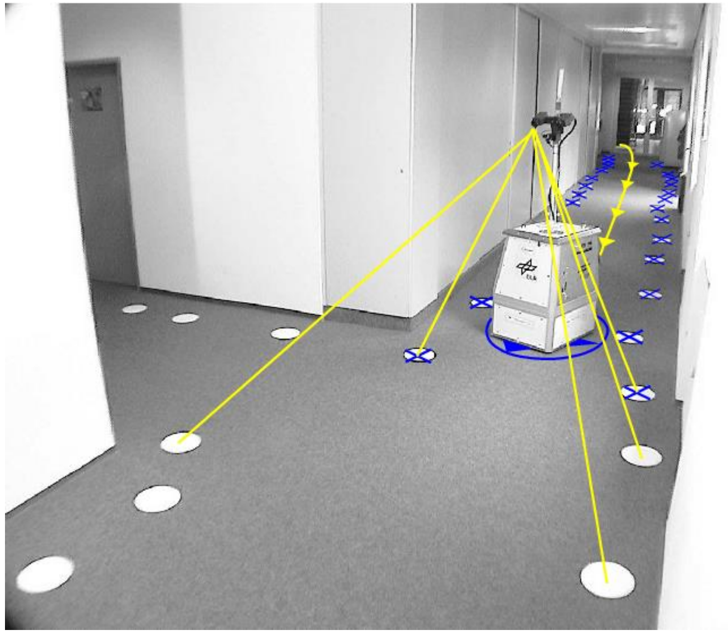
Robust Mechatronics

Localization and Mapping for Autonomous Mobile Systems

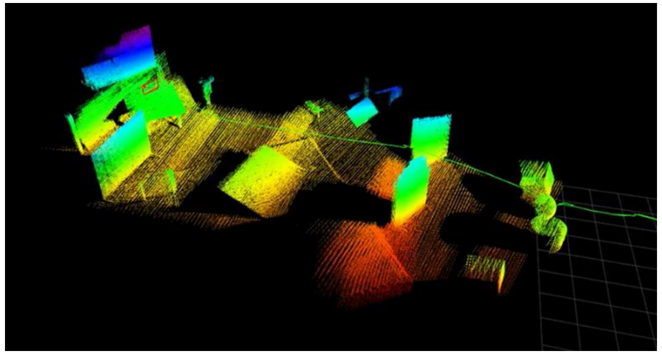


Dr Loukas Bampis, Assistant Professor
Mechatronics & Systems Automation Lab

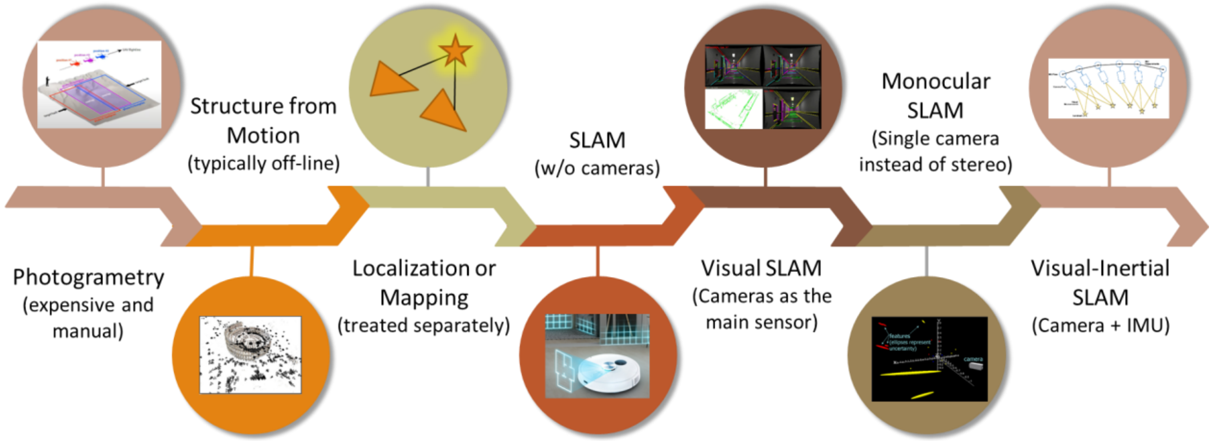
Localization and Mapping for Autonomous Mobile Systems

What we previously discussed:

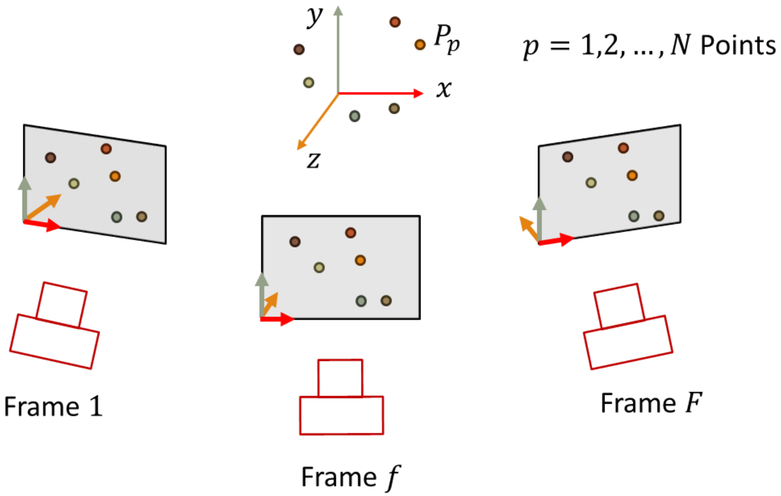
What is SLAM?



Evolution of SLAM



Structure from Motion



SLAM:: The online and real-time version

Difference between Structure from Motion and SLAM
Why not to use such approaches in robotics applications?

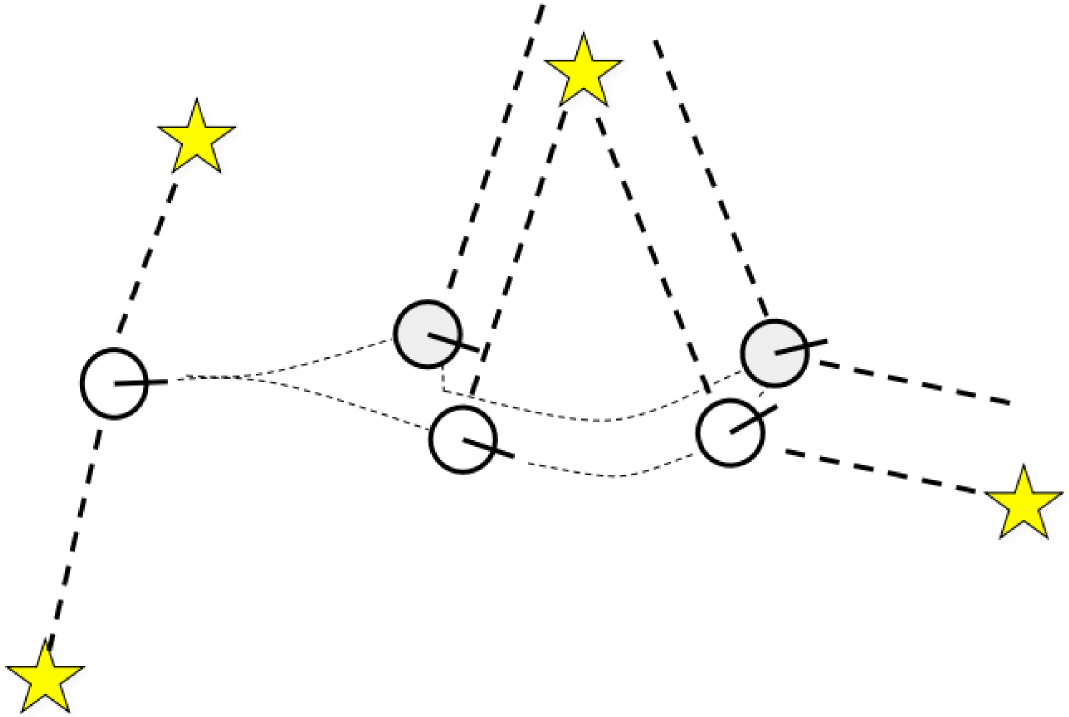
- We cannot wait for all the frames to be captured
- Even then, we cannot expect to process all these frames in real-time

Localization and Mapping for Autonomous Mobile Systems

SLAM:: If we were to split the two functionalities

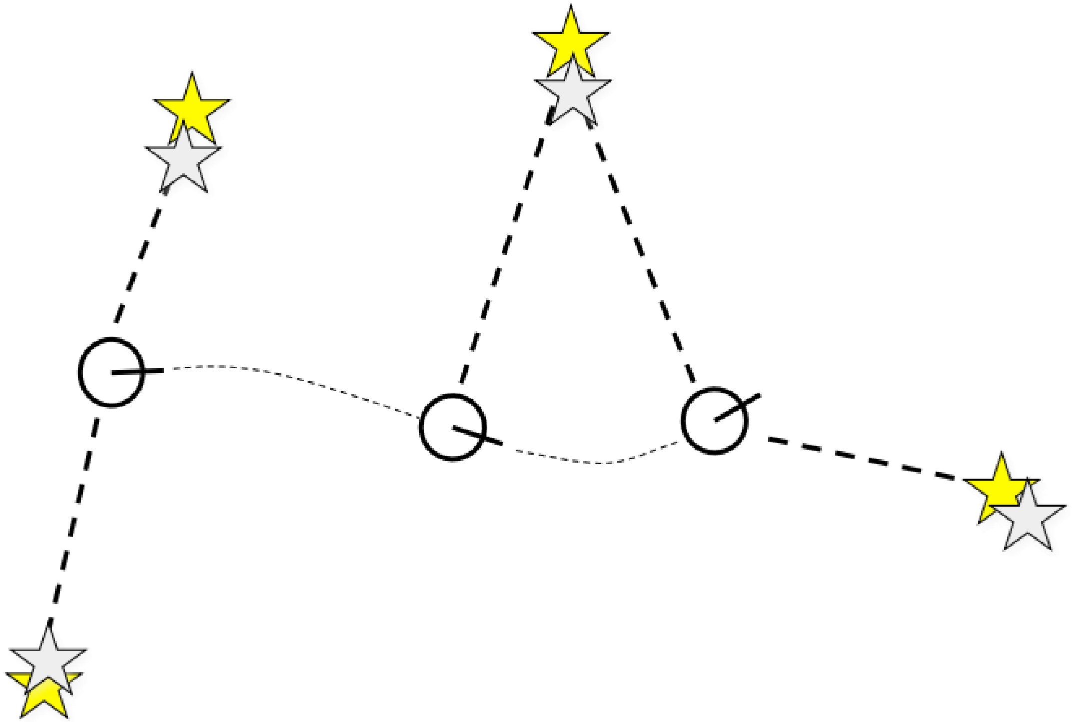
Localization

Estimate the robot's pose given landmarks



Mapping

Estimate the landmarks given the robot's poses



Modeling SLAM

Given

- The robot's controls
  $u_{1:T} = \{u_1, u_2, u_3, \dots, u_T\}$
- Observations
  $z_{1:T} = \{z_1, z_2, z_3, \dots, z_T\}$

