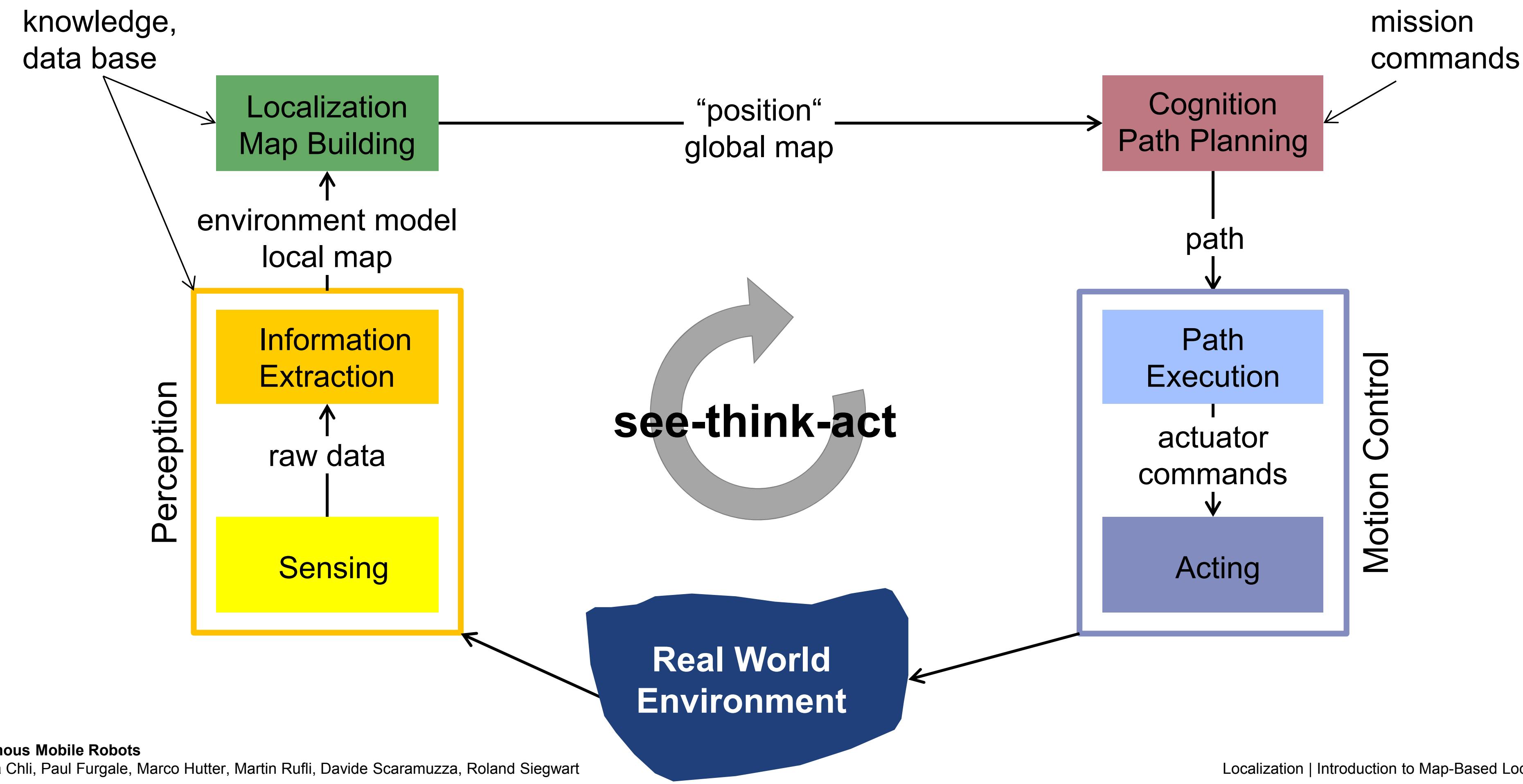
# Localization | **Introduction to Map-Based Localization**

*Autonomous Mobile Robots*

**Roland Siegwart**

Margarita Chli, Paul Furgale, Marco Hutter, Martin Rufli, Davide Scaramuzza

# Introduction | probabilistic map-based localization



# **Localization |** definition, challenges and approach

- Map-based localization
  - The robot estimates its position using perceived information and a map
  - The map
    - might be known (localization)
    - Might be built in parallel (simultaneous localization and mapping – SLAM)
- Challenges
  - Measurements and the map are inherently error prone
  - Thus the robot has to deal with uncertain information

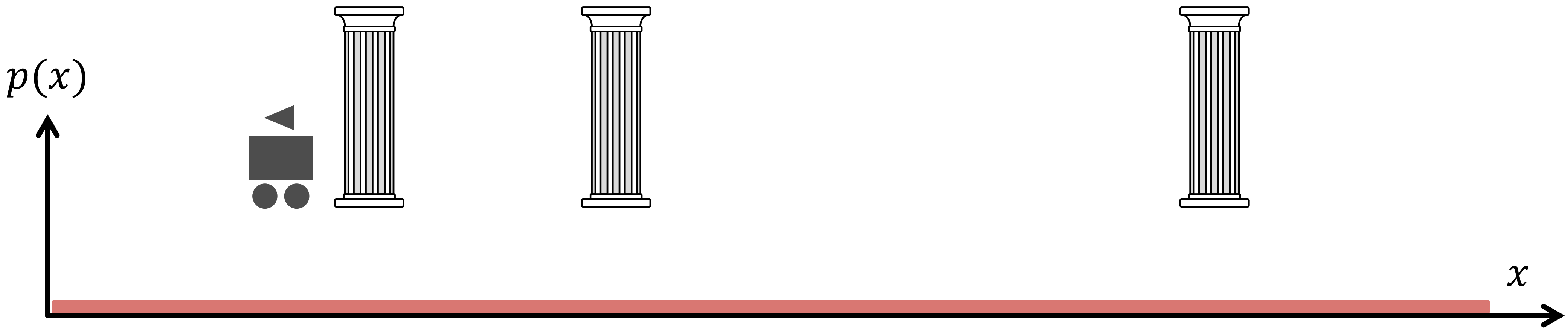
  → Probabilistic map-base localization
- Approach
  - The robot estimates the belief state about its position through an ACT and SEE cycle

# Robot Localization: Historical Context

- Initially, roboticists thought the world could be modeled <u>exactly</u>
- Path planning and control assumed perfect, exact, deterministic world
- Reactive robotics (behavior based, ala bug algorithms) were developed due to imperfect world models
- But Reactive robotics assumes accurate control and sensing to react – also not realistic
- Reality: imperfect world models, imperfect control, imperfect sensing
- Solution: Probabilistic approach, incorporating model, sensor and control uncertainties into localization and planning
- Reality: these methods work empirically!

# **Concept |** SEE and ACT to improve belief state

- Robot is placed somewhere in the environment → location unknown
- SEE: The robot queries its sensors → finds itself next to a pillar
- ACT: Robot moves one meter forward
  - motion estimated by wheel encoders
  - accumulation of uncertainty
- SEE: The robot queries its sensors again → finds itself next to a pillar
- Belief updates (information fusion)



# **Concept |** SEE and ACT to improve belief state

- Robot is placed somewhere in the environment → location unknown
- SEE: The robot queries its sensors
  → finds itself next to a pillar
- ACT: Robot moves one meter forward
  - motion estimated by wheel encoders
  - accumulation of uncertainty
- SEE: The robot queries its sensors again
  → finds itself next to a pillar
- Belief updates (information fusion)

# **Concept |** SEE and ACT to improve belief state

- Robot is placed somewhere in the environment → location unknown
- SEE: The robot queries its sensors → finds itself next to a pillar

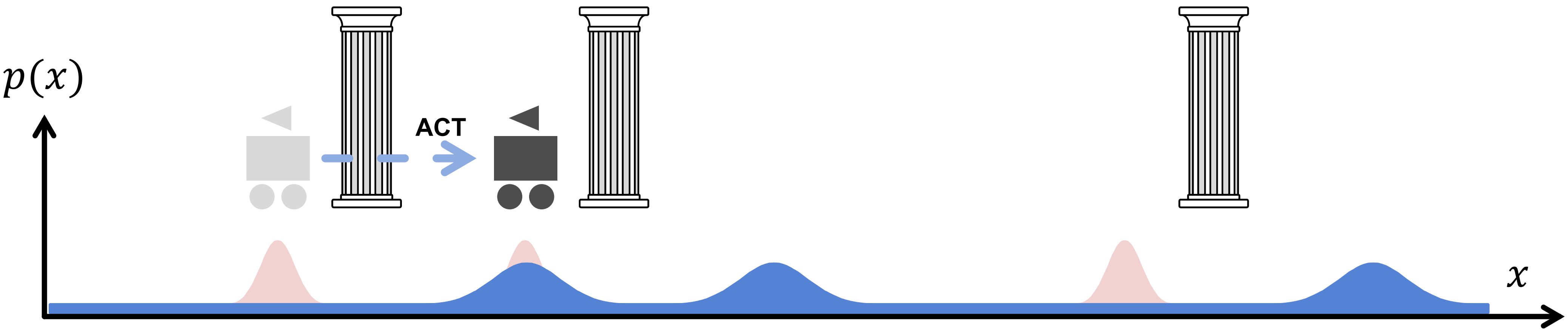
- ACT: Robot moves one meter forward
  - motion estimated by wheel encoders
  - accumulation of uncertainty
- SEE: The robot queries its sensors again → finds itself next to a pillar
- Belief updates (information fusion)

