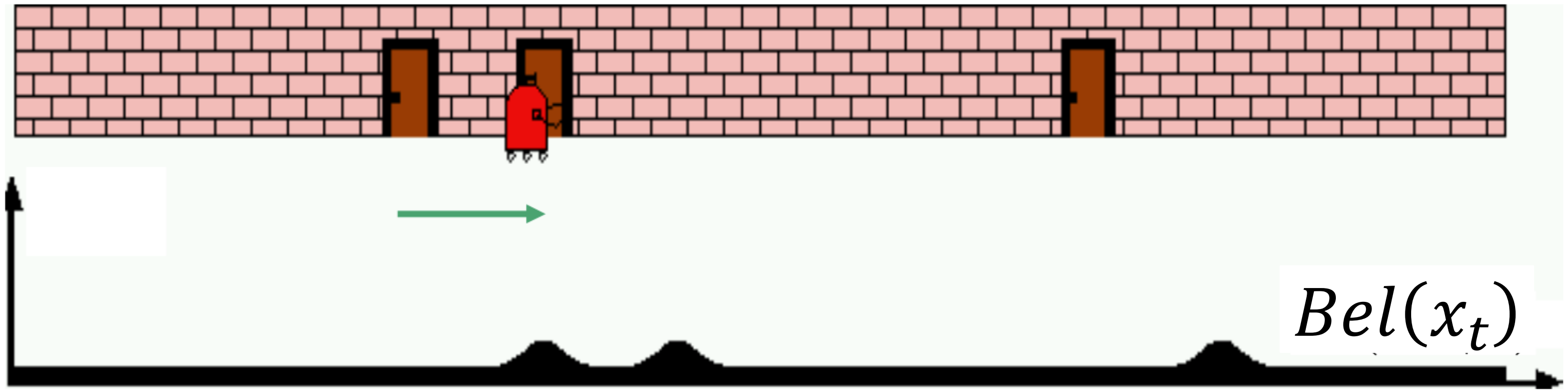
Observation update

- Predict the next state
- Correct based on observation

Before process update

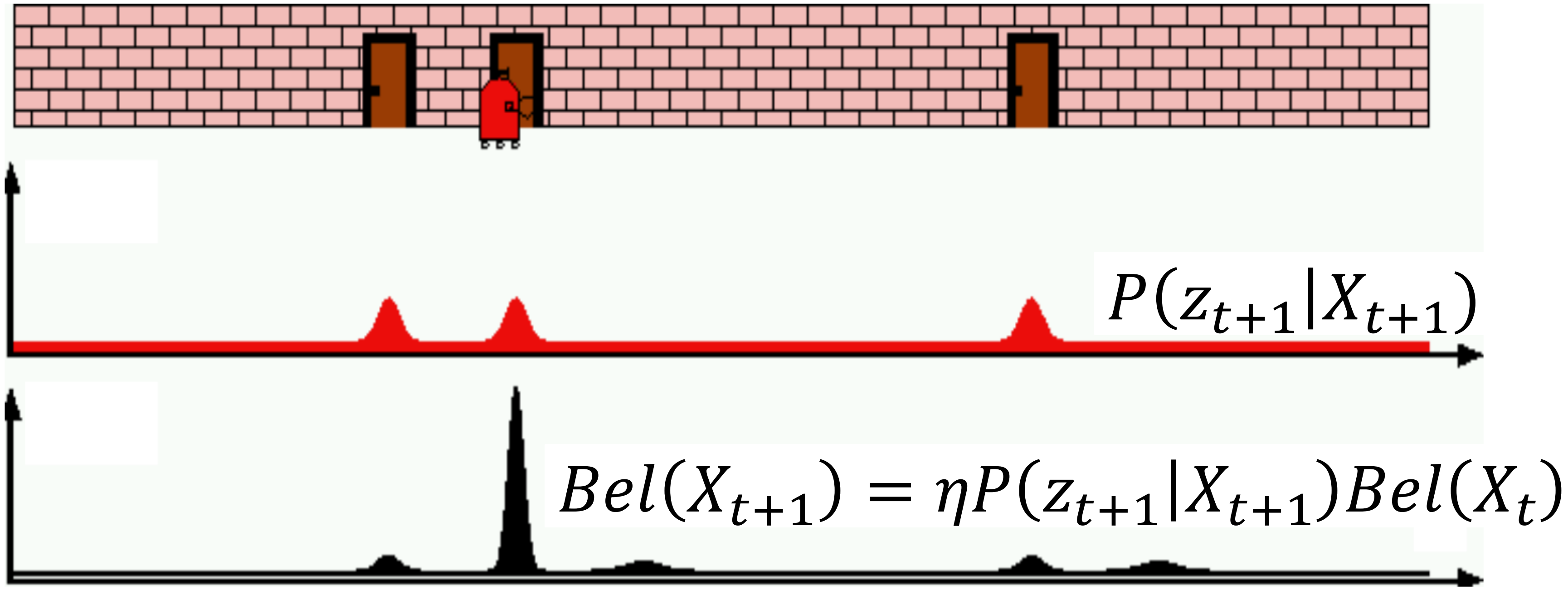


Before observation update



$$Bel(x_t)$$

After observation update