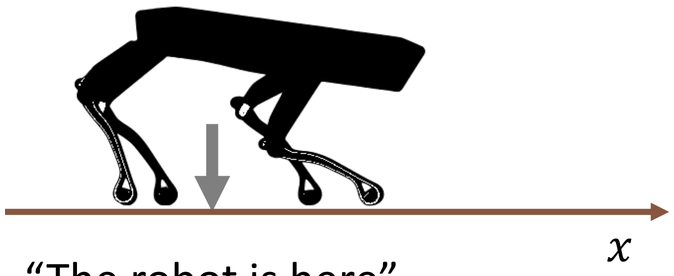
Wanted

- Map of the environment
  m
- Path of the robot
  $x_{0:T} = \{x_0, x_1, x_2, \dots, x_T\}$

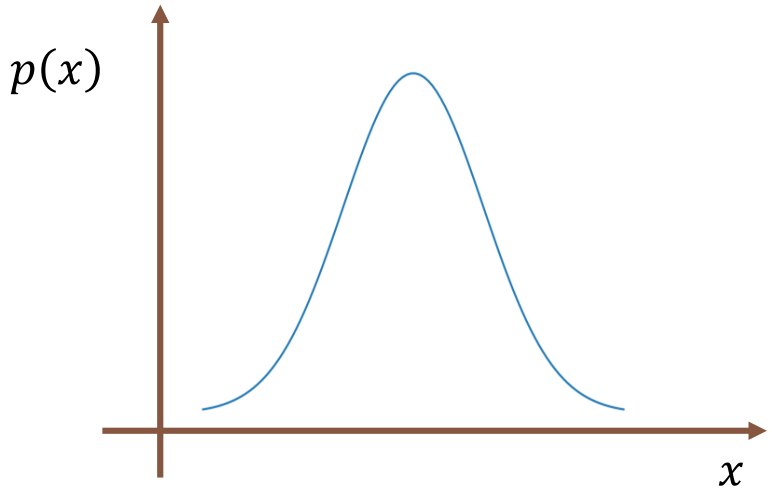
Modeling SLAM

Modeling this problem probabilistically

- There is uncertainty in the measured robot's motion and observations
- We can use probability theory to represent this uncertainty



"The robot is here"



"The robot is somewhere here with this level of certainty"

Modeling SLAM

Modeling this problem probabilistically

Given

Observations

Distribution

$$p(x_{0:T}, m \mid z_{1:T}, u_{1:T})$$

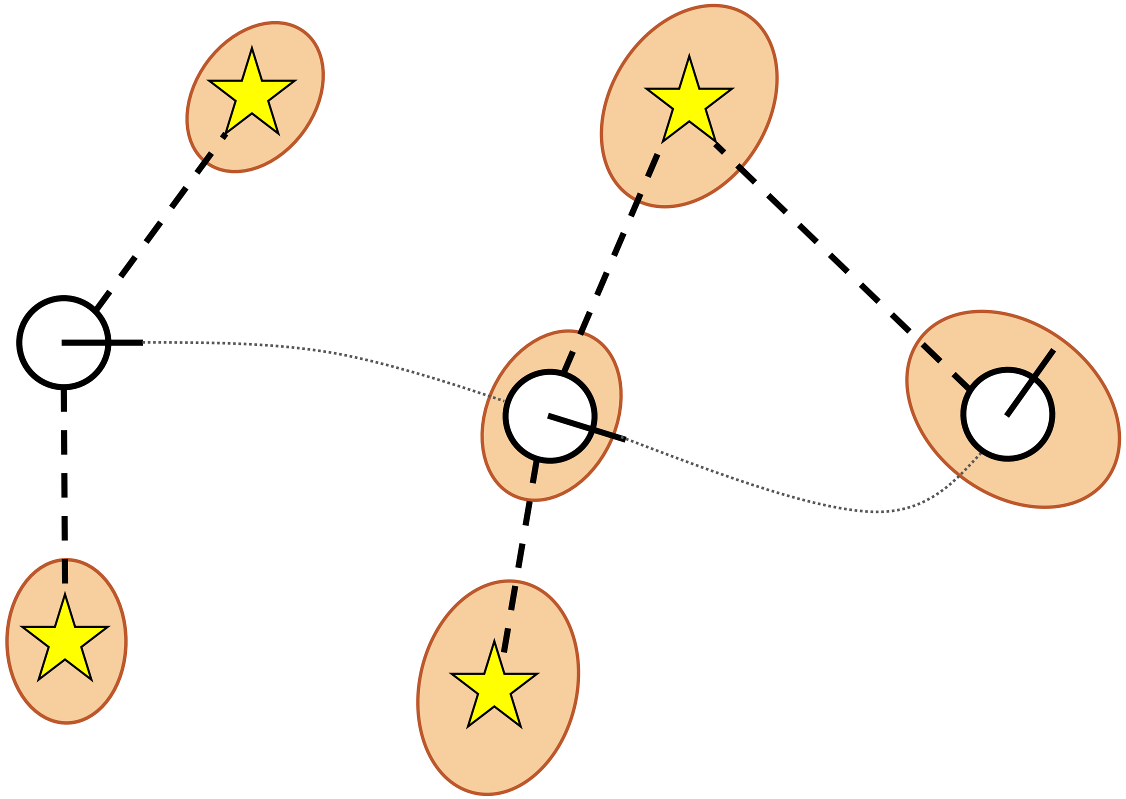
Path

Map

Control commands

Modeling SLAM

What do these probabilities actually mean?



Modeling SLAM: State Estimation

Estimate the state x of a system given observations z and controls u.

$$p(x \mid z, u)$$

- State is defined by us and can contain any combination of the robot's pose and the map
- As we propagate in time, this distribution will become better and better
- Finally, the expected value (mean) will give us our best estimate for the state

Modeling SLAM: State Estimation

Estimate the state x of a system given observations z and controls u.

$$p(x_t | z_{1:t}, u_{1:t})$$

- State is defined by us and can contain any combination of the robot's pose and the map
- As we propagate in time, this distribution will become better and better
- Finally, the expected value (mean) will give us our best estimate for the state

SLAM: Recursive Bayes Filter

The basis of on-line SLAM

- We need to build a recursive formula:

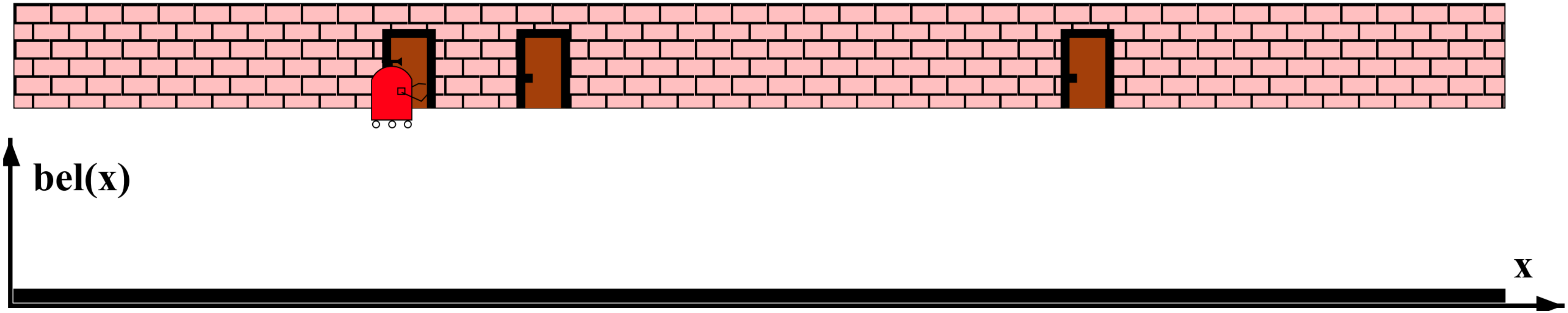
$$f(x_i) = g(f(x_{i-k}))$$

Examples

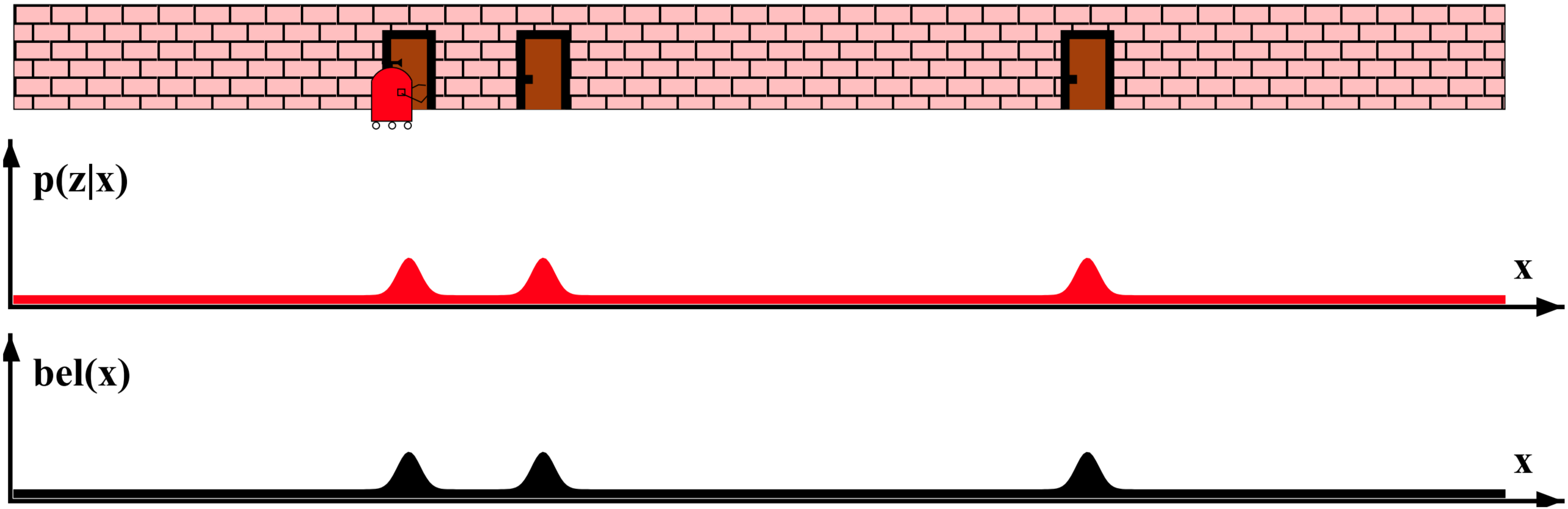
- $f(x) = f(x-2) + 1$
- $x_{i+1} = 3x_i + 2$