# **Concept |** SEE and ACT to improve belief state

- Robot is placed somewhere in the environment → location unknown
- SEE: The robot queries its sensors → finds itself next to a pillar
- ACT: Robot moves one meter forward
  - motion estimated by wheel encoders
  - accumulation of uncertainty
- SEE: The robot queries its sensors again → finds itself next to a pillar
- Belief updates (information fusion)

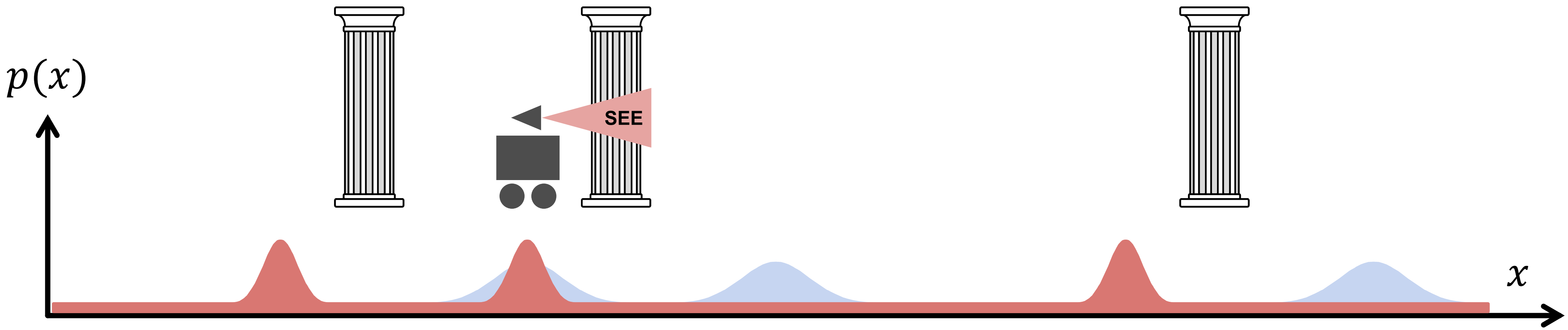
# **Concept |** SEE and ACT to improve belief state

- Robot is placed somewhere in the environment → location unknown
- SEE: The robot queries its sensors → finds itself next to a pillar
- ACT: Robot moves one meter forward
  - motion estimated by wheel encoders
  - accumulation of uncertainty
- SEE: The robot queries its sensors again → finds itself next to a pillar
- Belief updates (information fusion)



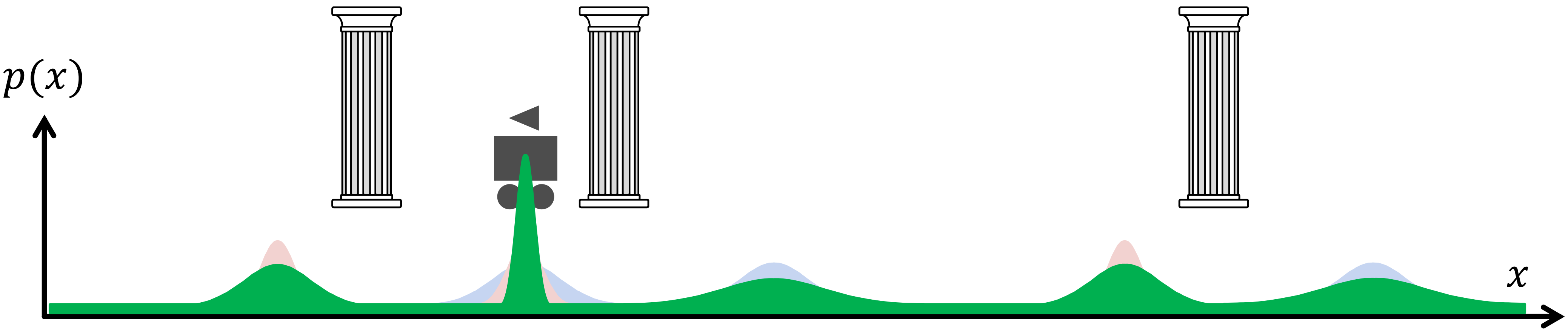
# **Concept |** SEE and ACT to improve belief state

- Robot is placed somewhere in the environment → location unknown
- SEE: The robot queries its sensors → finds itself next to a pillar
- ACT: Robot moves one meter forward
  - motion estimated by wheel encoders
  - accumulation of uncertainty
- SEE: The robot queries its sensors again → finds itself next to a pillar
- Belief updates (information fusion)



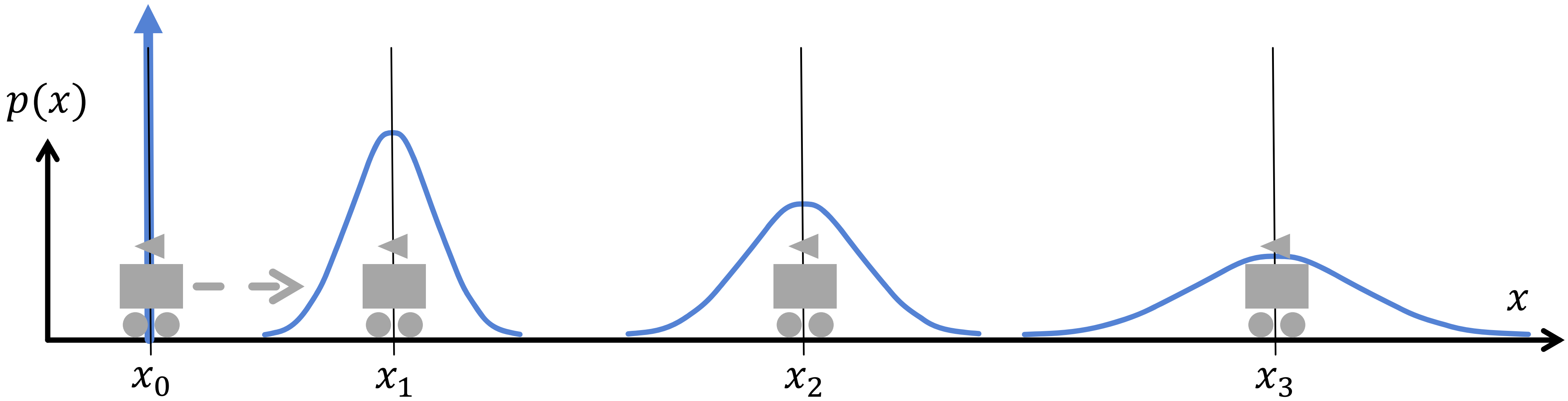
# Concept | SEE and ACT to improve belief state

- Robot is placed somewhere in the environment → location unknown
- SEE: The robot queries its sensors → finds itself next to a pillar
- ACT: Robot moves one meter forward
  - motion estimated by wheel encoders
  - accumulation of uncertainty
- SEE: The robot queries its sensors again → finds itself next to a pillar
- Belief update (information fusion)



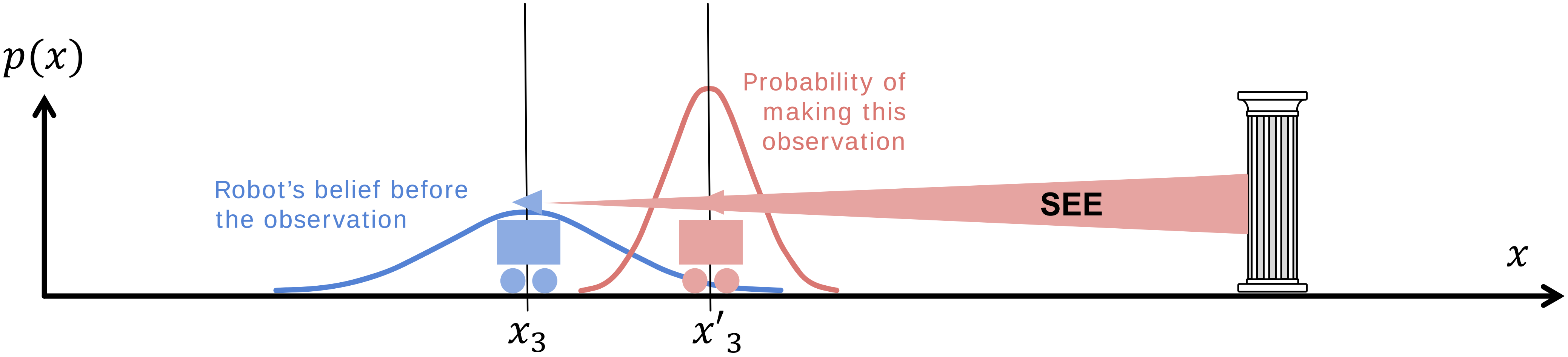
# ACT | using motion model and its uncertainties

- The robot moves and estimates its position through its proprioceptive sensors
  - Wheel Encoder (Odometry)
- During this step, the robot's state uncertainty grows