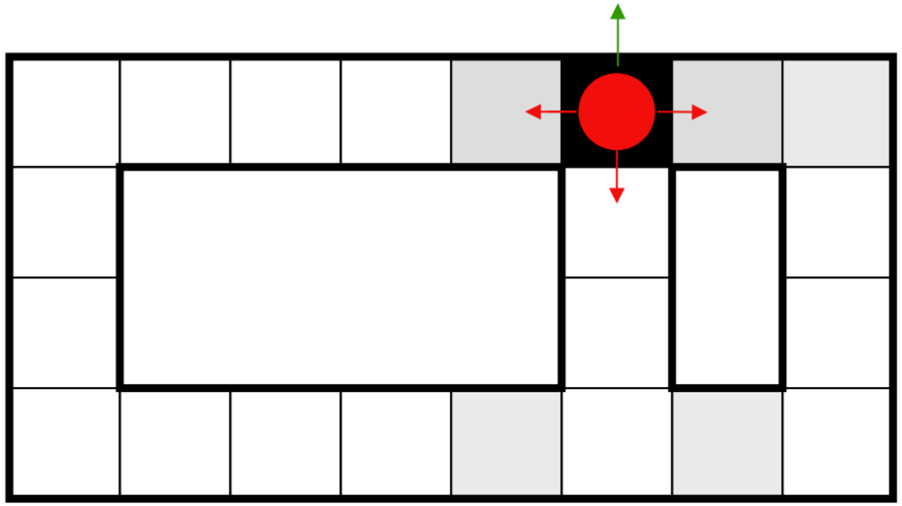


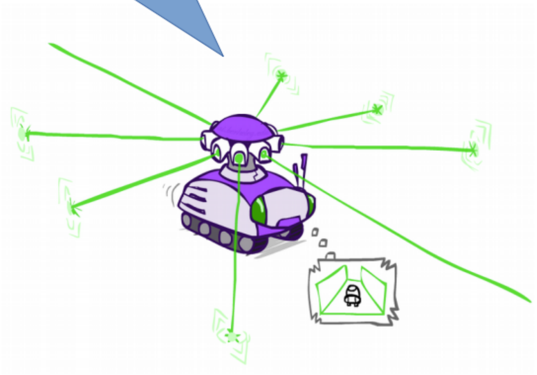
$$P(z_{t+1}|X_{t+1})$$

$$Bel(X_{t+1}) = \eta P(z_{t+1}|X_{t+1})Bel(X_t)$$

Assumptions we have made

Why must I be confined to this grid?