SLAM: Recursive Bayes Filter

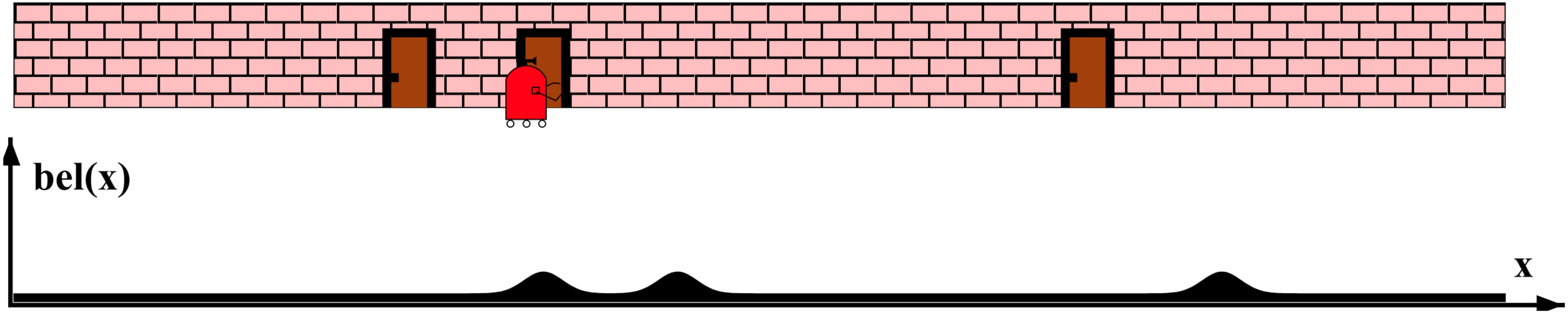
The door sensing robot example



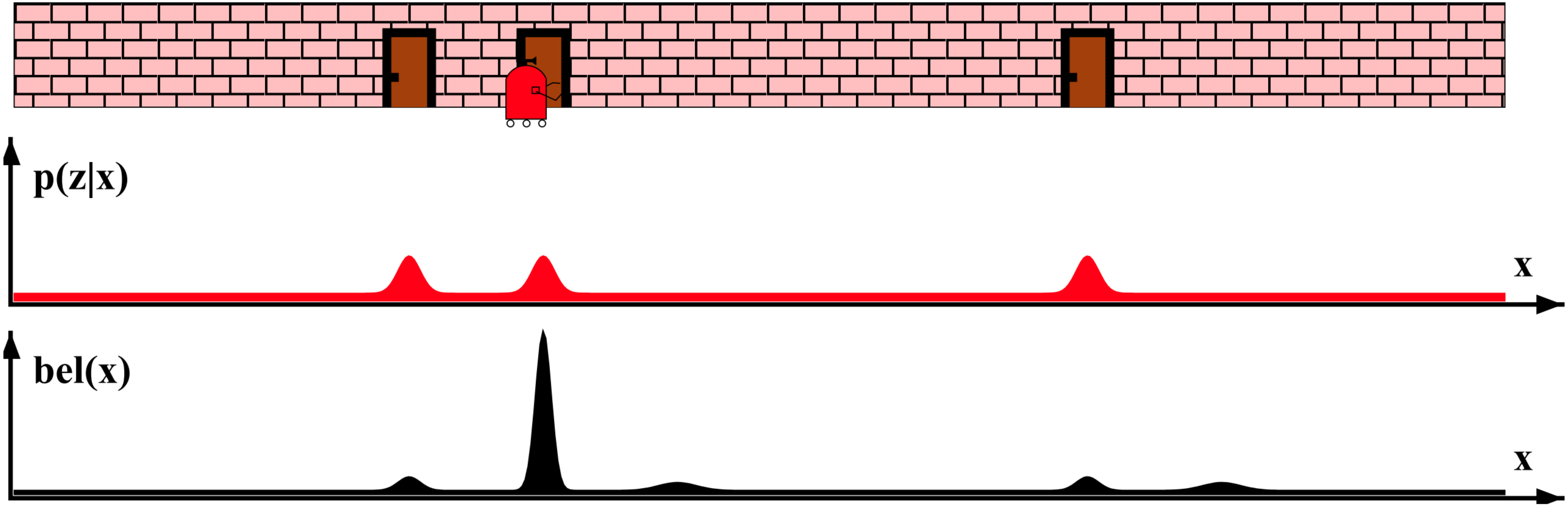
SLAM: Recursive Bayes Filter



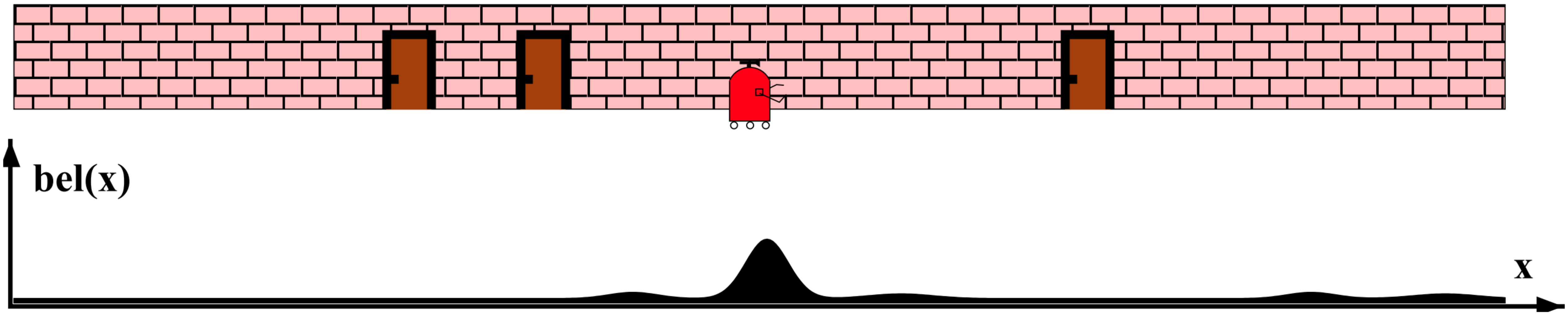
SLAM: Recursive Bayes Filter



SLAM: Recursive Bayes Filter



SLAM: Recursive Bayes Filter



SLAM: Recursive Bayes Filter

$bel(x_t) = p(x_t \mid z_{1:t}, u_{1:t})$ // Belief definition

SLAM: Recursive Bayes Filter

$$bel(x_t) = p(x_t \,|z_{1:t}, u_{1:t})$$
$$= \underline{\eta\, p(z_t|x_t, z_{1:t-1}, u_{1:t})\, p(x_t \mid z_{1:t-1}, u_{1:t})}$$

// Bayes' rule

η: normalization constant

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

SLAM: Recursive Bayes Filter

$$
\begin{aligned}
bel(x_t) &= p(x_t \mid z_{1:t}, u_{1:t}) \\
&= \eta\, p(z_t \mid x_t, z_{1:t-1}, u_{1:t})\, p(x_t \mid z_{1:t-1}, u_{1:t}) \\
&= \eta\, \underline{p(z_t \mid x_t)}\, p(x_t \mid z_{1:t-1}, u_{1:t}) \quad \text{// Markov assumption}
\end{aligned}
$$

SLAM: Recursive Bayes Filter

$$
\begin{aligned}
bel(x_t) &= p(x_t \mid z_{1:t}, u_{1:t}) \\
&= \eta\, p(z_t \mid x_t, z_{1:t-1}, u_{1:t})\, p(x_t \mid z_{1:t-1}, u_{1:t}) \\
&= \eta\, p(z_t \mid x_t)\, p(x_t \mid z_{1:t-1}, u_{1:t}) \\
&= \eta\, p(z_t \mid x_t) \int_{x_{t-1}} p(x_t \mid x_{t-1}, z_{1:t-1}, u_{1:t})\, p(x_{t-1} \mid z_{1:t-1}, u_{1:t})\, dx_{t-1}
\end{aligned}
$$

// Law of total probability

SLAM: Recursive Bayes Filter

$$
\begin{aligned}
bel(x_t) &= p(x_t \mid z_{1:t}, u_{1:t}) \\
&= \eta\, p(z_t \mid x_t, z_{1:t-1}, u_{1:t})\, p(x_t \mid z_{1:t-1}, u_{1:t}) \\
&= \eta\, p(z_t \mid x_t)\, p(x_t \mid z_{1:t-1}, u_{1:t}) \\
&= \eta\, p(z_t \mid x_t) \int_{x_{t-1}} p(x_t \mid x_{t-1}, z_{1:t-1}, u_{1:t})\, p(x_{t-1} \mid z_{1:t-1}, u_{1:t})\, dx_{t-1}
\end{aligned}
$$

// Law of total probability

Law of total probability:

$$P(A) = \int_B P(A|B)\, P(B)\, dB$$

For the discrete case, it may be more intuitive:

$$P(A) = \sum_B P(A|B)\, P(B)$$

SLAM: Recursive Bayes Filter

$$
\begin{aligned}
bel(x_t) &= p(x_t \mid z_{1:t}, u_{1:t}) \\
&= \eta \, p(z_t \mid x_t, z_{1:t-1}, u_{1:t}) \, p(x_t \mid z_{1:t-1}, u_{1:t}) \\
&= \eta \, p(z_t \mid x_t) \, p(x_t \mid z_{1:t-1}, u_{1:t}) \\
&= \eta \, p(z_t \mid x_t) \int_{x_{t-1}} p(x_t \mid x_{t-1}, z_{1:t-1}, u_{1:t}) \, p(x_{t-1} \mid z_{1:t-1}, u_{1:t}) \, dx_{t-1} \\
&= \eta \, p(z_t \mid x_t) \int_{x_{t-1}} \underline{p(x_t \mid x_{t-1}, u_t)} \, p(x_{t-1} \mid z_{1:t-1}, u_{1:t}) \, dx_{t-1} \quad \text{// Markov assumption}
\end{aligned}
$$

