Practical Course WS12/13

Introduction to Monte Carlo Localization

Cyrill Stachniss and Luciano Spinello

UNI FREIBURG

State Estimation

- Estimate the state x of a system given observations z and controls u
- **Goal:**

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

Bayes Filter

- General framework for state estimation
- Prediction step based on controls
- Correction step based on measurements
- Recursive structure:

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

Bayes Filter Derivation

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

Prediction and Correction Step

- Bayes filter is often formulated as a two step process
- **Prediction step**

$$\overline{bel}(x_t) = \int p(x_t \mid u_t, x_{t-1})\; bel(x_{t-1})\, dx_{t-1}$$

- **Correction step**

$$bel(x_t) = \eta\, p(z_t \mid x_t)\; \overline{bel}(x_{t-1})$$

Motion and Observation Model

- Prediction step

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

- Correction step

$$bel(x_t) = \eta\, \underbrace{p(z_t \mid x_t)}_{\text{sensor or observation model}}\ \overline{bel}(x_{t-1})$$

Different Realizations

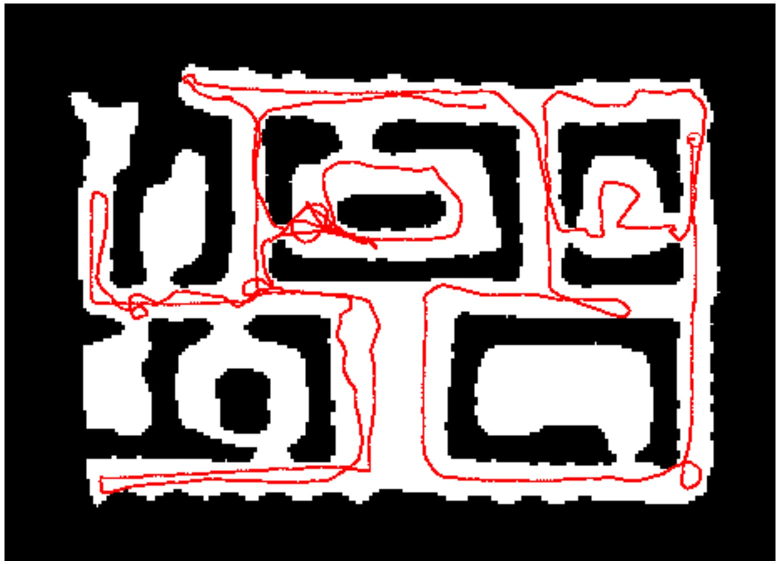
- The Bayes filter is a **framework** for recursive state estimation
- There are **different realizations**
- **Different properties**
  - Linear vs. non-linear models for motion and observation models
  - Gaussian distributions only?
  - Parametric vs. non-parametric filters
  - …

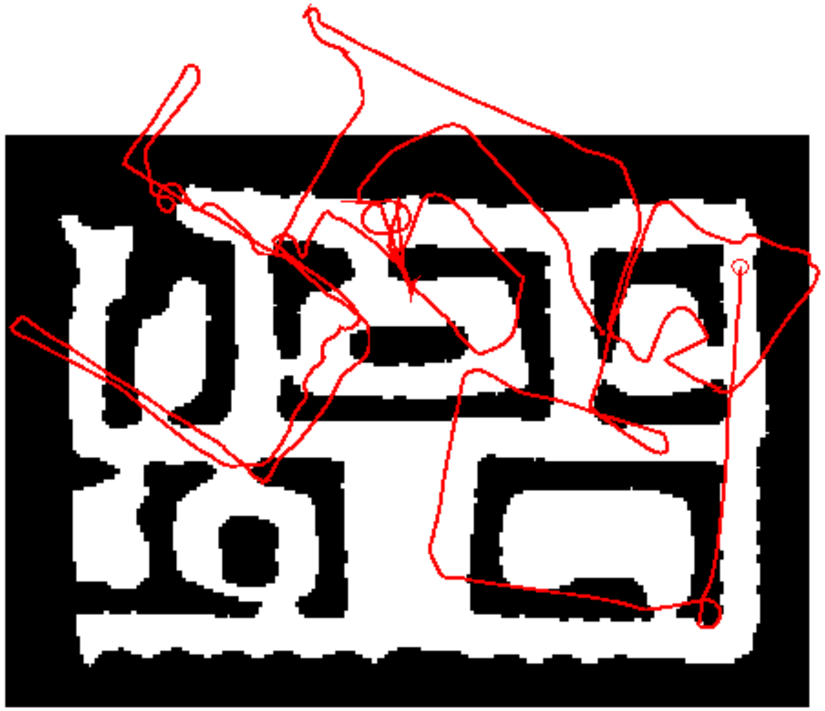
Motion Model

$$\overline{bel}(x_t) = \int \boxed{p(x_t \mid u_t, x_{t-1})}\, bel(x_{t-1})\, dx_{t-1}$$

Robot Motion Models

- Robot motion is inherently uncertain
- How can we model this uncertainty?