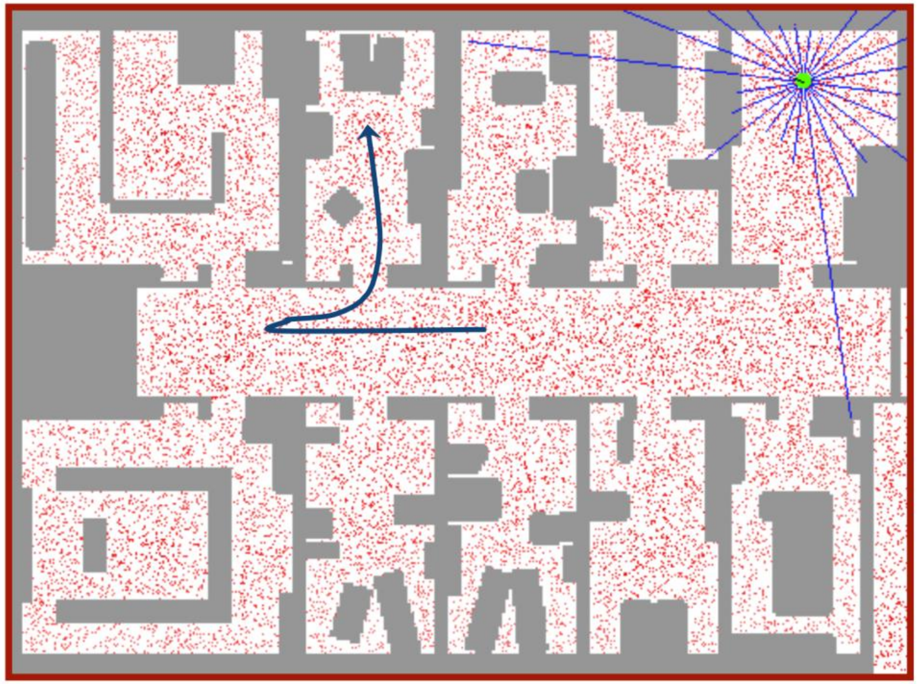
Particle filter localization

- Represents the location as a distribution of possible states
  - i.e. each particle is a hypothesis of where the robot is
  - Survival of the fittest (particles)

- The initial set of particle hypothesis can be uniformly distributed over the map

- Particle filtering is just an adaptation of the bayes filter using particles instead of grid cells

- Allows us to do non-gaussian distributions as well



Particle filter localization

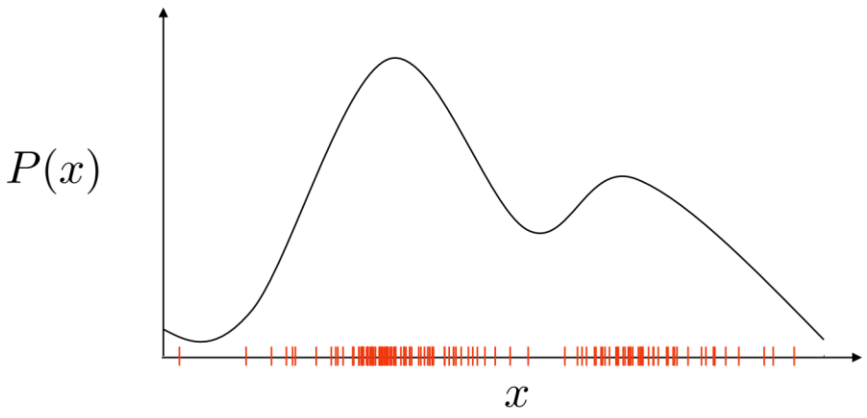
- Dense particles means higher probability mass