SLAM: Recursive Bayes Filter

$$
\begin{aligned}
bel(x_t) &= p(x_t \mid z_{1:t}, u_{1:t}) \\
&= \eta\, p(z_t \mid x_t, z_{1:t-1}, u_{1:t})\, p(x_t \mid z_{1:t-1}, u_{1:t}) \\
&= \eta\, p(z_t \mid x_t)\, p(x_t \mid z_{1:t-1}, u_{1:t}) \\
&= \eta\, p(z_t \mid x_t) \int_{x_{t-1}} p(x_t \mid x_{t-1}, z_{1:t-1}, u_{1:t})\, p(x_{t-1} \mid z_{1:t-1}, u_{1:t})\, dx_{t-1} \\
&= \eta\, p(z_t \mid x_t) \int_{x_{t-1}} p(x_t \mid x_{t-1}, u_t)\, p(x_{t-1} \mid z_{1:t-1}, u_{1:t})\, dx_{t-1} \\
&= \eta\, p(z_t \mid x_t) \int_{x_{t-1}} p(x_t \mid x_{t-1}, u_t)\, p(x_{t-1} \mid z_{1:t-1}, \underline{u_{1:t-1}})\, dx_{t-1}
\end{aligned}
$$

// Independence assumption

SLAM: Recursive Bayes Filter

$$bel(x_t) = p(x_t \mid z_{1:t}, u_{1:t})$$
$$= \eta \, p(z_t \mid x_t, z_{1:t-1}, u_{1:t}) \, p(x_t \mid z_{1:t-1}, u_{1:t})$$
$$= \eta \, p(z_t \mid x_t) \, p(x_t \mid z_{1:t-1}, u_{1:t})$$
$$= \eta \, p(z_t \mid x_t) \int_{x_{t-1}} p(x_t \mid x_{t-1}, z_{1:t-1}, u_{1:t}) \, p(x_{t-1} \mid z_{1:t-1}, u_{1:t}) \, dx_{t-1}$$
$$= \eta \, p(z_t \mid x_t) \int_{x_{t-1}} p(x_t \mid x_{t-1}, u_t) \, p(x_{t-1} \mid z_{1:t-1}, u_{1:t}) \, dx_{t-1}$$
$$= \eta \, p(z_t \mid x_t) \int_{x_{t-1}} p(x_t \mid x_{t-1}, u_t) \, p(x_{t-1} \mid z_{1:t-1}, u_{1:t-1}) \, dx_{t-1}$$
$$= \eta \, p(z_t \mid x_t) \int_{x_{t-1}} p(x_t \mid x_{t-1}, u_t) \, \underline{bel(x_{t-1})} \, dx_{t-1} \quad \text{// Recursive term}$$

SLAM: Recursive Bayes Filter

$$bel(x_t) = p(x_t \mid z_{1:t}, u_{1:t})$$

$$= \eta\, p(z_t \mid x_t, z_{1:t-1}, u_{1:t})\, p(x_t \mid z_{1:t-1}, u_{1:t})$$ // Bayes' rule

$$= \eta\, p(z_t \mid x_t)\, p(x_t \mid z_{1:t-1}, u_{1:t})$$ // Markov assumption

$$= \eta\, p(z_t \mid x_t) \int_{x_{t-1}} p(x_t \mid x_{t-1}, z_{1:t-1}, u_{1:t})\, p(x_{t-1} \mid z_{1:t-1}, u_{1:t})\, dx_{t-1}$$ // Law of total probability

$$= \eta\, p(z_t \mid x_t) \int_{x_{t-1}} p(x_t \mid x_{t-1}, u_t)\, p(x_{t-1} \mid z_{1:t-1}, u_{1:t})\, dx_{t-1}$$ // Markov assumption

$$= \eta\, p(z_t \mid x_t) \int_{x_{t-1}} p(x_t \mid x_{t-1}, u_t)\, p(x_{t-1} \mid z_{1:t-1}, u_{1:t-1})\, dx_{t-1}$$ // Independence assumption

$$= \eta\, p(z_t \mid x_t) \int_{x_{t-1}} p(x_t \mid x_{t-1}, u_t)\, bel(x_{t-1})\, dx_{t-1}$$ // Recursive term

SLAM: Recursive Bayes Filter

We typically split the process into a **Prediction** and a **Correction Step**

- Prediction Step

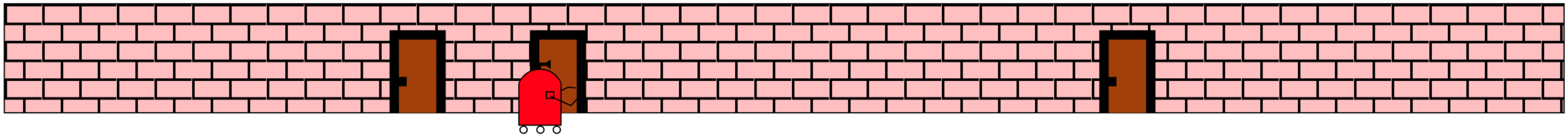
$$\overline{bel}(x_t) = \int_{x_{t-1}} \underbrace{p(x_t \mid x_{t-1}, u_t)}_{\text{Motion model}} \, bel(x_{t-1}) \, dx_{t-1}$$

- Correction Step

$$bel(x_t) = \eta \underbrace{p(z_t \mid x_t)}_{\text{Observation model}} \, \overline{bel}(x_t)$$

(Also: measurement or sensor model)

Localization and Mapping for Autonomous Mobile Systems



Motion Model

$p(x_t \mid x_{t-1}, u_t)$

Observation Model

$p(z_t \mid x_t)$

New belief generation

$bel(x_t)$

SLAM: Recursive Bayes Filter

Motion model: specifies a posterior probability that the control command u_t carries the robot from x_{t-1} to x_t

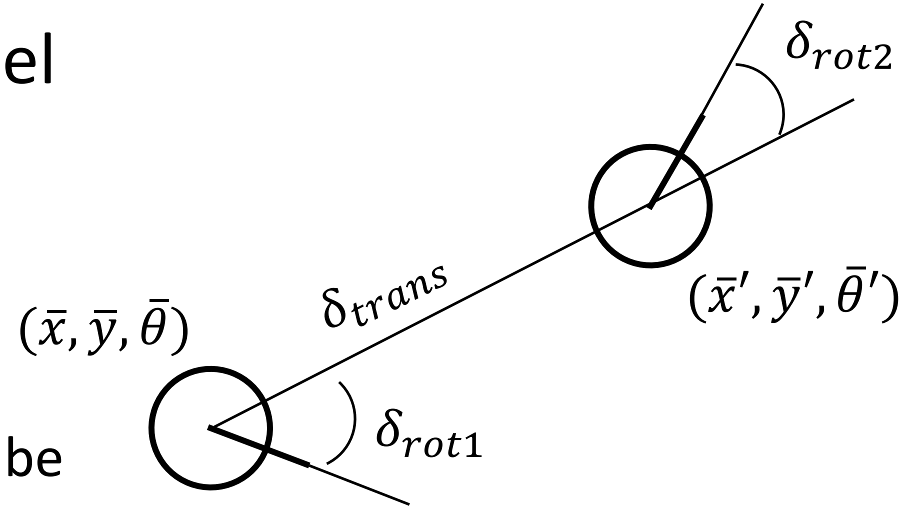
$p(x_t \mid x_{t-1}, u_t)$

Instead of control commands, u_t can come from a proprioceptive sensor

- Odometry-based
  - E.g., wheel encoders
- Velocity-based
  - E.g., IMU

SLAM: Recursive Bayes Filter

Motion model: Standard odometry model



- We define poses in the 2D world as (x, y, θ)
- The movement from $(\bar{x}, \bar{y}, \bar{\theta})$ to $(\bar{x}', \bar{y}', \bar{\theta}')$ can be expressed as: $u = (\delta_{rot1}, \delta_{trans}, \delta_{rot2})$

$$\delta_{trans} = \sqrt{(\bar{x}' - \bar{x})^2 + (\bar{y}' - \bar{y})^2}$$

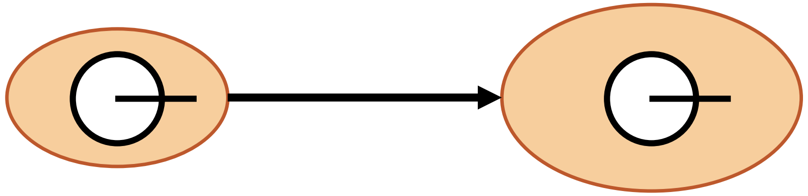
$$\delta_{rot1} = \text{atan2}(\bar{y}' - \bar{y}, \bar{x}' - \bar{x}) - \bar{\theta} \qquad \delta_{rot2} = \bar{\theta}' - \bar{\theta} - \delta_{rot1}$$

SLAM: Recursive Bayes Filter

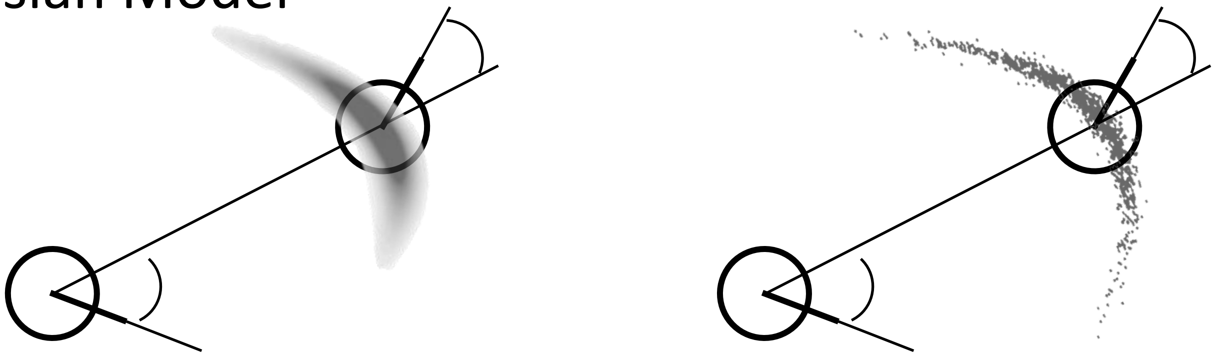
Motion model: Standard odometry model

Probability distribution for $p(x_t \mid x_{t-1}, u_t)$

- Gaussian Model



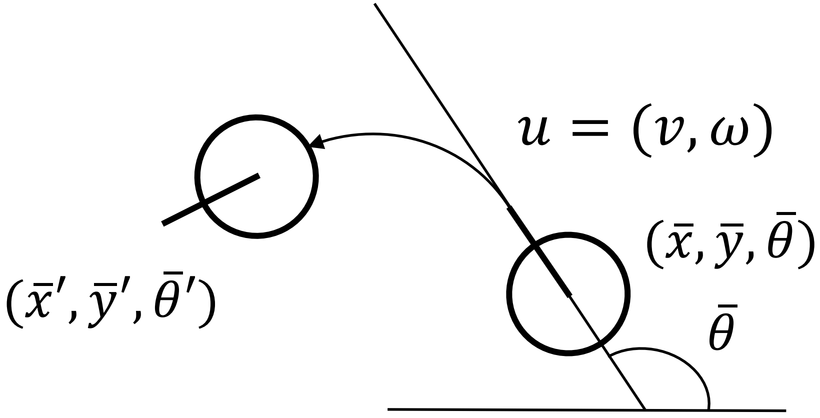
- Non-Gaussian Model



SLAM: Recursive Bayes Filter

Motion model: Standard velocity model

- We define poses in the 2D world as (x, y, θ)