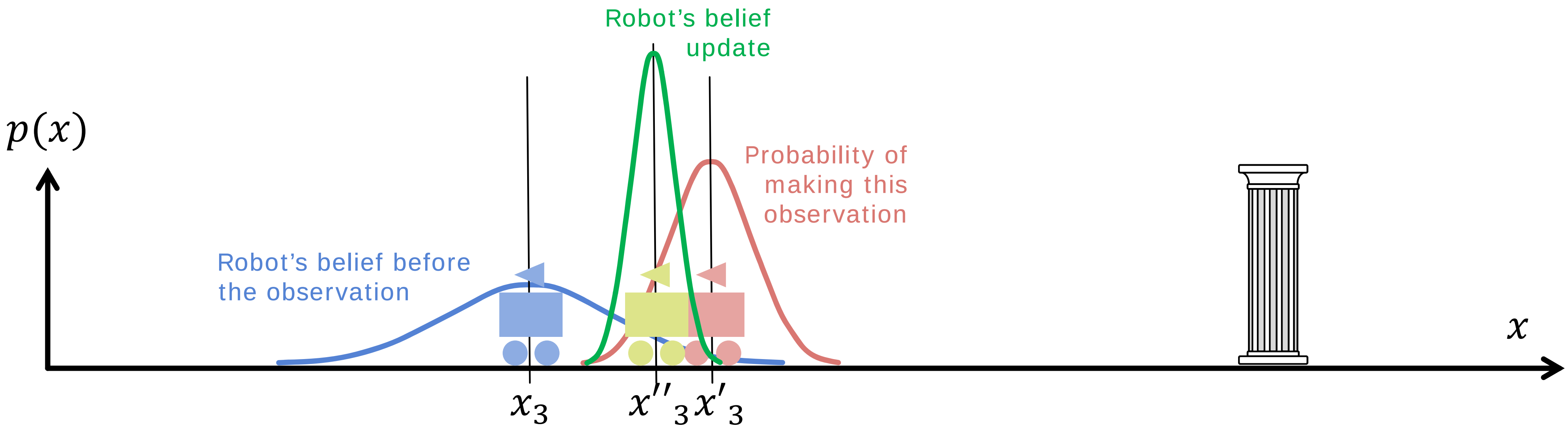
# SEE | estimation of position based on perception and map

- The robot makes an observation using its exteroceptive sensors
- This results in a second estimation of the current position

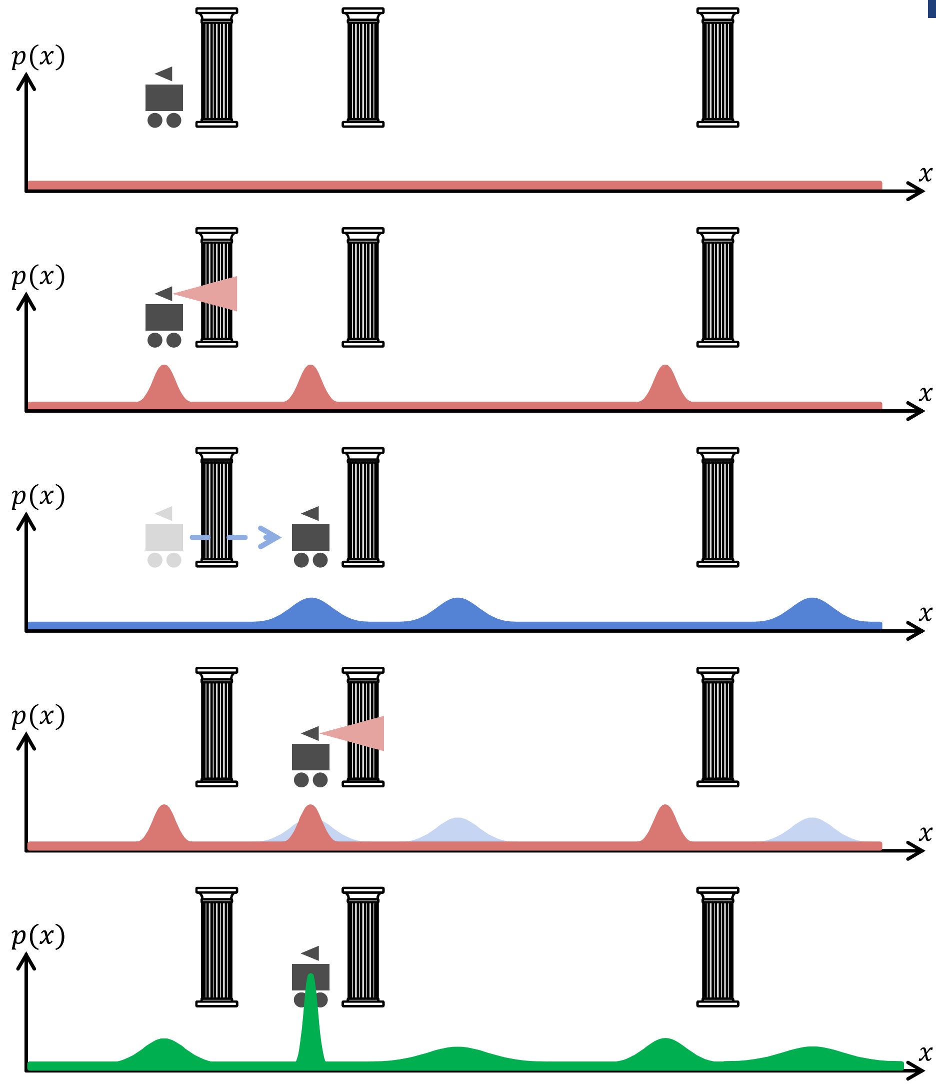


# **Belief update |** fusion of prior belief with observation

- The robot corrects its position by combining its belief before the observation with the probability of making exactly that observation
- During this step, the robot's state uncertainty shrinks



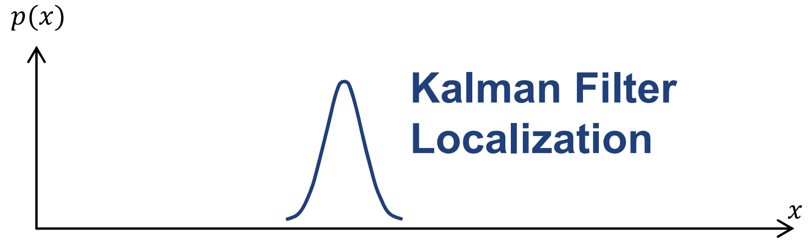
# Take home message |
ACT - SEE Cycle for Localization



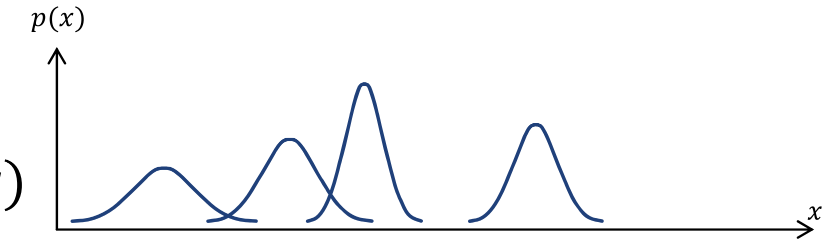

- SEE: The robot queries its sensors → finds itself next to a pillar
- ACT: Robot moves one meter forward
  - motion estimated by wheel encoders
  - accumulation of uncertainty
- SEE: The robot queries its sensors again → finds itself next to a pillar
- Belief update (information fusion)

# **Probabilistic localization** | belief representation

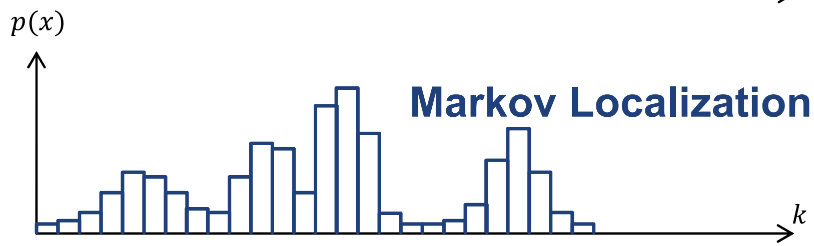
a) Continuous map with
single hypothesis probability distribution $p(x)$