Probabilistic Motion Models

- Specifies a posterior probability that action *u* carries the robot from *x* to *x′*.

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

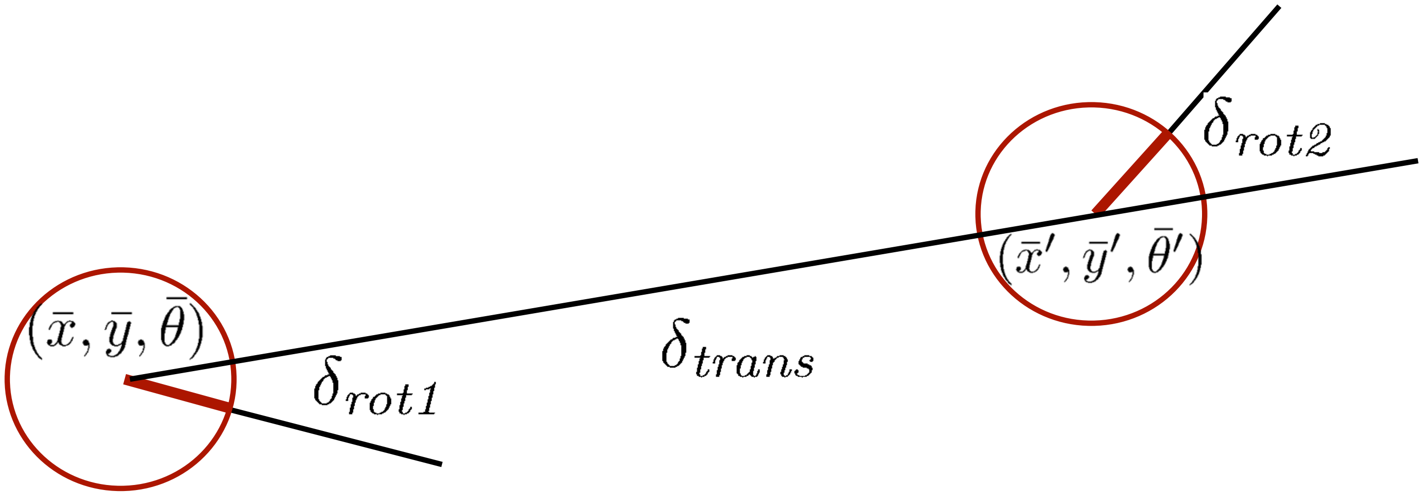
Typical Motion Models

- In practice, one often finds two types of motion models:
  - **Odometry-based**
  - **Velocity-based**
- Odometry-based models for systems that are equipped with wheel encoders
- Velocity-based when no wheel encoders are available

Odometry Model

- Robot moves from $(\bar{x}, \bar{y}, \bar{\theta})$ to $(\bar{x}', \bar{y}', \bar{\theta}')$
- Odometry information $u = (\delta_{rot1}, \delta_{trans}, \delta_{rot2})$

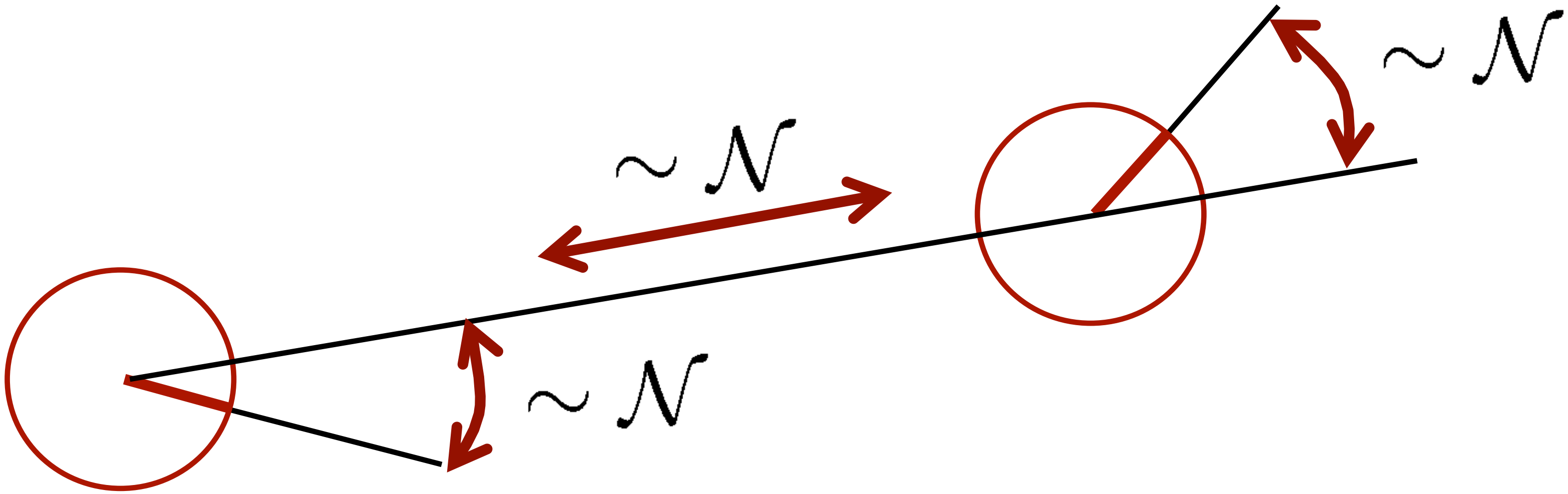
$$
\begin{aligned}
\delta_{trans} &= \sqrt{(\bar{x}' - \bar{x})^2 + (\bar{y}' - \bar{y})^2} \\
\delta_{rot1} &= \operatorname{atan2}(\bar{y}' - \bar{y}, \bar{x}' - \bar{x}) - \bar{\theta} \\
\delta_{rot2} &= \bar{\theta}' - \bar{\theta} - \delta_{rot1}
\end{aligned}
$$