- Weight the importance of each particle to modulate our distribution

- Weighted particles
  - $S = \{(s^i, w^i) | i = 1, \dots, N\}$

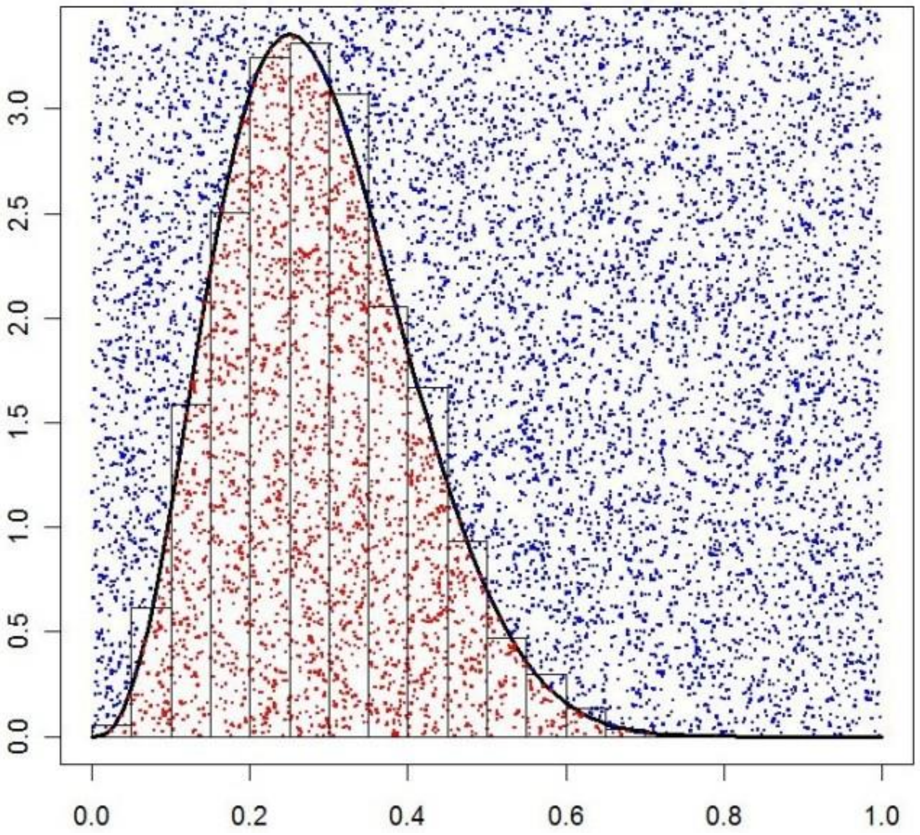
State hypothesis (particle)

Importance weight



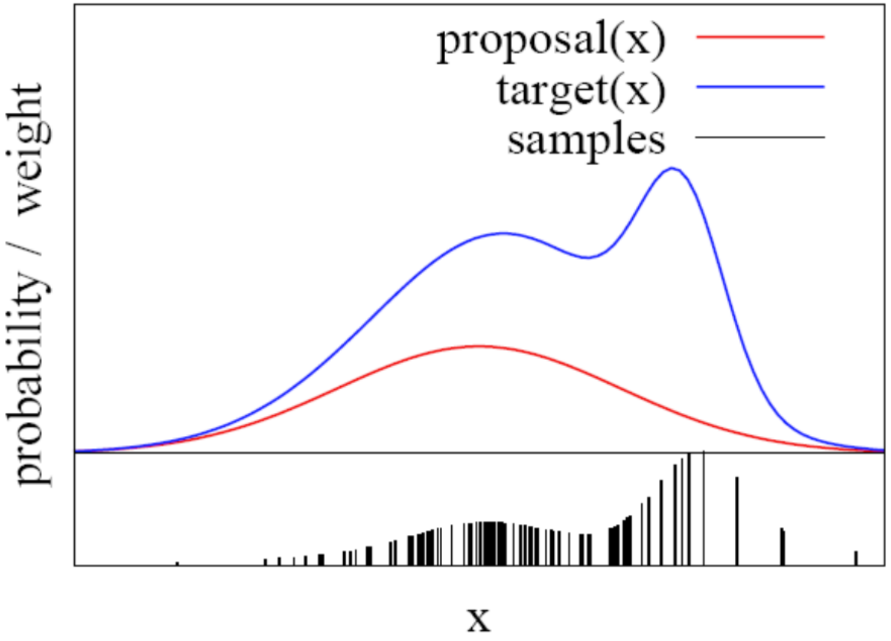
- The samples of hypotheses can then be our posterior
  - $P(x) = \sum_{i=1}^{N} w^i f(s^i)$