- The movement from $(\bar{x}, \bar{y}, \bar{\theta})$ to $(\bar{x}', \bar{y}', \bar{\theta}')$ can be expressed as: $u = (v, \omega)$

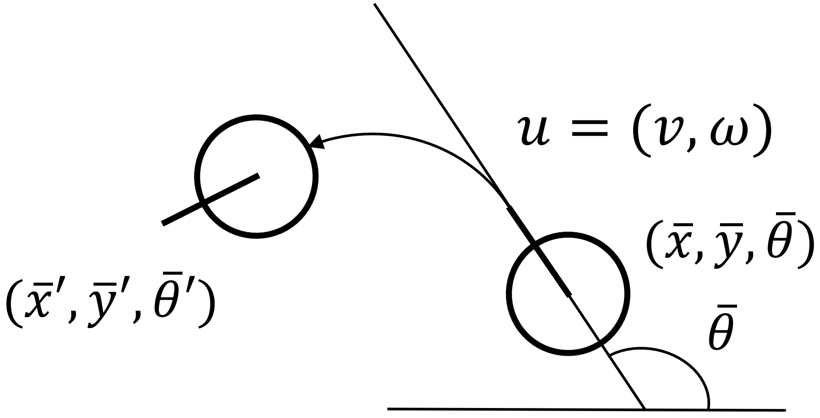
$$\begin{bmatrix} \bar{x}' \\ \bar{y}' \\ \bar{\theta}' \end{bmatrix} = \begin{bmatrix} \bar{x} \\ \bar{y} \\ \bar{\theta} \end{bmatrix} + \begin{bmatrix} -\frac{v}{\omega}\sin\bar{\theta} + \frac{v}{\omega}\sin(\bar{\theta} + \omega\Delta t) \\ \frac{v}{\omega}\cos\bar{\theta} - \frac{v}{\omega}\cos(\bar{\theta} + \omega\Delta t) \\ \omega\Delta t \end{bmatrix}$$

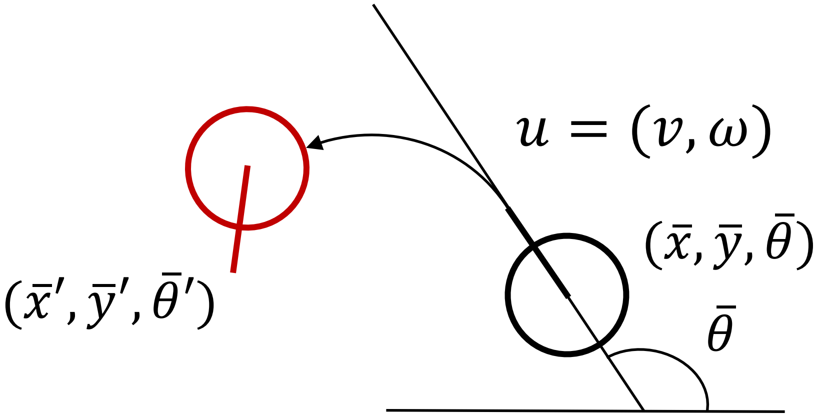
We have an issue here!

SLAM: Recursive Bayes Filter

Motion model: Standard velocity model

- The previous model forces the robot to execute a curve
- What if we need the robot to face on a different direction

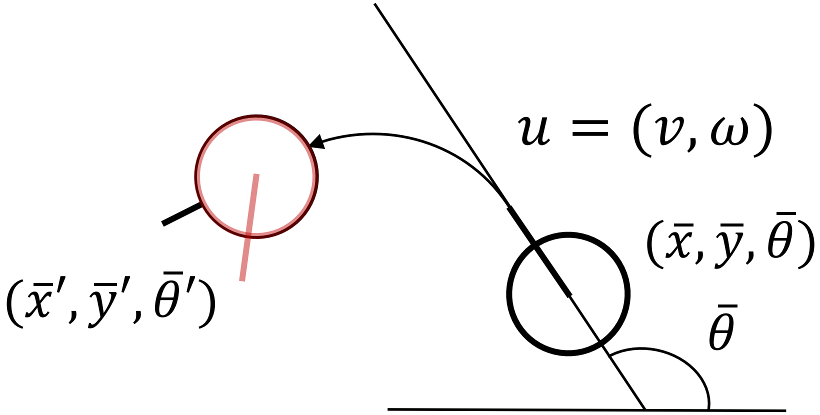




Odometry model: 3DoF – Velocity model: 2DoF

SLAM: Recursive Bayes Filter

Motion model: Standard velocity model



- We define poses in the 2D world as (x, y, θ)
- The movement from $(\bar{x}, \bar{y}, \bar{\theta})$ to $(\bar{x}', \bar{y}', \bar{\theta}')$ can be expressed as: $u = (v, \omega, \boldsymbol{\gamma})$

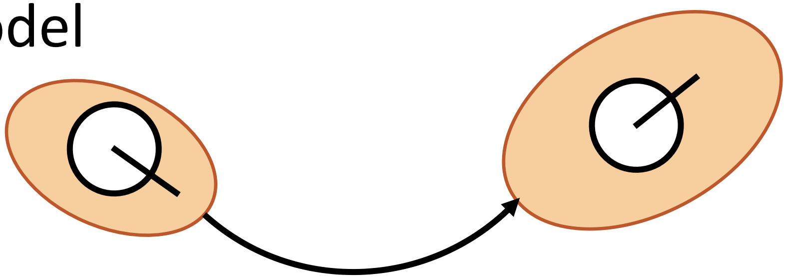
$$\begin{bmatrix} \bar{x}' \\ \bar{y}' \\ \bar{\theta}' \end{bmatrix} = \begin{bmatrix} \bar{x} \\ \bar{y} \\ \bar{\theta} \end{bmatrix} + \begin{bmatrix} -\frac{v}{\omega}\sin\bar{\theta} + \frac{v}{\omega}\sin(\bar{\theta} + \omega\Delta t) \\ \frac{v}{\omega}\cos\bar{\theta} - \frac{v}{\omega}\cos(\bar{\theta} + \omega\Delta t) \\ \omega\Delta t + \boldsymbol{\gamma} \end{bmatrix}$$

SLAM: Recursive Bayes Filter

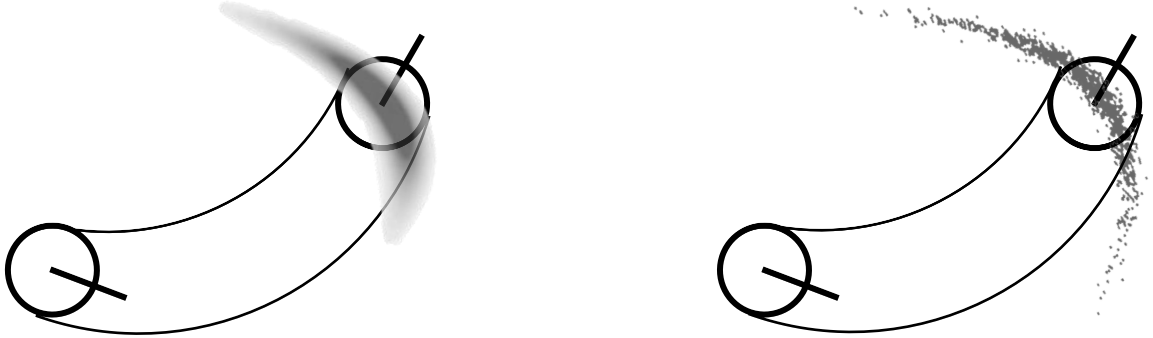
Motion model: Standard velocity model

Probability distribution for $p(x_t \,|x_{t-1}, u_t)$

- Gaussian Model



- Non-Gaussian Model



SLAM: Recursive Bayes Filter

Sensor model: Specifies a posterior probability that the current state x_t (robot+map) affects the observation z_t

$p(z_t \,|x_t)$

Heavily depends on the sensors; Let's assume a Laser Scanner

- Each observation z_t consists of K measurements

$$z_t = \{z_t^1, \ldots, z_t^k\}$$

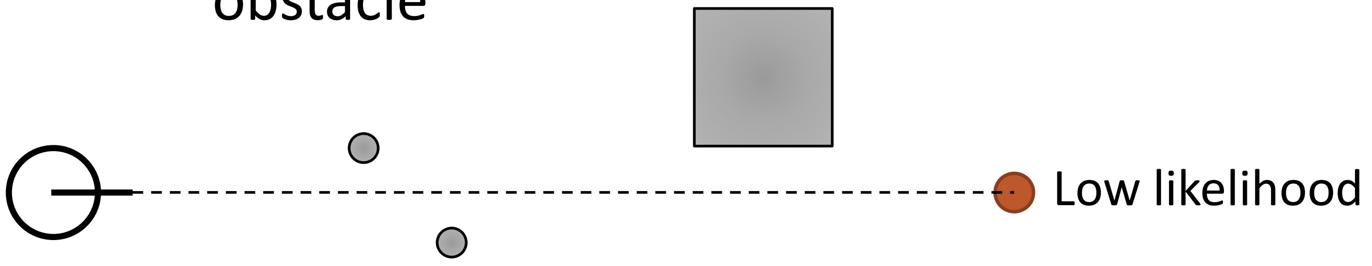
- Assumption: Individual measurements are independent from each other

$$p(z_t \,|x_t) = \prod_{i=1}^{k} p(z_t^i|x_t)$$

SLAM: Recursive Bayes Filter

Sensor model: Beam-Endpoint Model

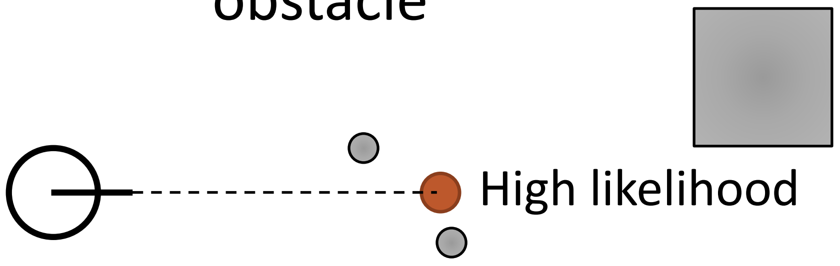
$p(z_t^i|x_t)$: How far away is the end point of the laser beam from the closest obstacle