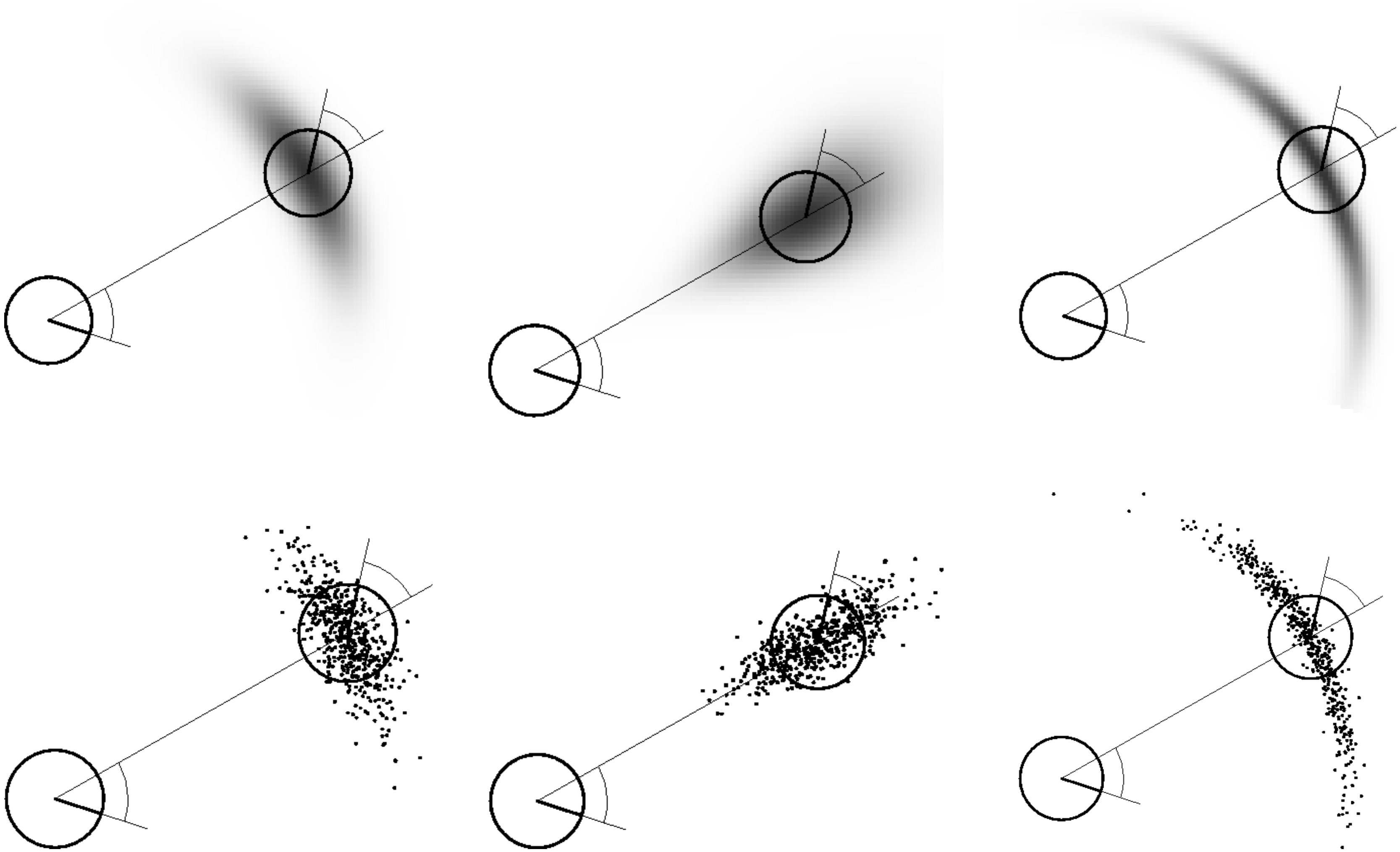
Probability Distribution

- Noise in odometry $u = (\delta_{rot1}, \delta_{trans}, \delta_{rot2})$
- Example: Gaussian noise

$$u \sim \mathcal{N}(0, \Sigma)$$



Examples (Odometry-Based)