Sampling from a distribution



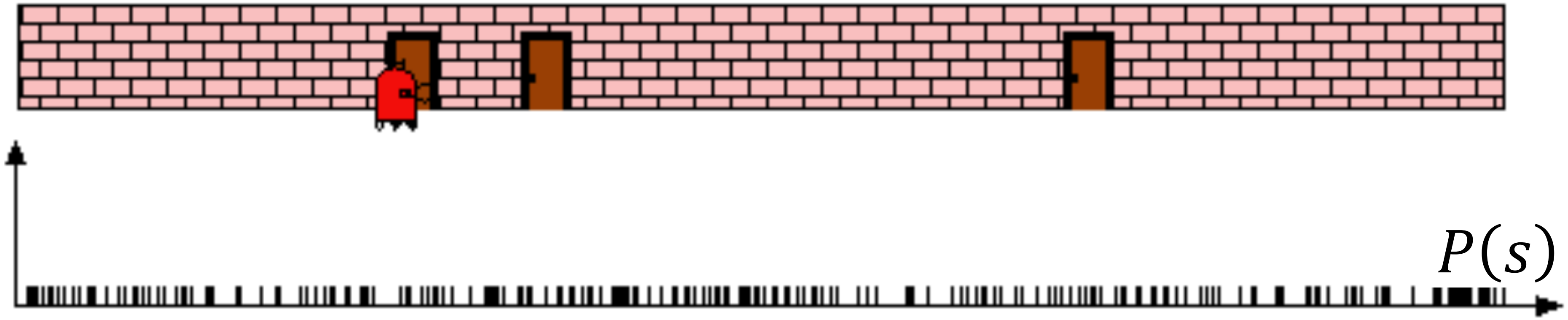
- Rejection sampling
  - Sample x from a uniform distribution
  - Sample c from $[0,1]$
  - If $f(x) > c$ keep the sample, reject otherwise

Sampling from a distribution



- We can even use a different distribution to generate sample from f
  - This is where importance sampling comes in
  - Account for differences in the two distributions:
    - $w = \frac{f}{g}$
    - We upscale the chance of selecting a sample from our proposal distribution according to our target distribution