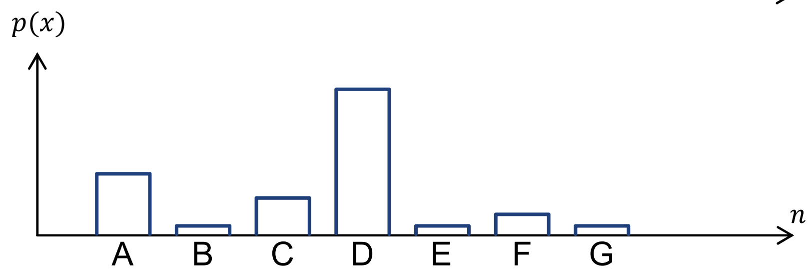

b) Continuous map with
multiple hypotheses probability distribution $p(x)$



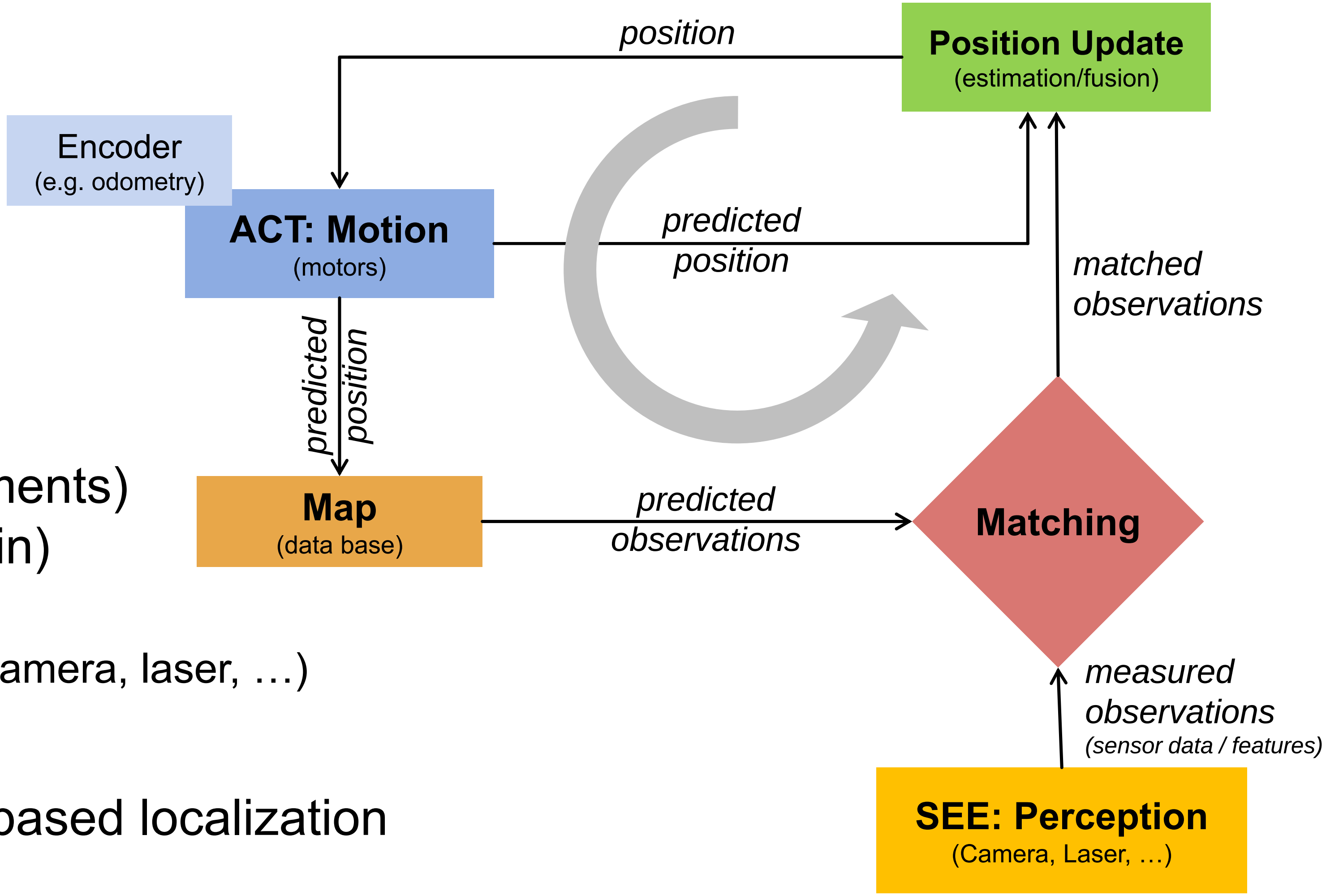

c) Discretized metric map (grid $k$) with
probability distribution $p(k)$




d) Discretized topological map (nodes $n$) with
probability distribution $p(n)$



# **Markov localization** | applying probability theory to localization




- Information (measurements) is error prone (uncertain)
  - Odometry
  - Exteroceptive sensors (camera, laser, …)
  - Map

→ Probabilistic map-based localization

# **Usage** | application of probability theory to robot localization

- Probability theory is widely and very successfully used for mobile robot localization

- In the following lecture segments, its application to localization will be illustration
  - Markov localization
    - Discretized pose representation
  - Kalman filter
    - Continuous pose representation and Gaussian error model

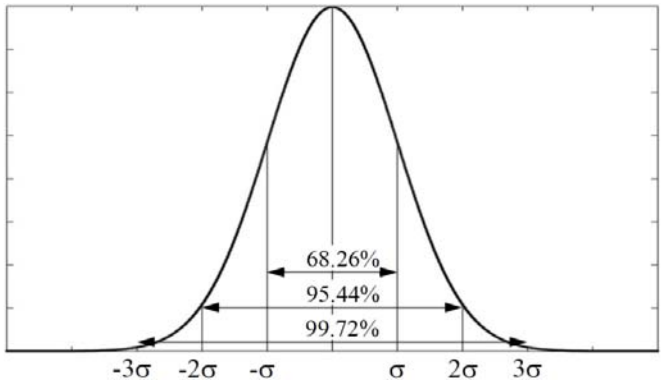
- Further reading:
  - “Probabilistic Robotics,” Thrun, Fox, Burgard, MIT Press, 2005.
  - “Introduction to Autonomous Mobile Robots”, Siegwart, Nourbakhsh, Scaramuzza, MIT Press 2011

# **Probability theory** | how to deal with uncertainty

- Mobile robot localization has to deal with error prone information
- Mathematically, error prone information (uncertainties) is best represented by random variables and probability theory

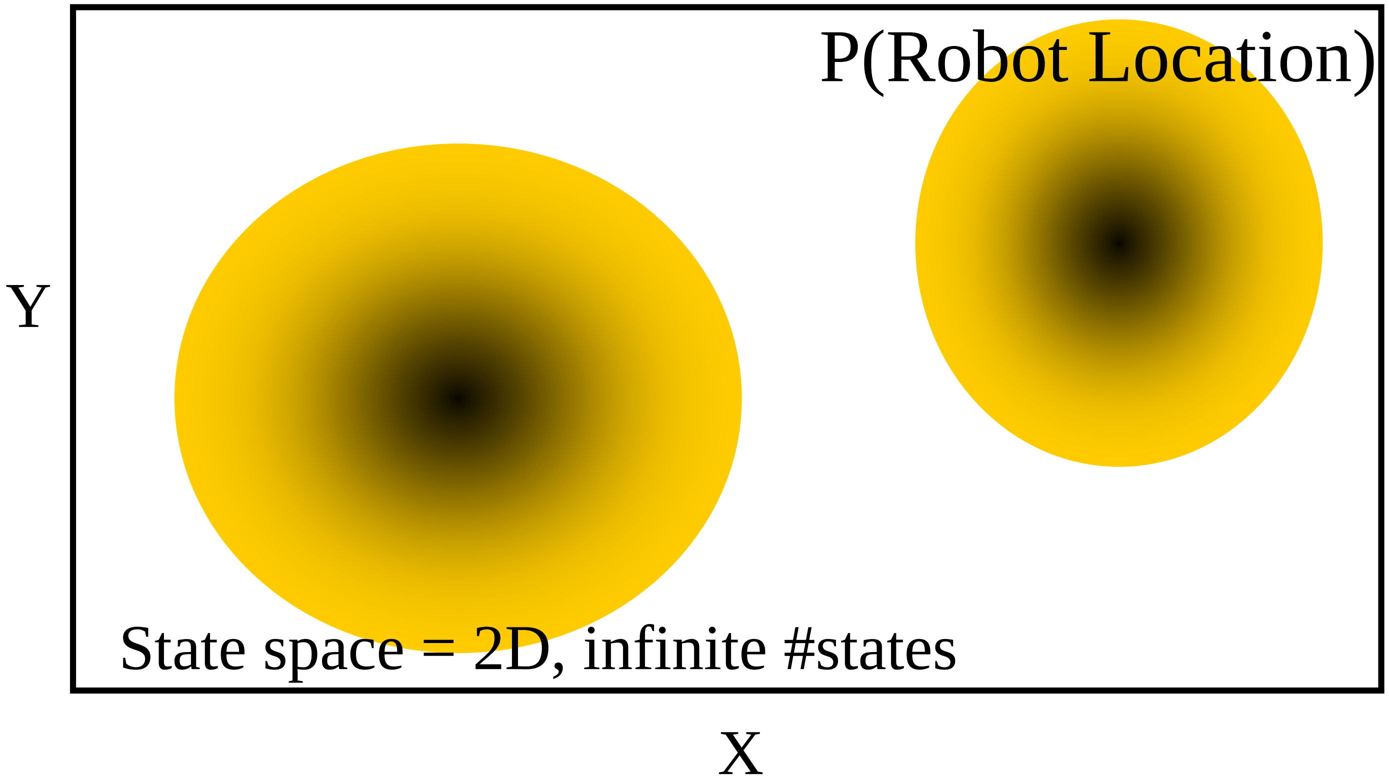
- $p(x) = p(X = x)$: probability that the random variable $X$ has value $x$ ($x$ is true).
  - $X$: random variable
  - $x$: a specific value that $X$ might assume.
  - The **Probability Density Functions** (PDF) describes the relative likelihood for a random variable to take on a given value
  - PDF example: The Gaussian distribution:

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

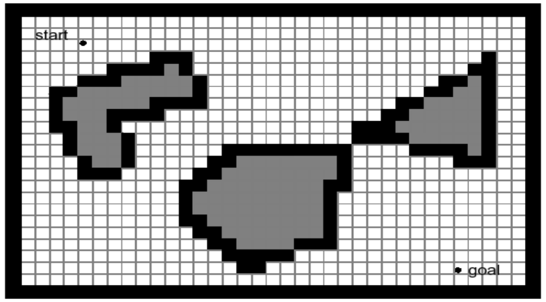


# Markov Localization

- Key idea: compute a probability distribution over all possible positions in the environment.
  - *This probability distribution represents the likelihood that the robot is in a particular location.*



# **Markov localization** | basics and assumption




- Discretized pose representation $x_t \rightarrow$ grid map

- Markov localization tracks the robot's belief state $bel(x_t)$ using an arbitrary probability density function to represent the robot's position

- *Markov assumption*: Formally, this means that the output of the estimation process is a function $x_t$ only of the robot's previous state $x_{t-1}$ and its most recent actions (odometry) $u_t$ and perception $z_t$.

$$p(x_t | x_0, u_t \cdots u_0, z_t \cdots z_0) = p(x_t | x_{t-1}, u_t, z_t)$$

- Markov localization addresses the *global localization problem*, the *position tracking problem*, and the *kidnapped robot problem*.