Sensor Model

$$bel(x_t) = \eta \boxed{p(z_t \mid x_t)} \overline{bel}(x_{t-1})$$

Model for Laser Scanners

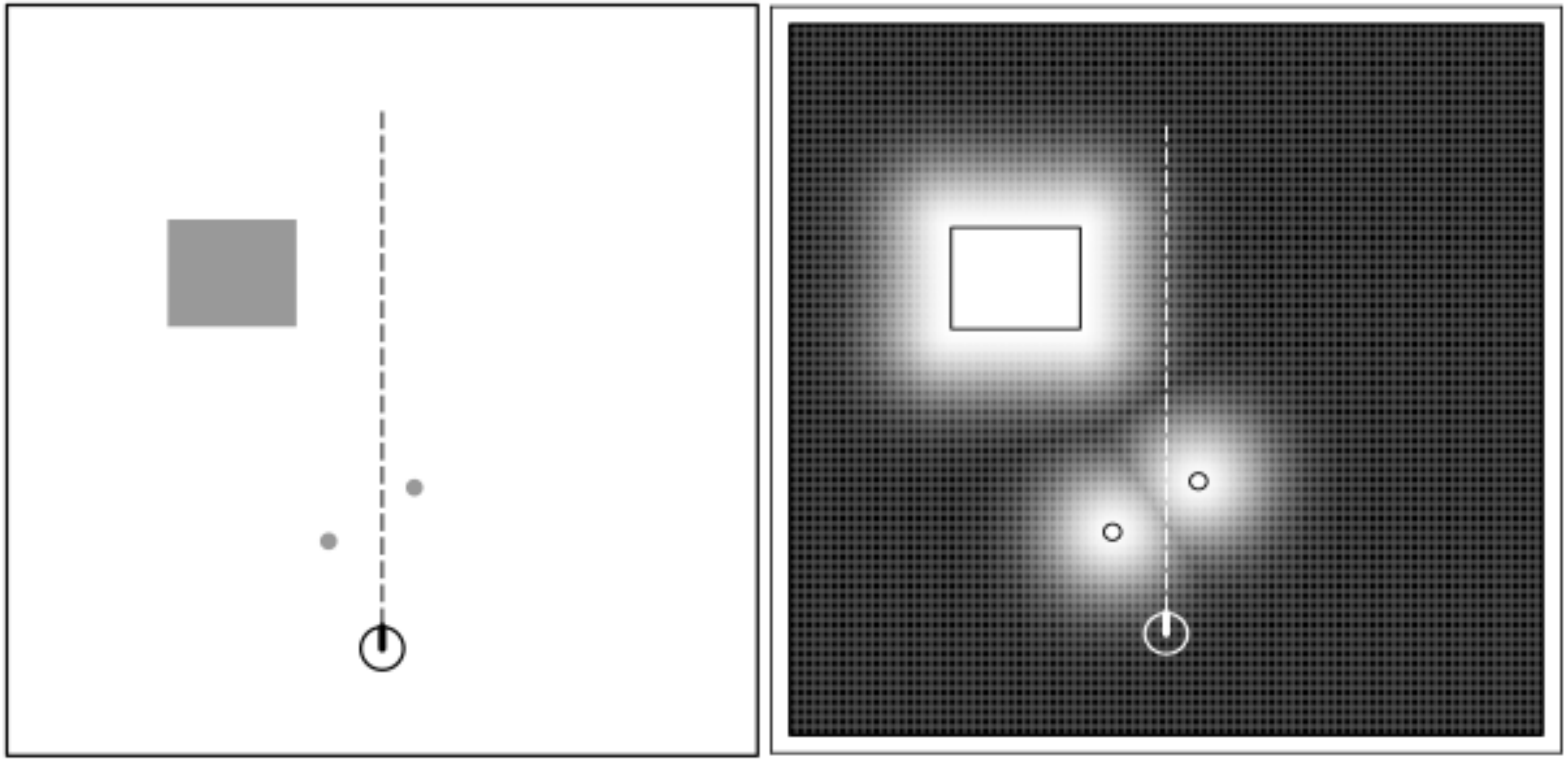
- Scan *z* consists of *K* measurements.