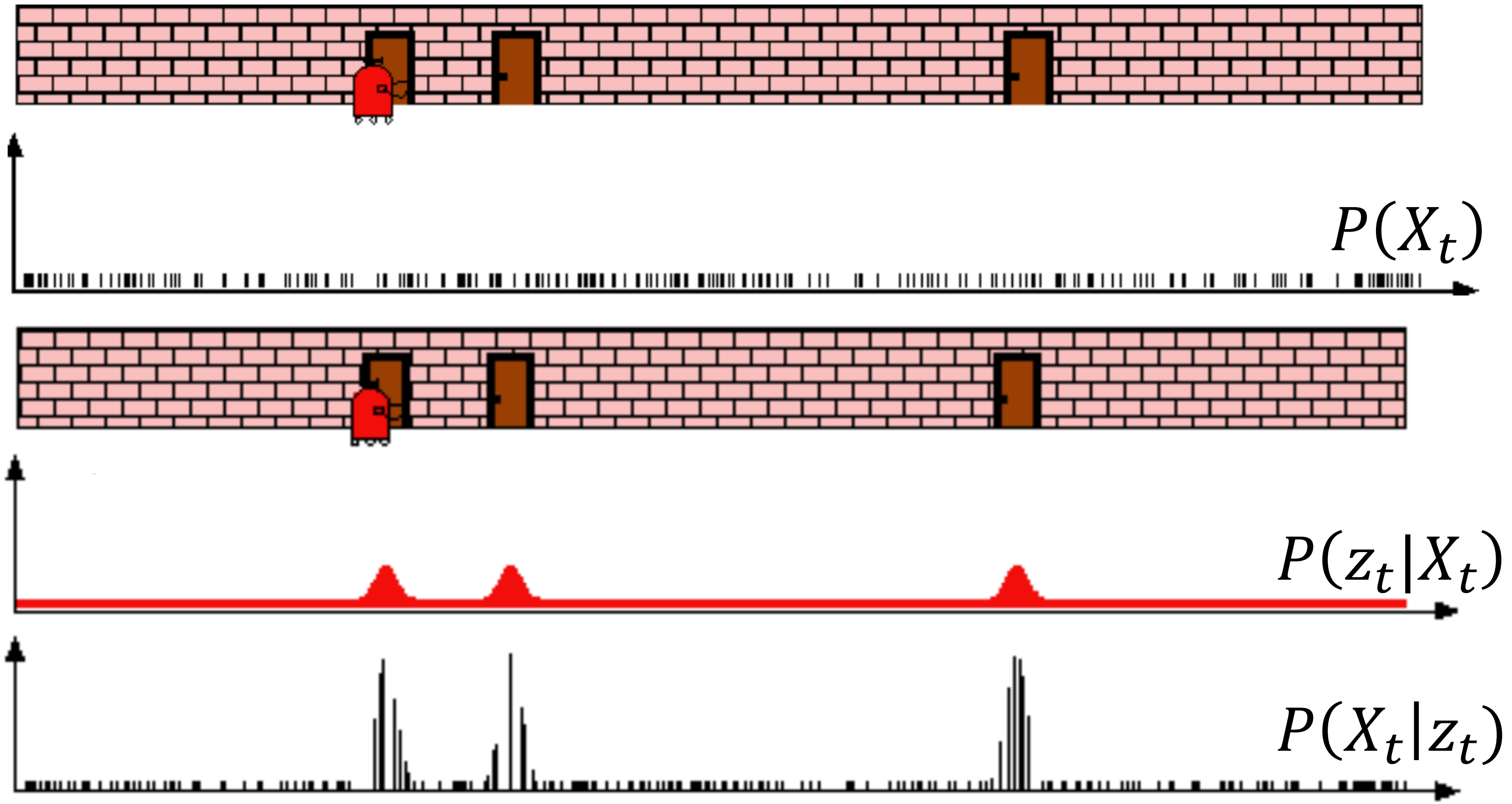
Example



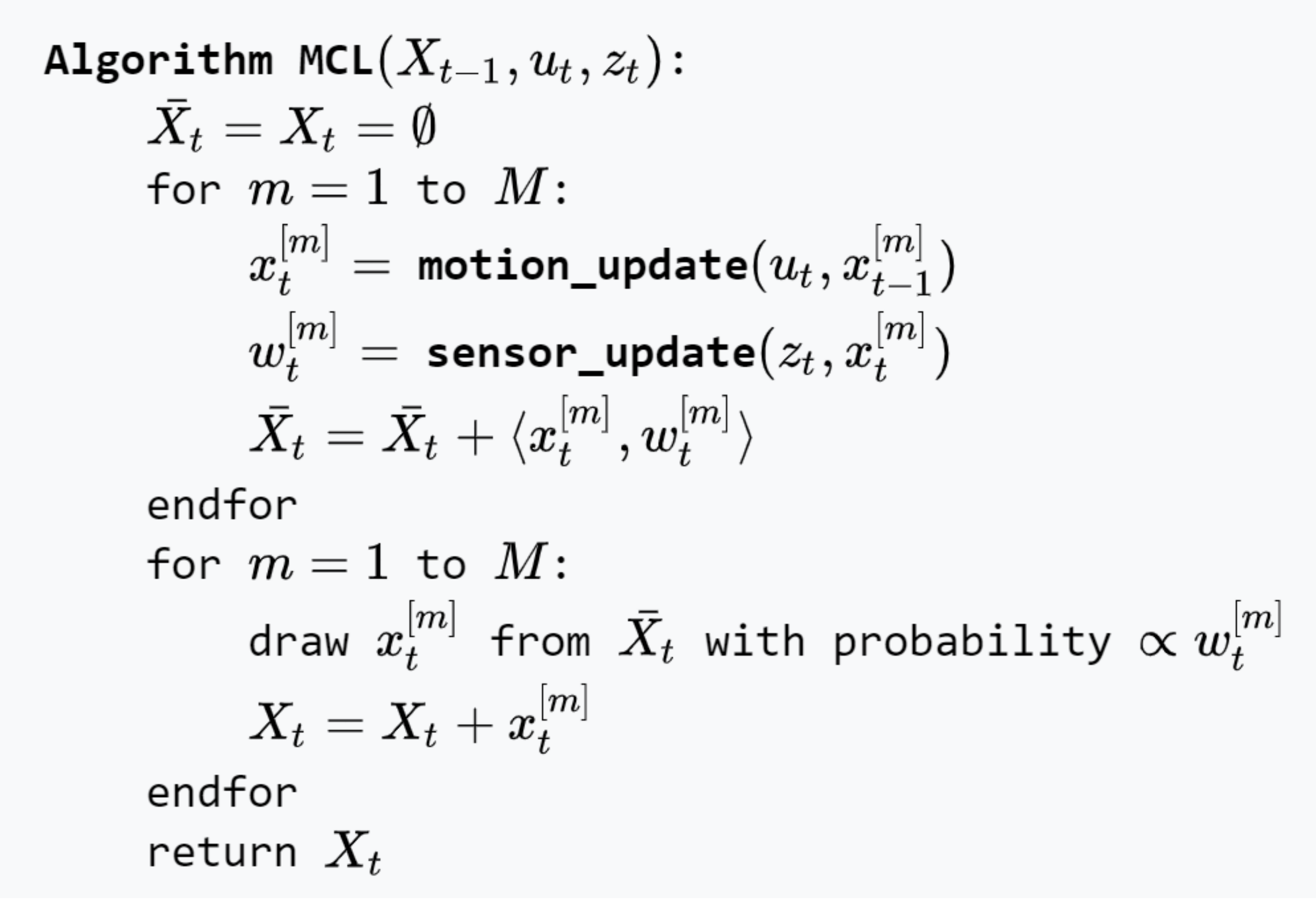
Example

$$Bel(x_{t+1}) = \eta P(z_{t+1}|X_{t+1})Bel(X_t)$$

$$w = \frac{\eta P(z_{t+1}|X_{t+1})Bel(X_t)}{Bel(X_{t+1})} = \eta P(z_{t+1}|X_{t+1})$$