# **Basic concepts of probability theory** | theorem of total probability

- The **theorem of total probability** (*convolution*) originates from the axioms of probability theory and is written as:

$$p(x) = \sum_y p(x|y)p(y)$$ for discrete probabilities

$$p(x) = \int_y p(x|y)p(y)dy$$ for continuous probabilities

- This theorem is used by both *Markov* and *Kalman-filter* localization algorithms during the prediction update.

# **Markov localization** | applying probability theory to localization

- **ACT** | probabilistic estimation of the robot's new belief state $\overline{bel}\,(x_t)$ based on the previous location $bel(x_{t-1})$ and the probabilistic motion model $p(x_t|u_t, x_{t-1})$ with action $u_t$ (control input).

→ application of ***theorem of total probability*** **/ *convolution***

$$\overline{bel}(x_t) = \int p(x_t|u_t, x_{t-1}) bel(x_{t-1})\, dx_{t-1}$$ for continuous probabilities

$$\overline{bel}(x_t) = \sum_{x_{t-1}} p(x_t|u_t, x_{t-1}) bel(x_{t-1})$$ for discrete probabilities

# **Markov localization** | applying probability theory to localization

- **SEE** | probabilistic estimation of the robot's new belief state $bel(x_t)$ as a function of its measurement data $z_t$ and its former belief state $\overline{bel}\,(x_t)$:

→ application of ***Bayes rule***

$$bel(x_t) = \eta p(z_t|x_t, M)\overline{bel}(x_t)$$

where $p(z_t|x_t, M)$ is the probabilistic measurement model (SEE), that is, the probability of observing the measurement data $z_t$ given the knowledge of the map $M$ and the robot's position $x_t$. Thereby $\eta = p(y)^{-1}$ is the normalization factor so that $\sum p = 1$.

# Markov Localization makes use of Bayes Rule

- P(A): Probability that A is true.
  - *e.g. $p(r_t = l)$: probability that the robot r is at position l at time t*

- We wish to compute the probability of each individual robot position given actions and sensor measures.
- P(A|B): Conditional probability of A given that we know B.
  - *e.g. $p(r_t = l \mid i_t)$: probability that the robot is at position l given the sensors input $i_t$.*
- Product rule:

$$p(A \wedge B) = p(A|B)p(B)$$

$$p(A \wedge B) = p(B|A)p(A)$$

- Bayes rule:

$$p(A|B) = \frac{p(B|A)p(A)}{p(B)}$$

# The "See" update step

- Bayes rule:

$$p(A|B) = \frac{p(B|A)p(A)}{p(B)}$$

  - *"See" operation: Maps from a belief state and a sensor input to a refined belief state:*

$$p(l|i) = \frac{p(i|l)p(l)}{p(i)} \qquad (5.21)$$

$$s_t = See(i_t, s_t')$$

  - *p(l): belief state before perceptual update process*
  - *p(i |l): probability we get measurement i when being at position l*
    - *To obtain this info: consult robot's map and identify the probability of a certain sensor reading if the robot were at position l*
  - *p(i): normalization factor so that sum over all l equals* 1.

- We apply this operation to all possible robot positions, *l*

# **Basic concepts of probability theory** | the Bayes rule

- The **Bayes rule** relates the conditional probability $p(x|y)$ to its inverse $p(y|x)$.
- Under the condition that $p(y) > 0$, the Bayes rule is written as:

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

$$p(x|y) = \eta p(y|x)p(x)$$

$\eta = p(y)^{-1}$ normalization factor ($\int p = 1$)

- This theorem is used by both *Markov* and *Kalman-filter* localization algorithms during the measurement update.

# **Markov localization** | the basic algorithms for Markov localization

For all $x_t$ do

$$\overline{bel}(x_t) = \sum_{x_{t-1}} p(x_t|u_t, x_{t-1})bel(x_{t-1}) \qquad \text{(prediction update)}$$

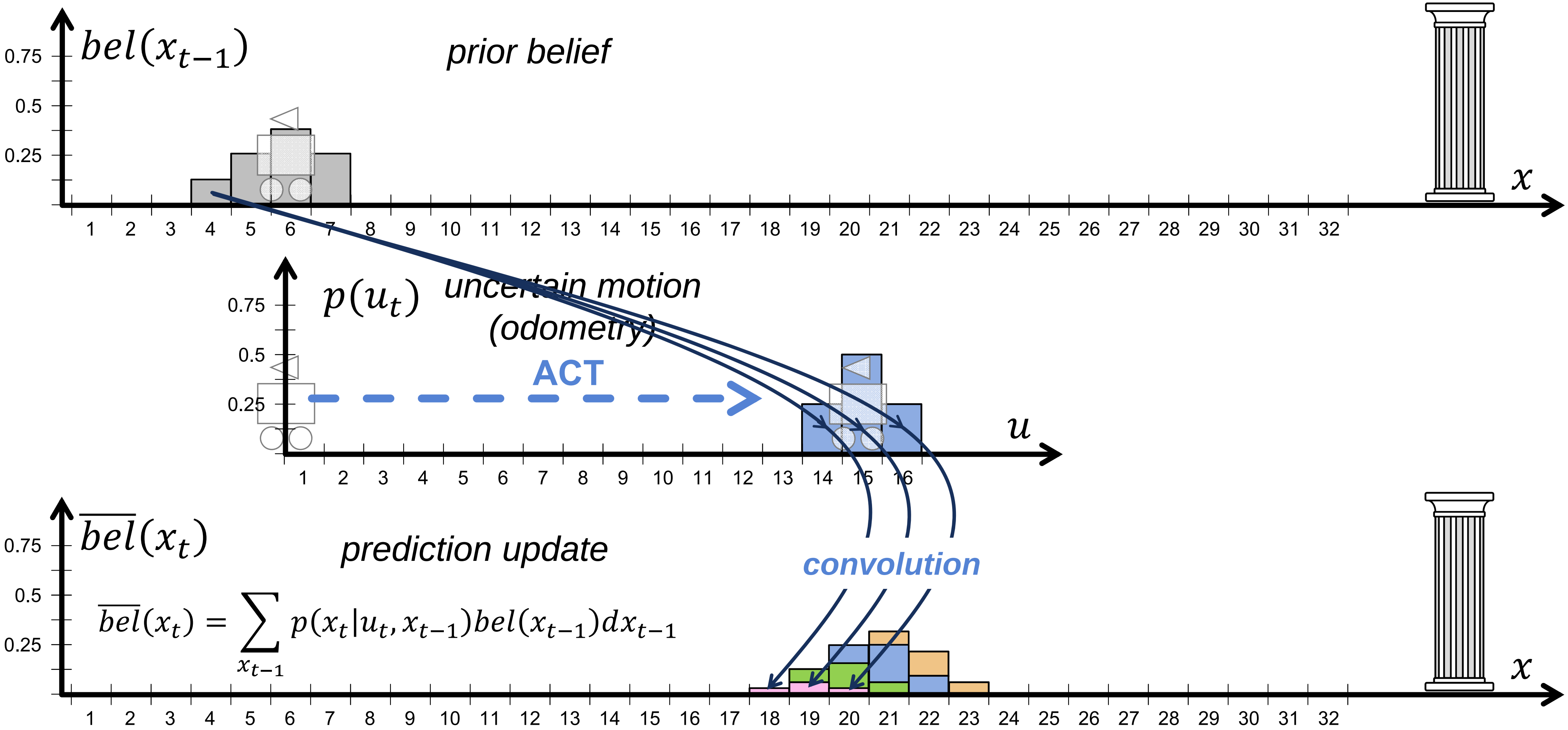
$$bel(x_t) = \eta p(z_t|x_t, M)\overline{bel}(x_t) \qquad \text{(measurement update)}$$