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

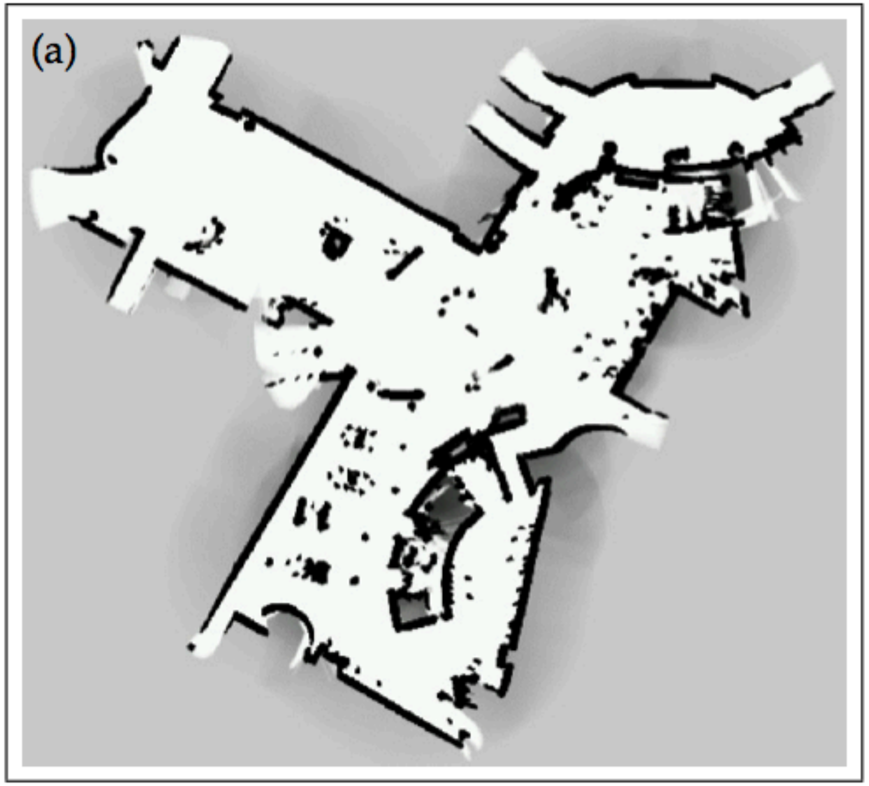
- Individual measurements are independent given the robot position