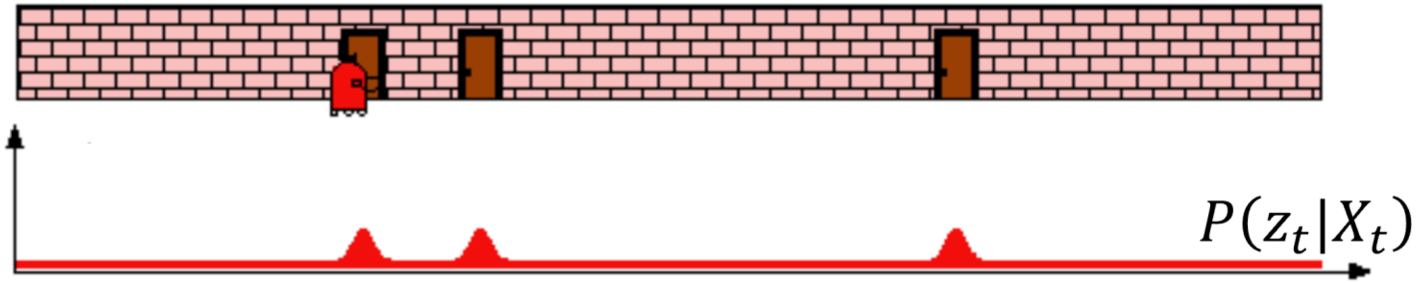
The particle filter algorithm

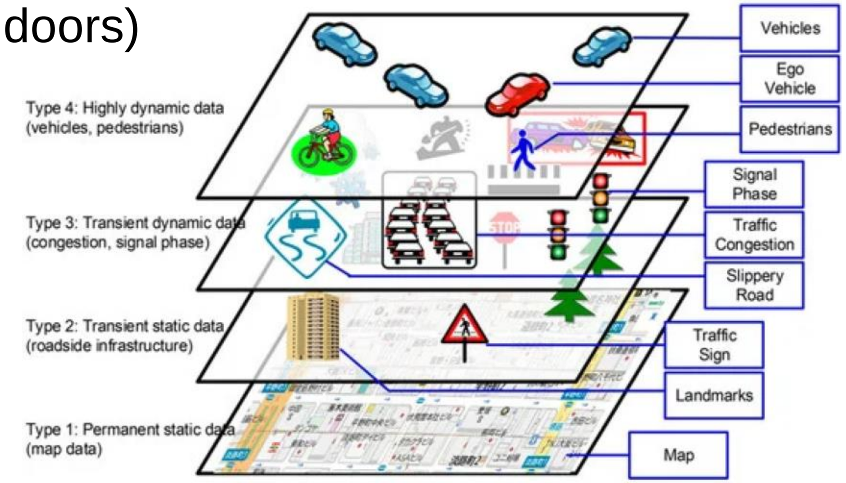


```
Algorithm MCL($(X_{t-1}, u_t, z_t)$):
    $\bar{X}_t = X_t = \emptyset$
    for $m = 1$ to $M$:
        $x_t^{[m]} =$ motion_update($(u_t, x_{t-1}^{[m]})$)
        $w_t^{[m]} =$ sensor_update($(z_t, x_t^{[m]})$)
        $\bar{X}_t = \bar{X}_t + \langle x_t^{[m]}, w_t^{[m]} \rangle$
    endfor
    for $m = 1$ to $M$:
        draw $x_t^{[m]}$ from $\bar{X}_t$ with probability $\propto w_t^{[m]}$
        $X_t = X_t + x_t^{[m]}$
    endfor
    return $X_t$
```

What do we need to be aware of?

- Distinctive features in the environment are necessary
  - To do the sensor update, we need to compute $P(z_t|x_t)$ which require us to quantify the state
  - This is done through features in the environment (like the doors)




Type 4: Highly dynamic data (vehicles, pedestrians)
Type 3: Transient dynamic data (congestion, signal phase)
Type 2: Transient static data (roadside infrastructure)
Type 1: Permanent static data (map data)
Vehicles
Ego Vehicle
Pedestrians
Signal Phase
Traffic Congestion
Slippery Road
Traffic Sign
Landmarks
Map

- What if the environment is very dynamic?
  - Utilize the mapping abstractions, such as HD map representations

- High sensitivity to the number of points!

- We need to move – i.e. we need to change the state to get updates

Exercises

- Using your own localization and accumulate a map when you drive around in the environment
  - Use a counter to define if a cell is free or occupied

- The simulation we are using already uses a particle filter to perform localization
  - Set an initial starting point of the robot in rviz using the “2D pose estimate” button
  - Create a ROS2 node that:
    - Prints the number of particles used
    - Computes the expected position of the robot based on the particles
  - Change the number of particles when you launch the simulation

- Implement your own particle filter for localization in a map
  - Use the cmd_topic to estimate your motion model
  - Subscribe to the LIDAR topic and compute features
  - For each particle compute the error for each feature
  - Update your importance weights based on the error