endfor

Return $bel(x_t)$

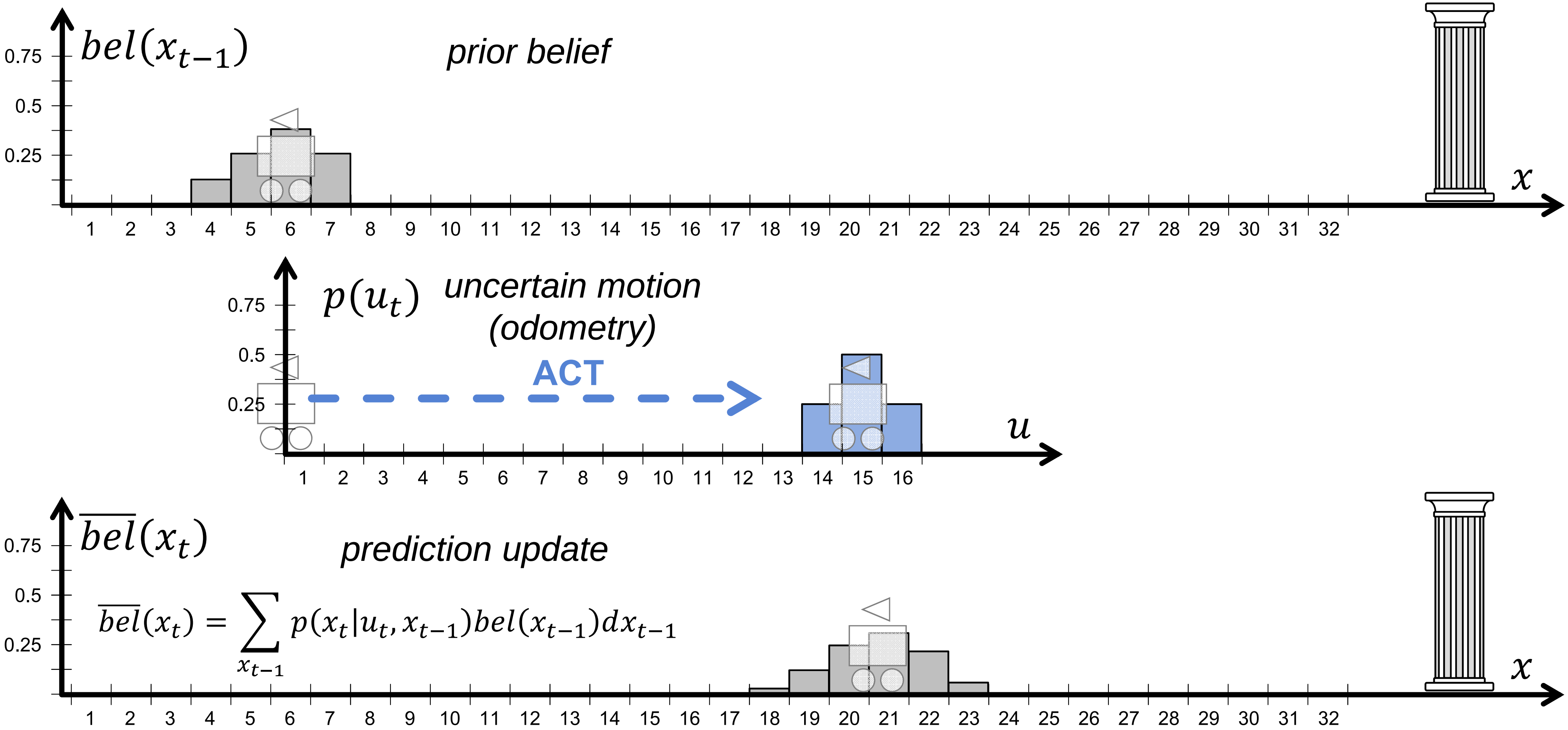
- ***Markov assumption***: Formally, this means that the output is a function $x_t$ only of the robot's previous state $x_t$ and its most recent actions (odometry) $u_t$ and perception $z_t$.

# ACT | using motion model and its uncertainties




$$\overline{bel}(x_t) = \sum_{x_{t-1}} p(x_t|u_t, x_{t-1})bel(x_{t-1})dx_{t-1}$$

# ACT | using motion model and its uncertainties



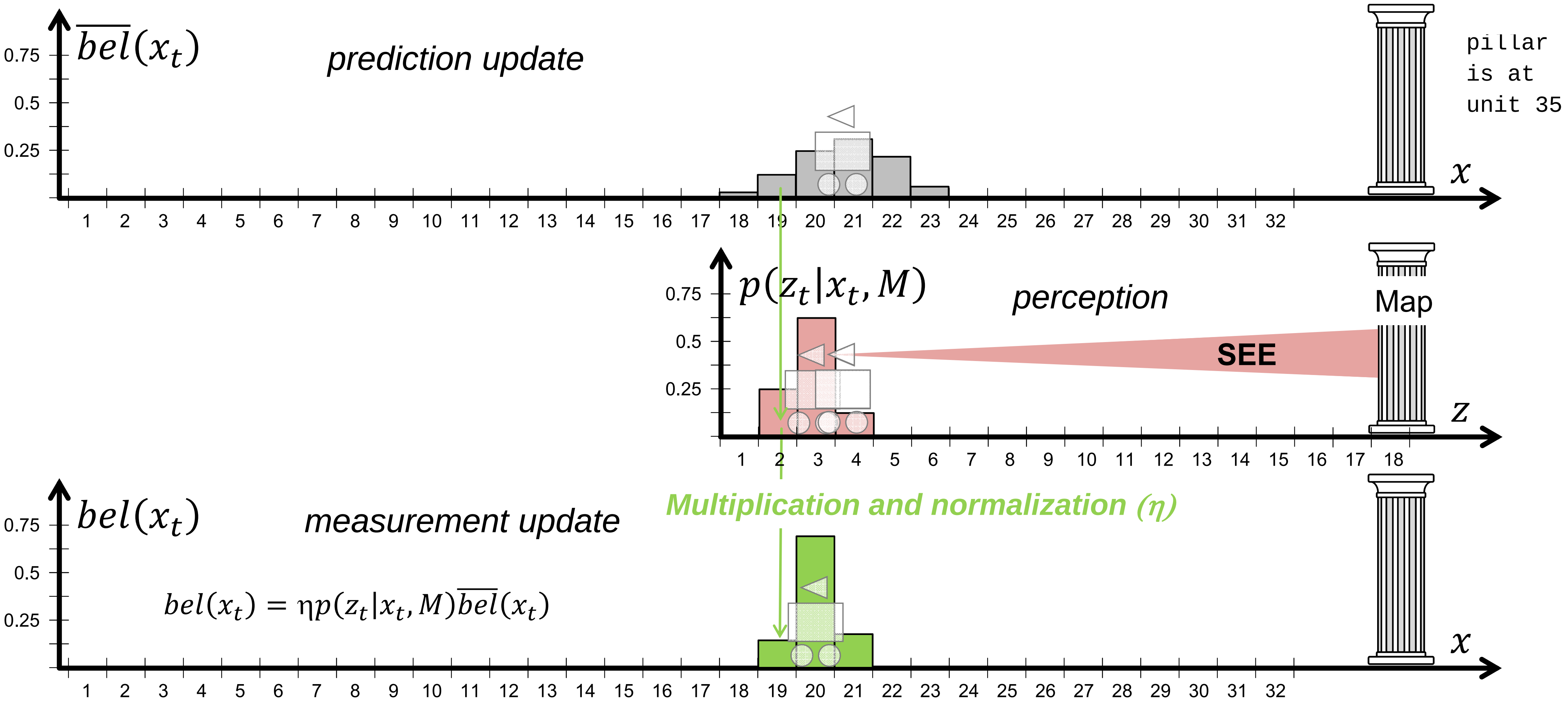

$bel(x_{t-1})$

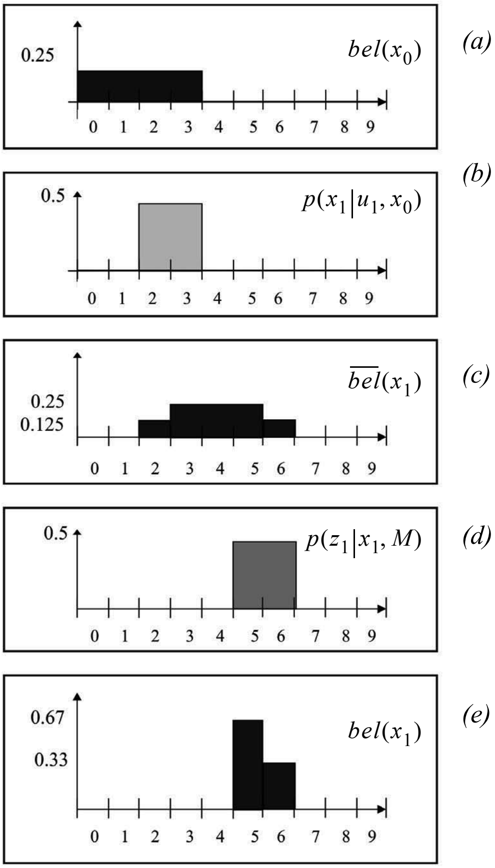
$p(u_t)$

$\overline{bel}(x_t)$

$$\overline{bel}(x_t) = \sum_{x_{t-1}} p(x_t | u_t, x_{t-1}) bel(x_{t-1}) dx_{t-1}$$

# SEE | estimation of position based on perception and map






initial belief of robot's position

ACT: motion probability.
0.5 probability the robot moves 2 units,
0.5 probability the robot moves 3 units

After applying the ACT probability of motion
to the current belief state in (a) we compute
updated belief in (c). Note uncertainty increases

SEE: a range sensor on the robot measures
the robot's distance from the origin. The
sensor has equal probability of measuring the robot
as 5 or 6 units from the origin (0.5 probability each.
This is the sensor error model.