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

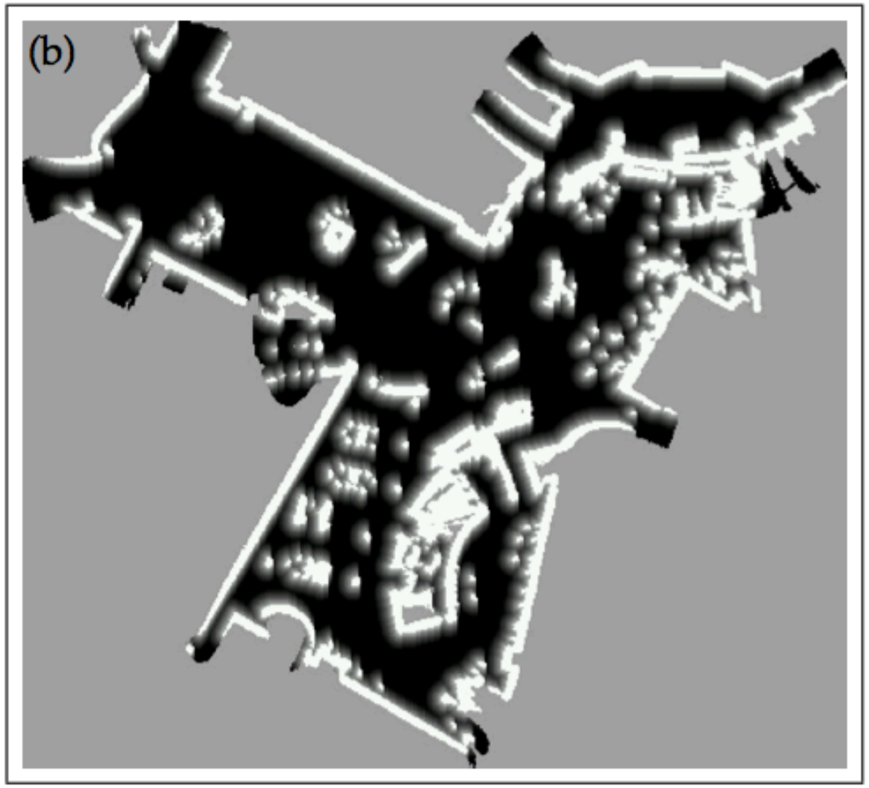
Beam-Endpoint Model



Beam-Endpoint Model



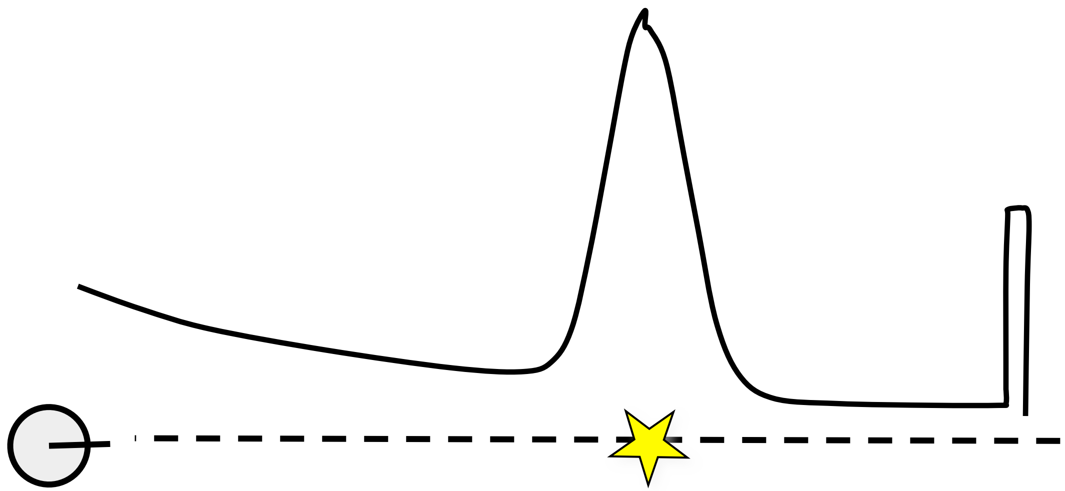
map



likelihood field

Ray-cast Model

- Ray-cast model considers the first obstacle long the line of sight
- Mixture of four models



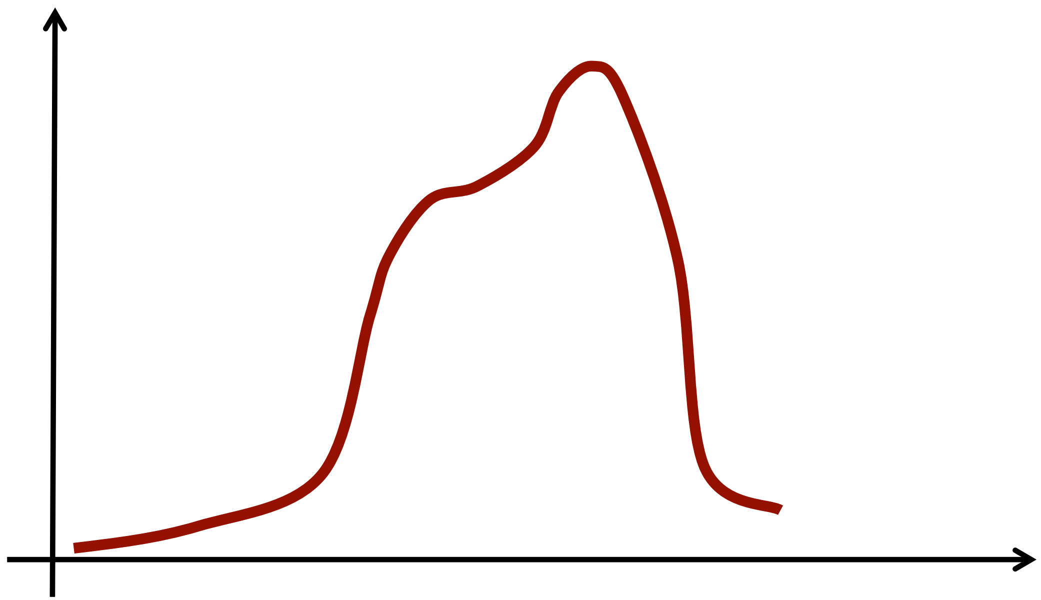
Particle Filter

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

- Realization of the Bayes filter
- Non-parametric approach
- Models arbitrary distributions