SLAM: Recursive Bayes Filter

Sensor model: Beam-Endpoint Model

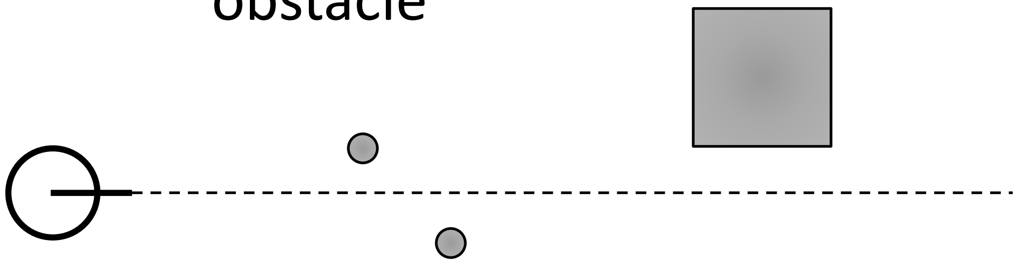
$p(z_t^i | x_t)$: How far away is the end point of the laser beam from the closest obstacle



SLAM: Recursive Bayes Filter

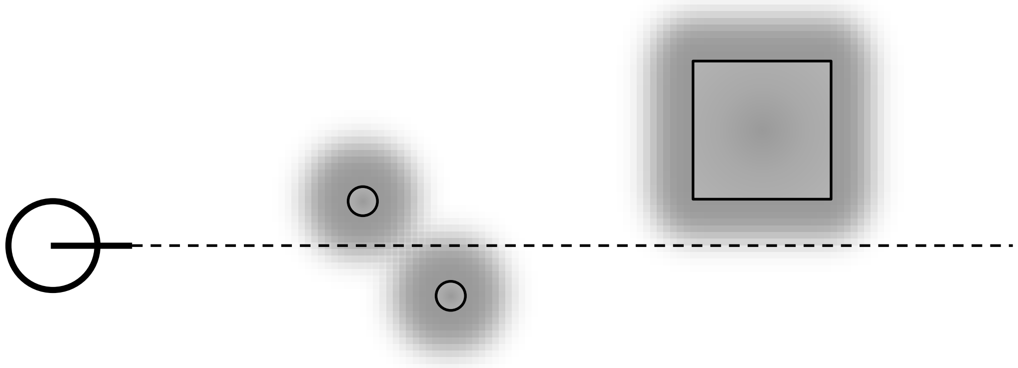
Sensor model: Beam-Endpoint Model

$p(z_t^i|x_t)$: How far away is the end point of the laser beam from the closest obstacle



Map

Likelihood field

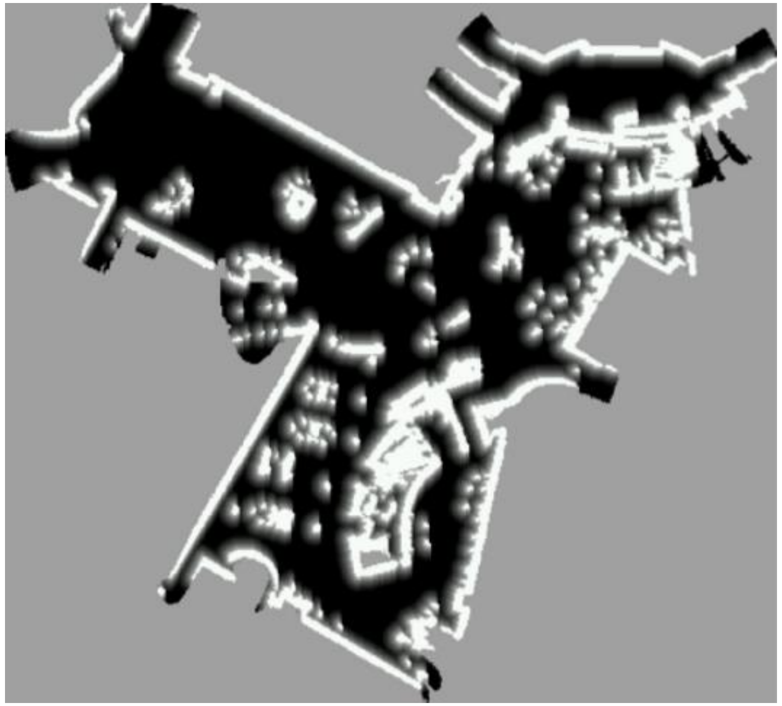


SLAM: Recursive Bayes Filter

Sensor model: Beam-Endpoint Model

Map

Likelihood field

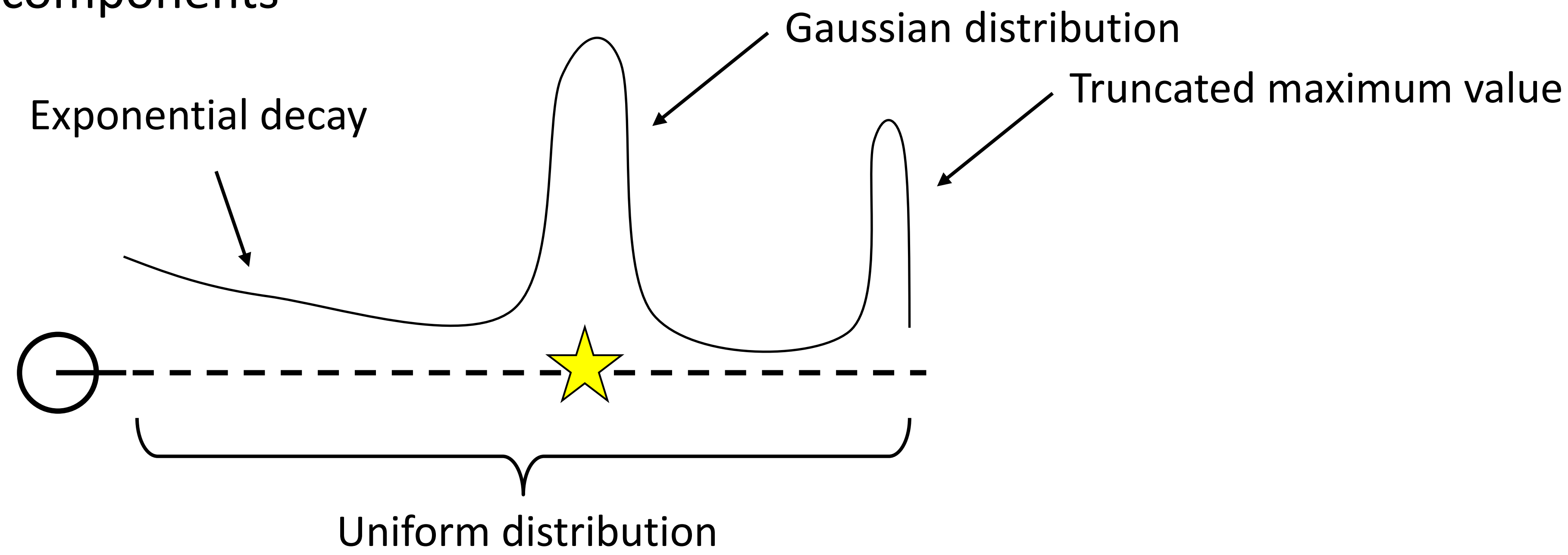


The brighter the value that the beam ends, the higher the $p(z_t^i | x_t)$

SLAM: Recursive Bayes Filter

Sensor model: Ray-cast Model

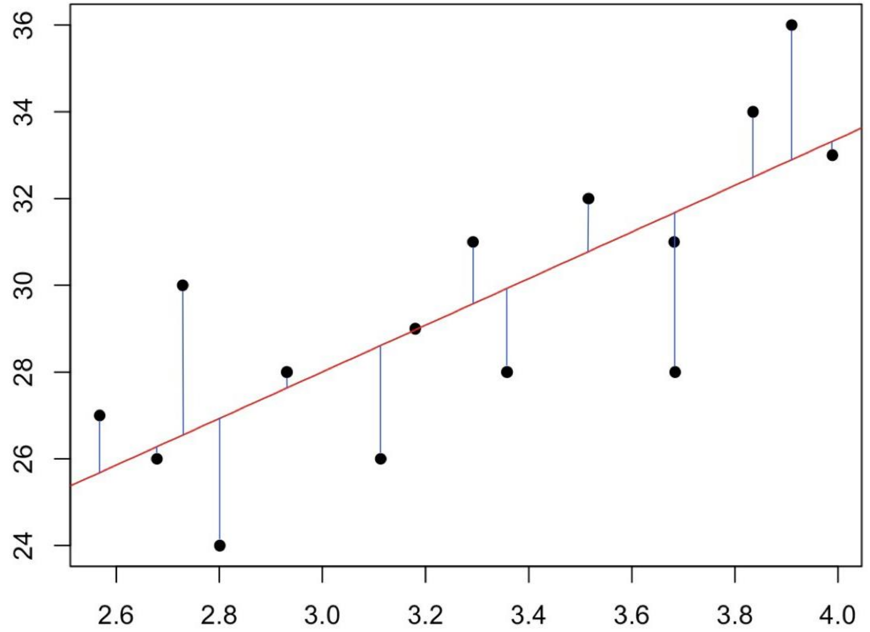
- Additionally considers the first obstacle along the beam
- Mixture of 4 components



SLAM: Least Squares

Least Squares in general:

- An approach for computing a solution for an overdetermined system
  - “More equations than unknowns”
- Minimizes the sum of the squared errors in between measurements and the function that we wish to compute
- Standard approach to a large set of problems



$$\underset{a_{1:k}}{\operatorname{argmin}} \sum_{i=1}^{n} \underbrace{[y_i - f(x_i, a_{1:k})]^2}_{\text{Error function}}$$

SLAM: Least Squares

Least Squares in SLAM:

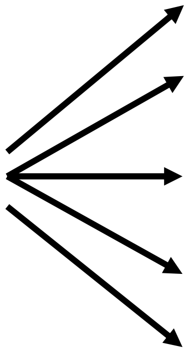
Given:
- A set of n observation functions: $\{f_i(\boldsymbol{x})\}_{, i=1:n}$ where:
  - $\boldsymbol{x}$ is the state vector (e.g., robot+map)
  - $\hat{\boldsymbol{z}}_i = f_i(\boldsymbol{x})$ are the functions that map $\boldsymbol{x}$ to predicted measurements $\hat{\boldsymbol{z}}_i$ (what I am expecting to observe)
- A set of n noisy measurements $\boldsymbol{z}_{i:n}$ about $\boldsymbol{x}$

Goal:
- Estimate the state $\boldsymbol{x}$ which best explains $\boldsymbol{z}_i$

SLAM: Least Squares

Least Squares in SLAM: in other “words”