The robot corrects its position by combining its
belief before the observation with the probability
of that observation using Bayes rule. This reduces
the uncertainty.
Note we need to use a scaling factor to make sure
all probabilities add up to 1

**Figure 5.23** Markov localization using a grid-map.

Calculation of the robot's position after the ACT move in (a),(b) above:

$$p(x_1 = 2) = p(x_0 = 0)p(u_1 = 2) = 0.125 \quad (5.44$$

$$p(x_1 = 3) = p(x_0 = 0)p(u_1 = 3) + p(x_0 = 1)p(u_1 = 2) = 0.25 \quad (5.45)$$

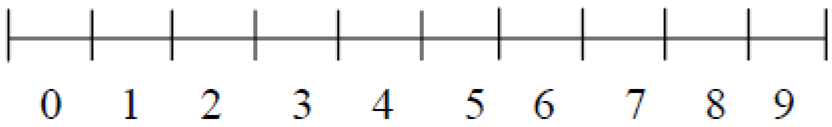
$$p(x_1 = 4) = p(x_0 = 1)p(u_1 = 3) + p(x_0 = 2)p(u_1 = 2) = 0.25 \quad (5.46)$$

$$p(x_1 = 5) = p(x_0 = 2)p(u_1 = 3) + p(x_0 = 3)p(u_1 = 2) = 0.25 \quad (5.47)$$

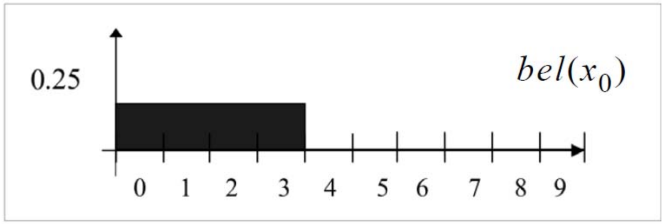
$$p(x_1 = 6) = p(x_0 = 3)p(u_1 = 3) = 0.125 \quad (5.48)$$

# Markov localization

- Let us discretize the configuration space into 10 cells



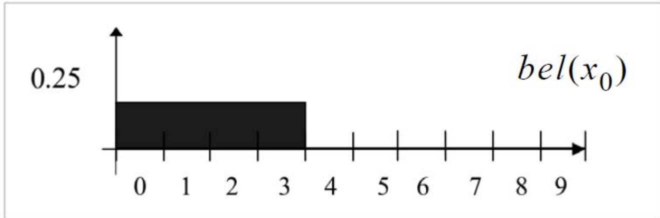

- Suppose that the robot's initial belief is a uniform distribution from 0 to 3. Observe that all the elements were normalized so that their sum is 1.




a)

# Markov localization

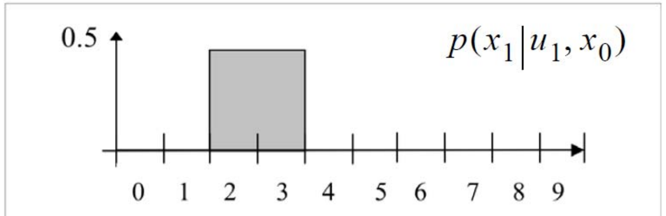
- Initial belief distribution




a)

- **Action phase:**
  Let us assume that the robot moves forward with the following statistical model

b)

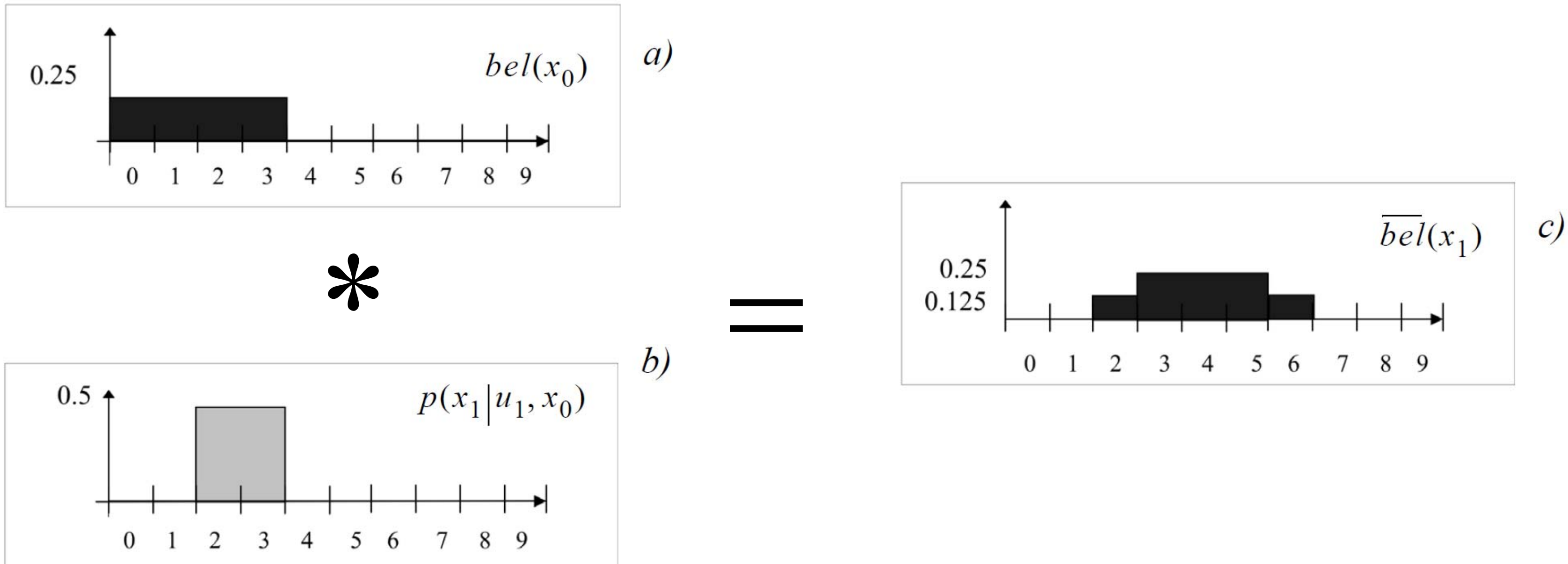



- This means that we have 50% probability that the robot moved 2 or 3 cells forward.
- Considering what the probability was before moving, what will the probability be after the motion?

# Markov localization
## *Action update*

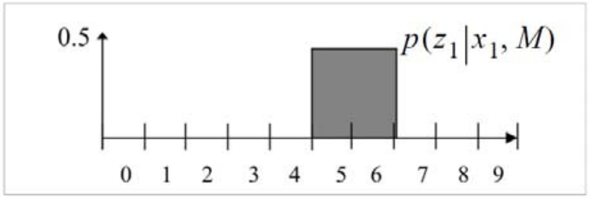
- The solution is given by the *convolution (cross correlation)* of the two distributions

$$\overline{bel}(x_t) = (x_1|u_1, x_0) * bel(x_0) = \sum_0^3 p(x_1|u_1, x_0) bel(x_0)$$



# Markov localization
## *Perception update*

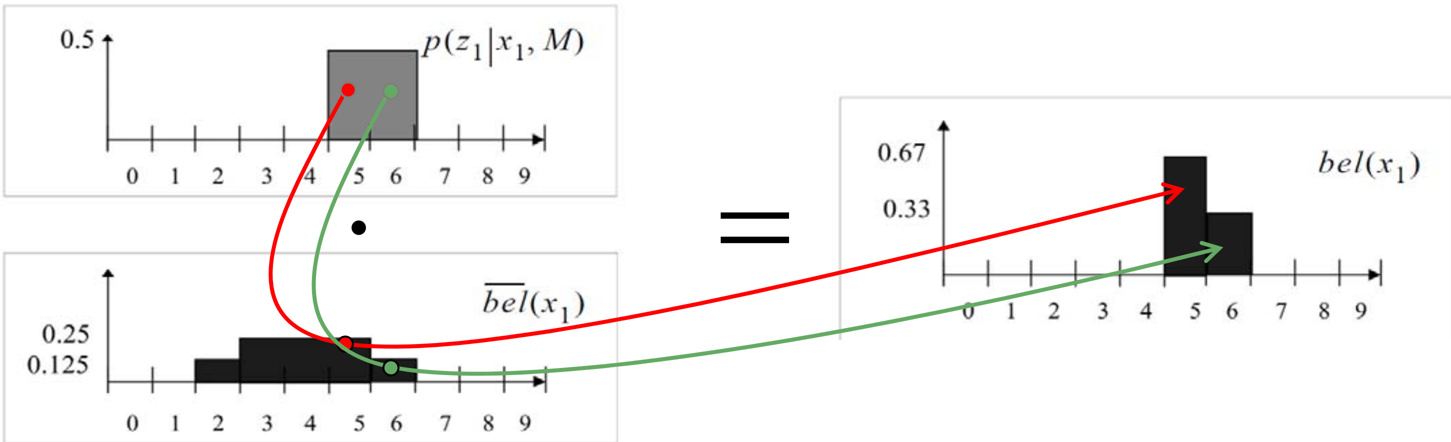
- Let us now assume that the robot uses its onboard range finder and measures the distance from the origin. Assume that the statistical error model of the sensors is:




This plot tells us that the distance of the robot from the origin can be equally 5 or 6 units.