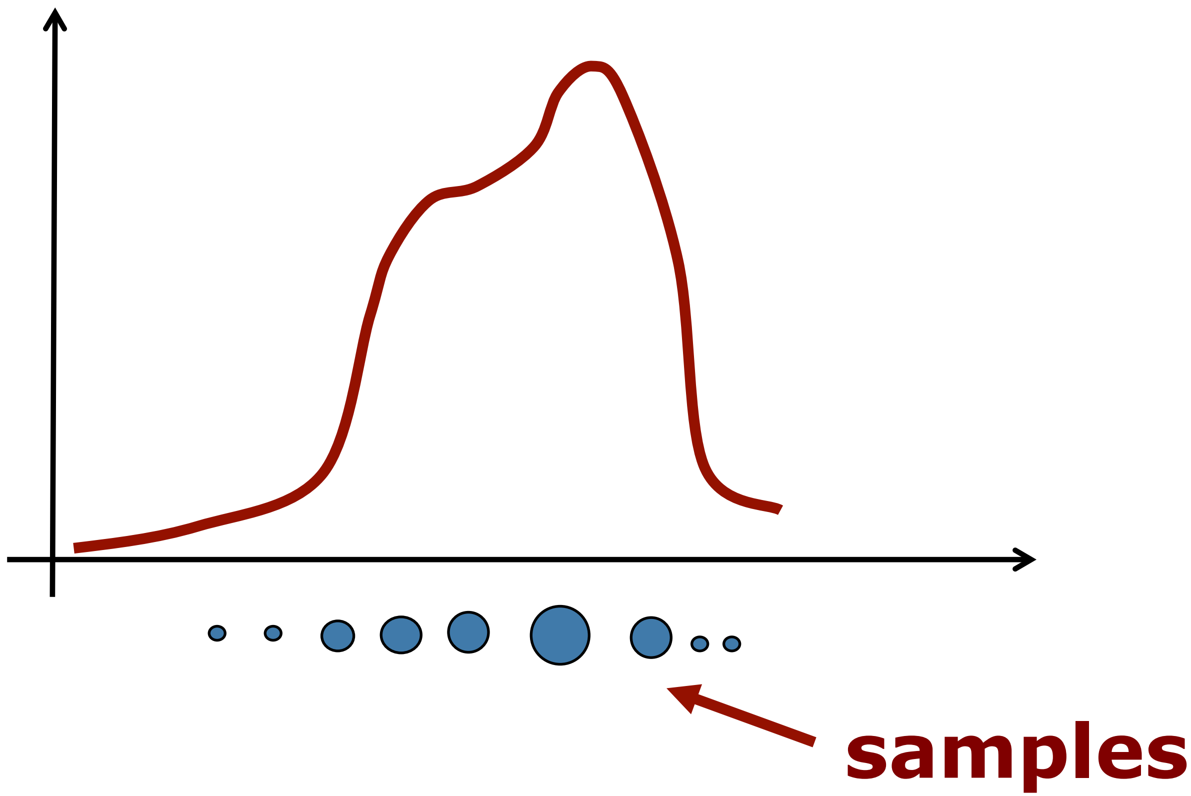
Motivation

- Goal: approach for dealing with **arbitrary distributions**



Key Idea: Samples

- Use **multiple samples** to represent arbitrary distributions



Particle Set

- Set of weighted samples

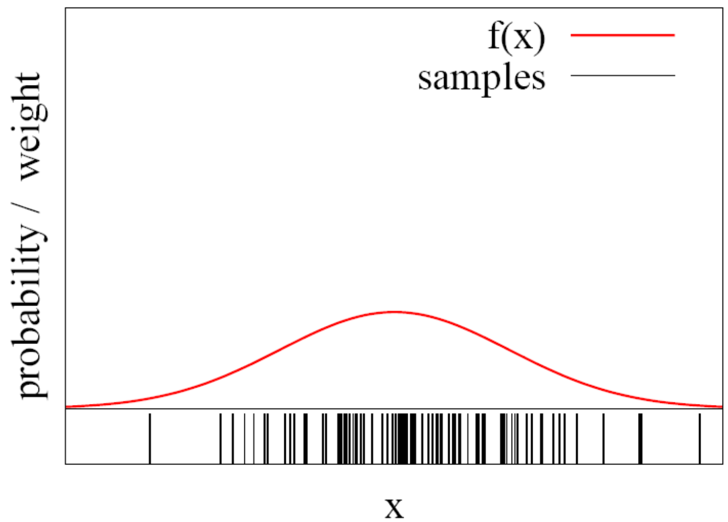
$$\mathcal{X} = \left\{ \left\langle \underbrace{x^{[i]}}_{\text{state hypothesis}}, \underbrace{w^{[i]}}_{\text{importance weight}} \right\rangle \right\}_{i=1,\dots,N}$$

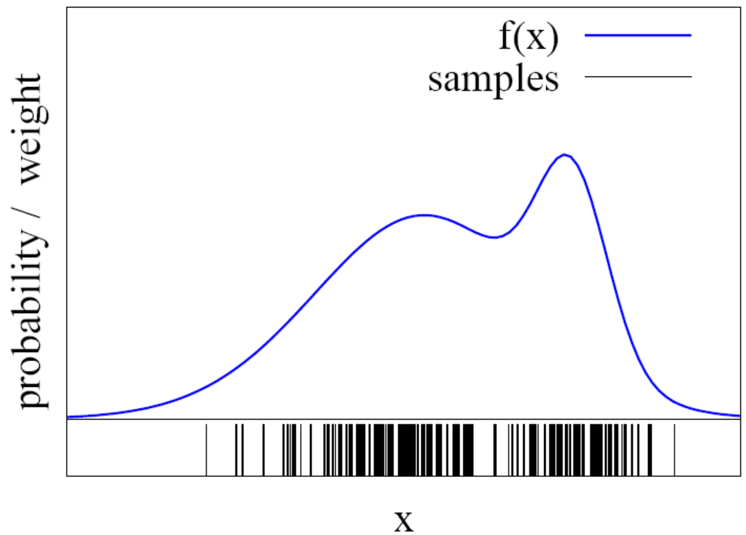
- The samples represent the posterior

$$p(x) = \sum_{i=1}^{N} w^{[i]} \, \delta_{x^{[i]}}(x)$$

Particles for Approximation

- Particles for function approximation





- The more particles fall into an interval, the higher its probability density

Particle Filter

- Recursive Bayes filter
- Non-parametric approach
- Models the distribution by samples
- Prediction: propagate the samples given the motion model (proposal)
- Correction: weighting by considering the observation

The more samples we use, the better is the estimate!