$\boldsymbol{x}$



$$f_1(\boldsymbol{x}) = \hat{\boldsymbol{z}}_1 \qquad \boldsymbol{z}_1$$

$$f_2(\boldsymbol{x}) = \hat{\boldsymbol{z}}_2 \qquad \boldsymbol{z}_2$$

$$f_3(\boldsymbol{x}) = \hat{\boldsymbol{z}}_3 \qquad \boldsymbol{z}_3$$

$$\ldots \qquad \ldots$$

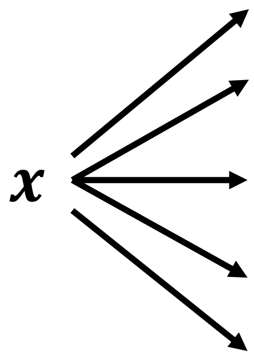
$$f_n(\boldsymbol{x}) = \hat{\boldsymbol{z}}_n \qquad \boldsymbol{z}_n$$

State (unknown)	Predicted measurements	Real measurements

SLAM: Least Squares

Example

- $\boldsymbol{x}$: position of 3D world points and
  6DoF robot poses
- $\boldsymbol{z}_i$: depth measurements of the 3D
  points recorded by a LiDAR
- $\{f_i(\boldsymbol{x})\}$: LiDAR projection function
- Estimate the most likely position of
  3D points based on the laser projections



$$f_1(\boldsymbol{x}) = \hat{\boldsymbol{z}}_1 \qquad \boldsymbol{z}_1$$
$$f_2(\boldsymbol{x}) = \hat{\boldsymbol{z}}_2 \qquad \boldsymbol{z}_2$$
$$f_3(\boldsymbol{x}) = \hat{\boldsymbol{z}}_3 \qquad \boldsymbol{z}_3$$
$$\dots \qquad \dots$$
$$f_n(\boldsymbol{x}) = \hat{\boldsymbol{z}}_n \qquad \boldsymbol{z}_n$$

SLAM: Least Squares

Error function

- We can define the error of a single measurement as:
$$\boldsymbol{e}_i = \boldsymbol{z}_i - \widehat{\boldsymbol{z}}_i = \boldsymbol{z}_i - f_i(\boldsymbol{x})$$
for each measurement
- We assume zero-mean Gaussian error with information matrix (inverse of covariance): $\boldsymbol{\Omega}_i$
- The squared error is:
$$e_i = \boldsymbol{e}_i^T \boldsymbol{\Omega}_i \boldsymbol{e}_i$$

weighted since measurements may not have the same uncertainty

SLAM: Least Squares

Minimization over all measurements

- Find the state $\boldsymbol{x}^*$ that minimizes the error of all measurements

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}}{\operatorname{argmin}}\, F(\boldsymbol{x})$$ ← Global error (scalar)

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}}{\operatorname{argmin}} \sum_{i=1}^{n} e_i(x)$$ ← Squared error terms (scalar)

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}}{\operatorname{argmin}} \sum_{i=1}^{n} \boldsymbol{e}_i^T(\boldsymbol{x})\boldsymbol{\Omega}_i\boldsymbol{e}_i(\boldsymbol{x})$$ ← Error terms (vector)

SLAM: Least Squares

Minimization over all measurements

- Find the state $\boldsymbol{x}^*$ that minimizes the error of all measurements

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}}{\operatorname{argmin}} \sum_{i=1}^{n} \boldsymbol{e}_i^T(\boldsymbol{x}) \boldsymbol{\Omega}_i \boldsymbol{e}_i(\boldsymbol{x})$$

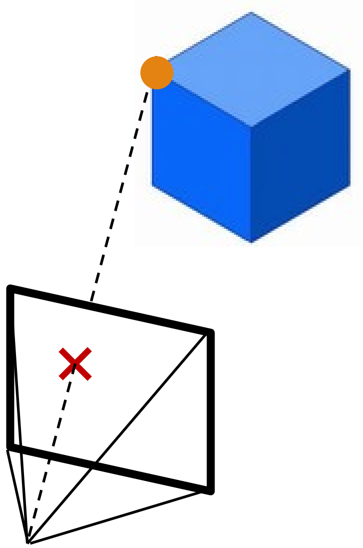
- $e_i(\boldsymbol{x})$ is typically non-linear
- no closed-form solution

} Iterative local linearizations

SLAM: Bundle Adjustment

Bundle Adjustment (BA) is a least square approach, where

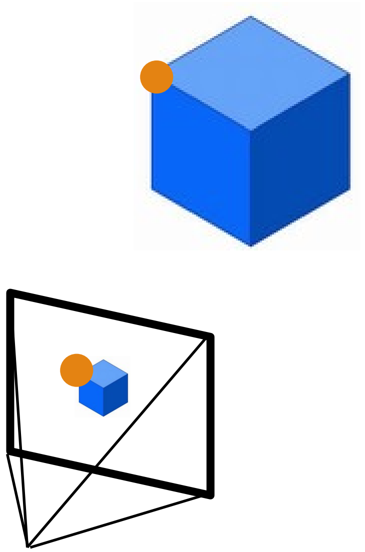
- State contains both the robot poses and the map
- Error is computed as the displacement of representative points captured by the camera and their projection
  - $\hat{\boldsymbol{z}}_i = f_i(\boldsymbol{x})$



SLAM: Bundle Adjustment

Bundle Adjustment (BA) is a least square approach, where

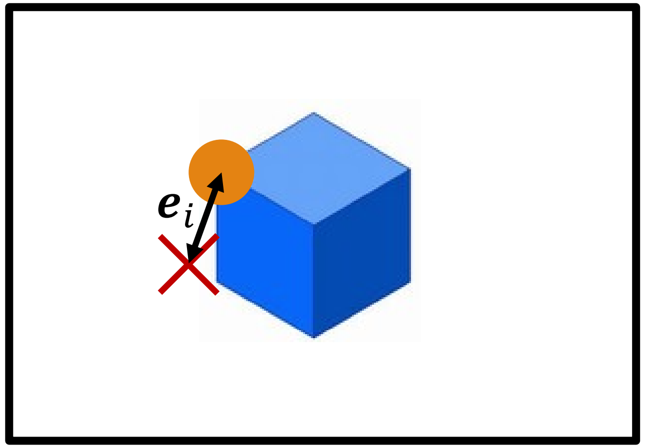
- State contains both the robot poses and the map
- Error is computed as the displacement of representative points captured by the camera and their projection
    - $\hat{\boldsymbol{z}}_i = f_i(\boldsymbol{x})$
    - $\boldsymbol{z}_i$



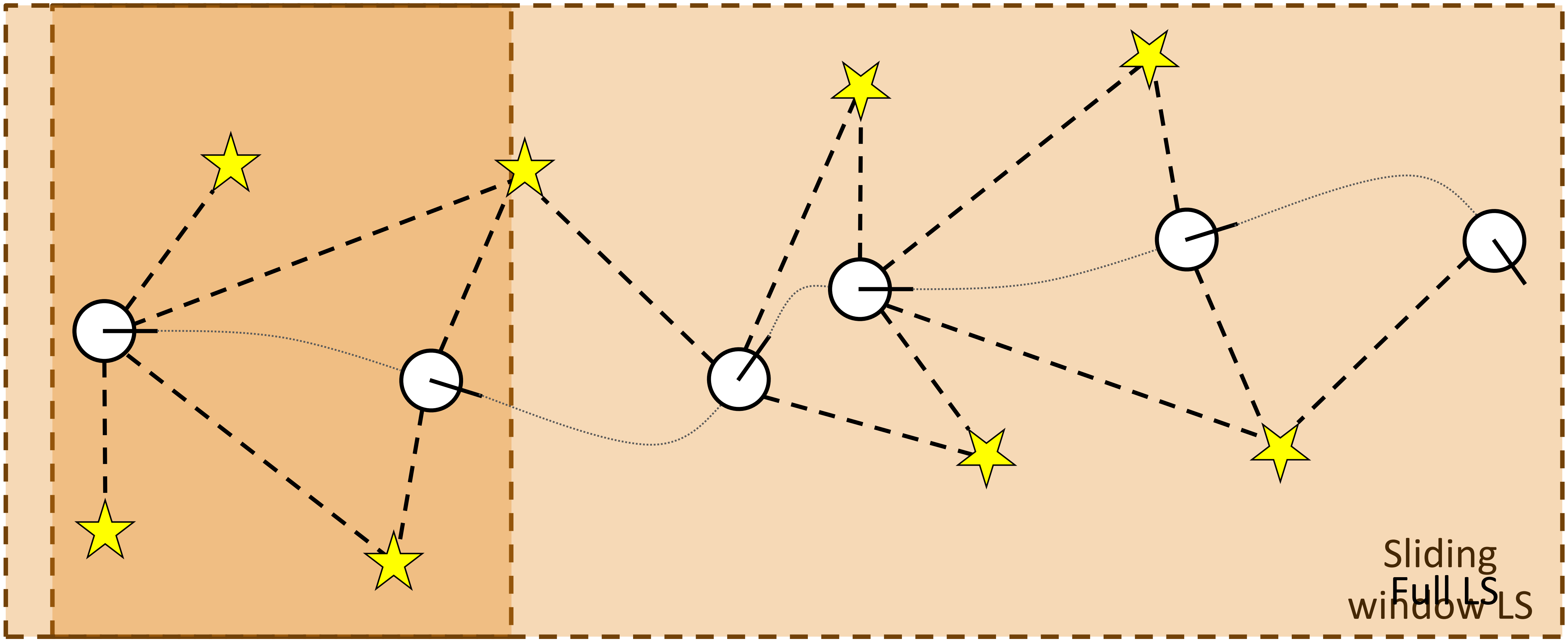
SLAM: Bundle Adjustment

Bundle Adjustment (BA) is a least square approach, where

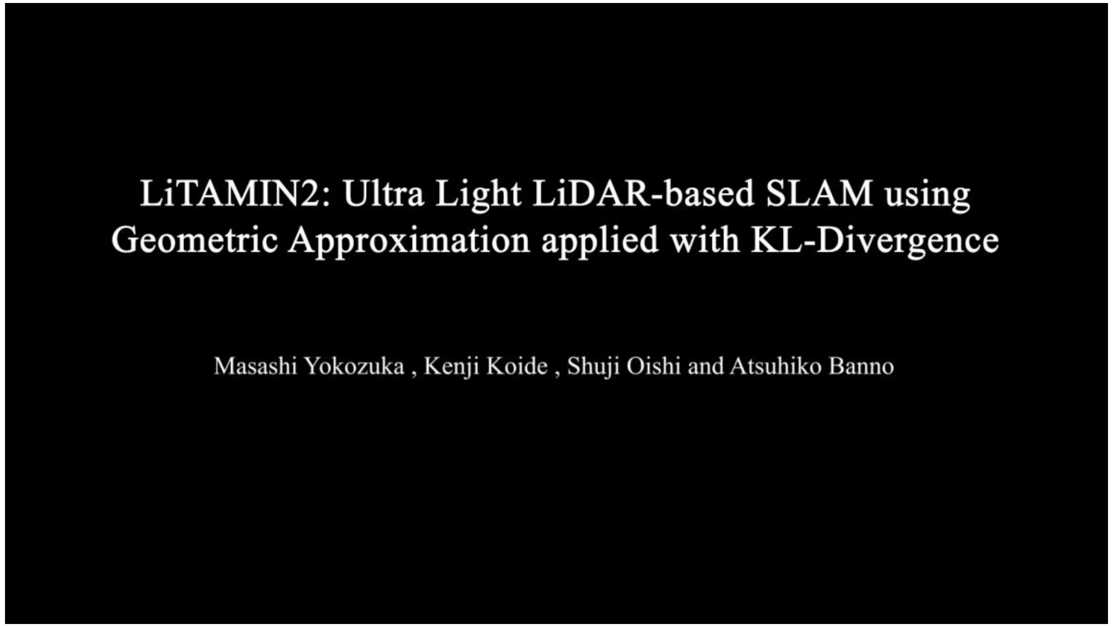
- State contains both the robot poses and the map
- Error is computed as the displacement of representative points captured by the camera and their projection
    - $\hat{\boldsymbol{z}}_i = f_i(\boldsymbol{x})$
    - $\boldsymbol{z}_i$
    - $\boldsymbol{e}_i = \boldsymbol{z}_i - f_i(\boldsymbol{x})$