- What will the final robot belief be after this measurement?
  The answer is again given by the Bayes rule:

$$bel(x_t) = \eta p(z_t|x_t, M)\overline{bel}(x_t)$$



# Markov Localization Example, p. 313 Siegwart

1 INITIAL BELIEF: Bel(X) at time t
GRID CELL

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Bel(X) at time t | 0.25 | 0.25 | 0.25 | 0.25 | 0 | 0 | 0 | 0 | 0 | 0 |
| GRID CELL | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

2 Now move the robot with probabilities below:

3 MOTION PROBABILITY: U(t) -robot moves 2 or 3 units
GRID CELL

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| U(t) | 0 | 0 | 0.5 | 0.5 | 0 | 0 | 0 | 0 | 0 | 0 |
| GRID CELL | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

4 Now CONVOLVE Bel(X) with U(t)

5 UPDATED BELIEF: <u>Bel(X)</u>
GRID CELL

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Bel(X) | 0 | 0 | 0.125 | 0.25 | 0.25 | 0.25 | 0.125 | 0 | 0 | 0 |
| GRID CELL | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

6 Now use sensor to update your <u>Bel(X)</u>

7 SENSOR Probabilities: Z(t) - origin is 5 or 6 units away
GRID CELL

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Z(t) | 0 | 0 | 0 | 0 | 0 | 0.5 | 0.5 | 0 | 0 | 0 |
| GRID CELL | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

8 Apply sensor measurement to current <u>Bel(X)</u>

9 UNNORMALIZED SENSOR UPDATE
GRID CELL

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Unnormalized | 0 | 0 | 0 | 0 | 0 | 0.125 | 0.0625 | 0 | 0 | 0 |
| GRID CELL | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

10 NORMALIZATION = .0625 + 0.125= 0.1875

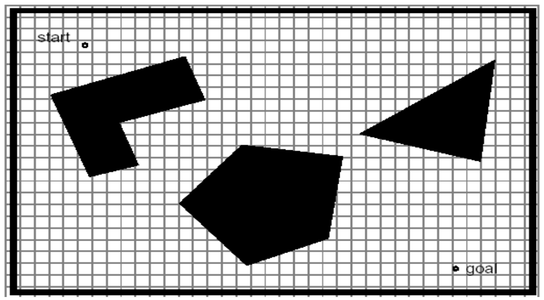
0.125 / 0.1875 = .667 , 0.0625/ 0.1875 = .33

11 NORMALIZED SENSOR UPDATE: Bel(X) at t+1
GRID CELL

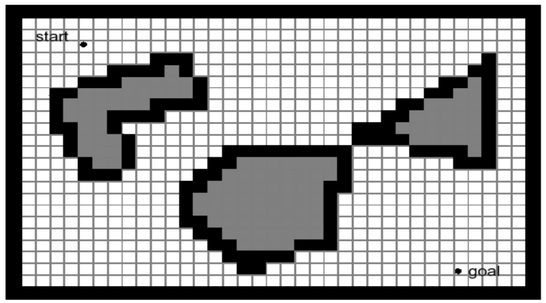
| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Bel(X) at t+1 | 0 | 0 | 0 | 0 | 0 | 0.6667 | 0.3333 | 0 | 0 | 0 |
| GRID CELL | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

# **Markov localization** | extension to 2D

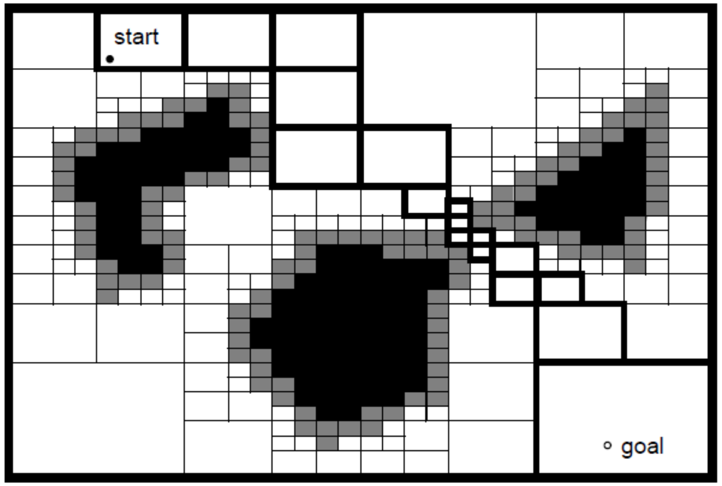
- The real world for mobile robot is at least 2D (moving in the plane)
  - → discretized pose state space (grid) consists of $x, y, \theta$
  - → Markov Localization scales badly with the size of the environment
- Space: 10 m x 10 m with a grid size of 0.1 m and an angular resolution of 1°
  - → $100 \cdot 100 \cdot 360 = 3.6\ 10^6$ grid points (states)
  - → prediction step requires in worst case $(3.6\ 10^6)^2$ multiplications and summations




- Fine fixed decomposition grids result in a huge state space
  - Very important processing power needed
  - Large memory requirement



# **Markov localization** | reducing computational complexity

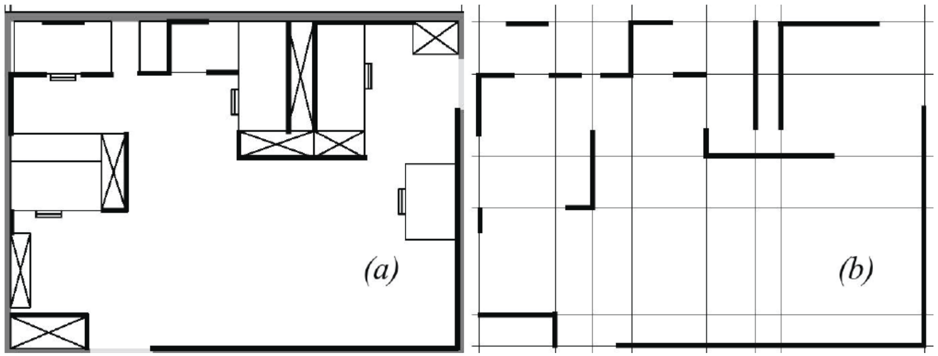


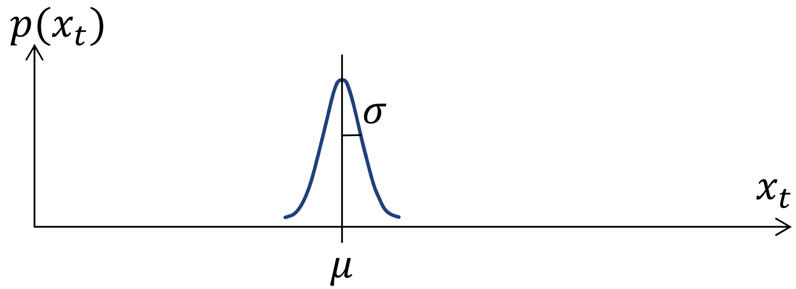

- Adaptive cell decomposition
- Motion model (Odometry) limited to a small number of grid points
- Randomized sampling
  - Approximation of belief state by a representative subset of possible locations
  - weighting the sampling process with the probability values
  - Injection of some randomized (not weighted) samples

  - randomized sampling methods are also known as particle filter algorithms, condensation algorithms, and Monte Carlo algorithms.

# **Kalman Filter Localization** | Basics and assumption

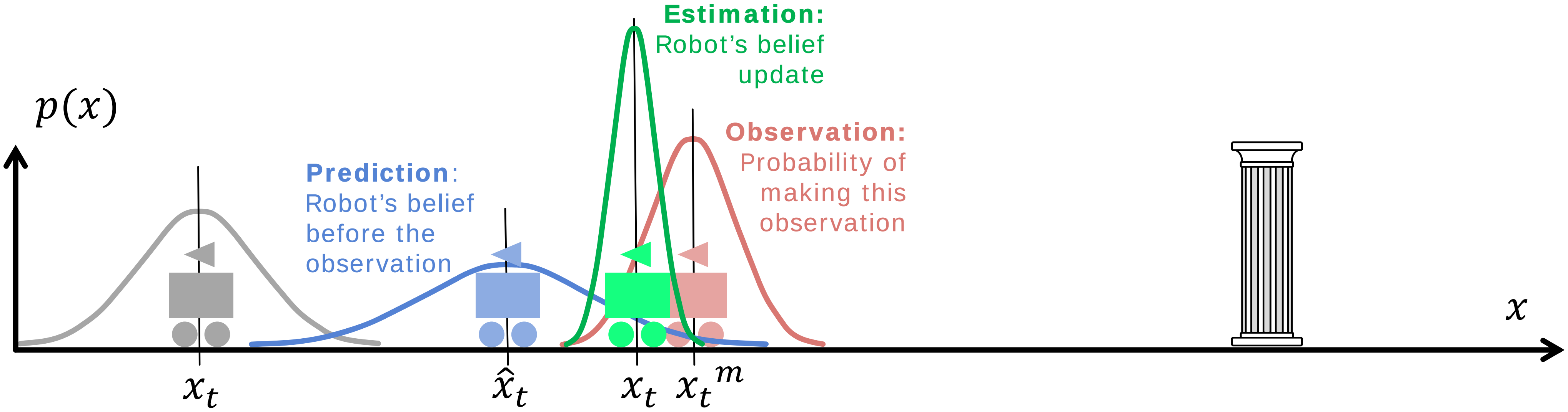
- Continuous pose representation $x_t$
- Kalman Filter Assumptions:
  - Error approximation with normal distribution: $x = N(\mu, \sigma^2)$ (Gaussian model)
  - Output $y_t$ distribution is a linear (or linearized) function of the input distribution: $y = Ax_1 + Bx_2$
- Kalman filter localization tracks the robot's belief state $p(x_t)$ typically as a single hypothesis with normal distribution.
- Kalman localization thus addresses the *position tracking problem*, but ***not*** the *global localization* or the *kidnapped robot problem*.



# Kalman Filter Localization | in summery

1. **Prediction (ACT)** based on previous estimate and odometry
2. **Observation (SEE)** with on-board sensors
3. **Measurement prediction** based on prediction and map
4. **Matching** of observation and map
5. **Estimation** → position update (posteriori position)



# Two general approaches: Markov and Kalman Filter Localization

- Markov localization
  - *Maintains multiple estimates of robot position*
  - *Localization can start from any unknown position*
  - *Can recover from ambiguous situations*
  - *However, to update the probability of all positions within the state space requires a discrete representation of the space (grid); if a fine grid is used (or many estimates are maintained), the computational and memory requirements can be large.*

- Kalman filter localization
  - *Single estimate of robot position*
  - *Requires known starting position of robot*
  - *Tracks the robot and can be very precise and efficient*
  - *However, if the uncertainty of the robot becomes too large (e.g. due collision with an object) the Kalman filter will fail and the robot becomes "lost".*