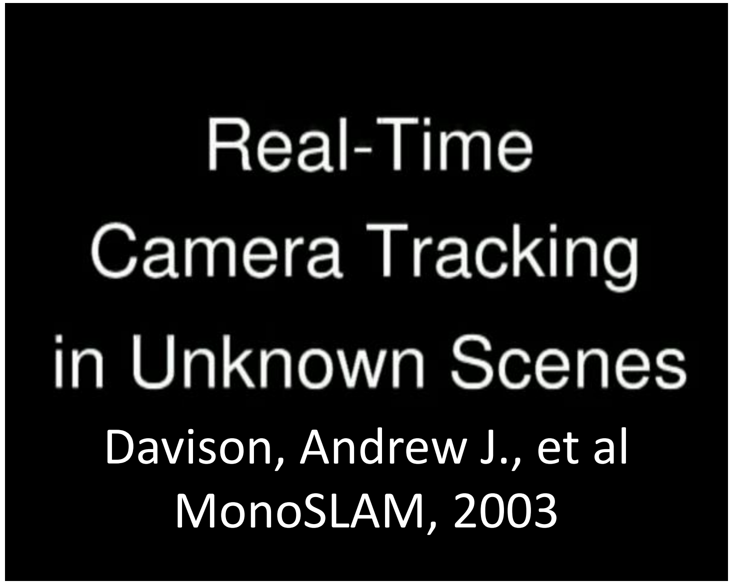
SLAM: Sliding-window Least Squares



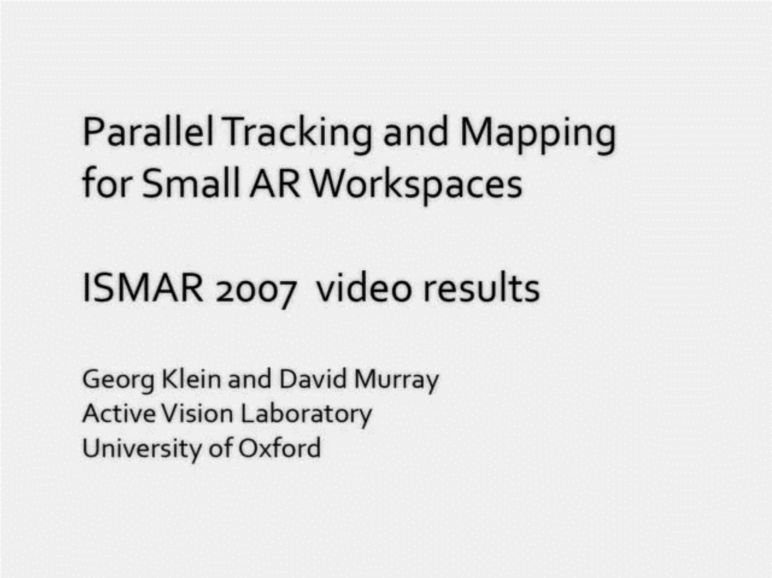
SLAM: Some of the most representative approaches

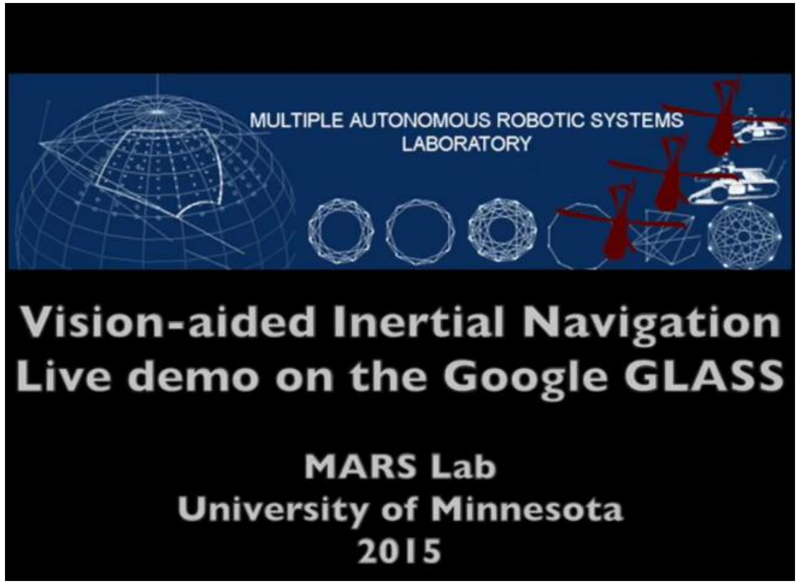


SLAM: Some of the most representative approaches



Davison, Andrew J., et al
MonoSLAM, 2003





SLAM: Some of the most representative approaches

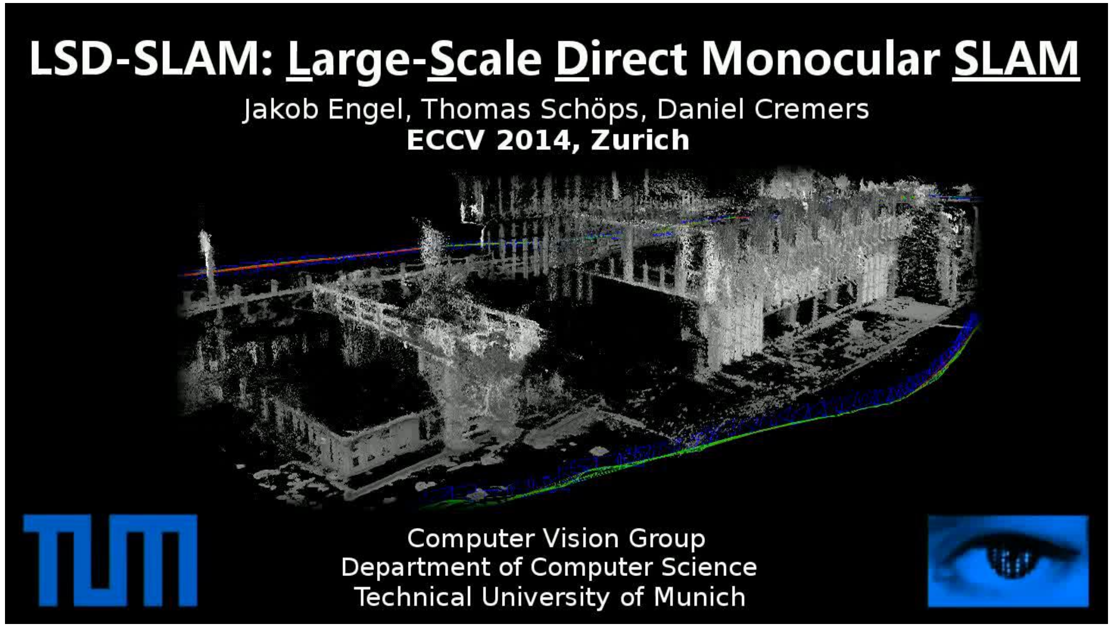
StructSLAM:

Visual SLAM with Building Structure Lines

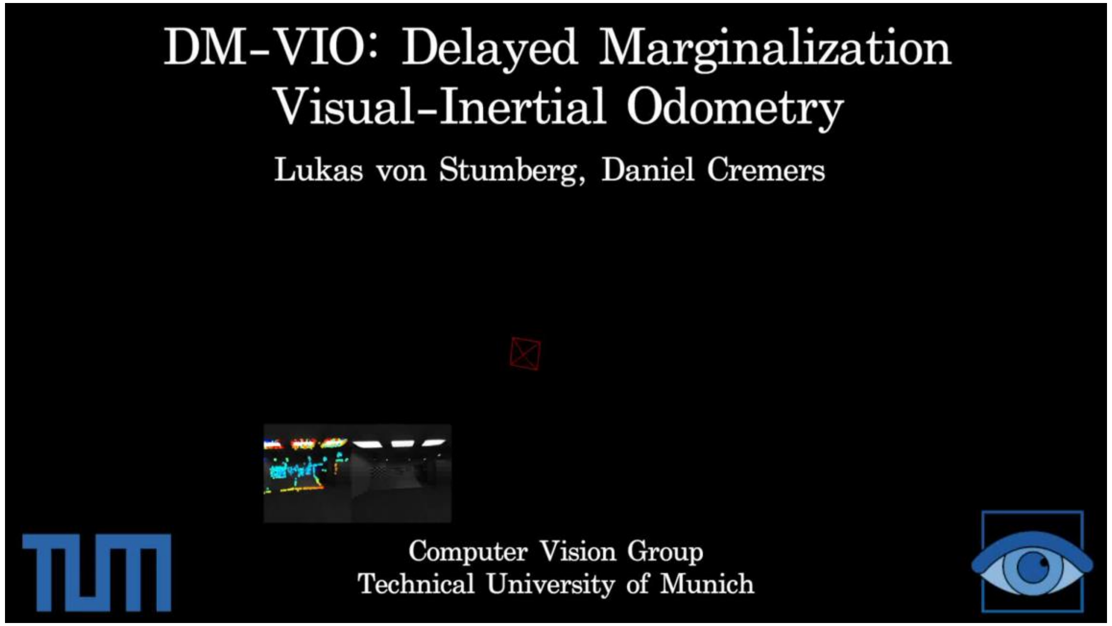
Hui Zhong Zhou, Danping Zou et al.

Shanghai Key Laboratory of Navigation and Location Based Services
Shanghai Jiao Tong University
Apirl,2014

SLAM: Some of the most representative approaches



SLAM: Some of the most representative approaches



SLAM: Some of the most representative approaches