Monte Carlo Localization

- Each particle is a pose hypothesis
- Proposal is the motion model

$$x_t^{[i]} \sim p(x_t \mid x_{t-1}, u_t)$$

- Correction via the observation model

$$w_t^{[i]} \propto p(z_t \mid x_t, m)$$

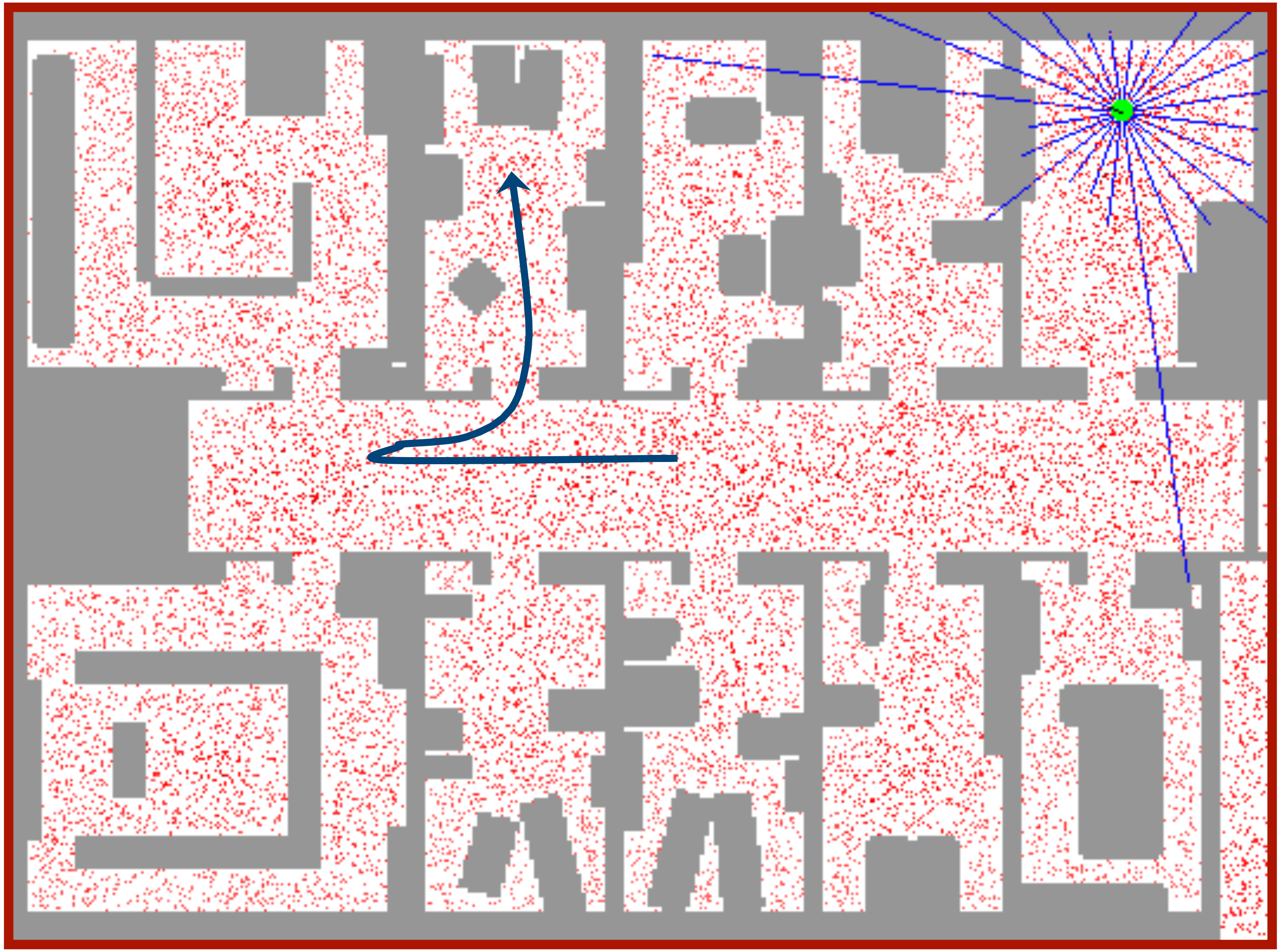
- Resampling: “Replace unlikely samples by more likely ones”

Particle Filter for Localization

Particle_filter$(\mathcal{X}_{t-1}, u_t, z_t)$**:**

$\bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset$
  for $m = 1$ to M do
    sample $x_t^{[m]} \sim p(x_t \mid u_t, x_{t-1}^{[m]})$
    $w_t^{[m]} = p(z_t \mid x_t^{[m]})$
    $\bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \langle x_t^{[m]}, w_t^{[m]} \rangle$
  endfor
  for $m = 1$ to M do
    draw i with probability $\propto w_t^{[i]}$
    add $x_t^{[i]}$ to $\mathcal{X}_t$
  endfor
  return $\mathcal{X}_t$

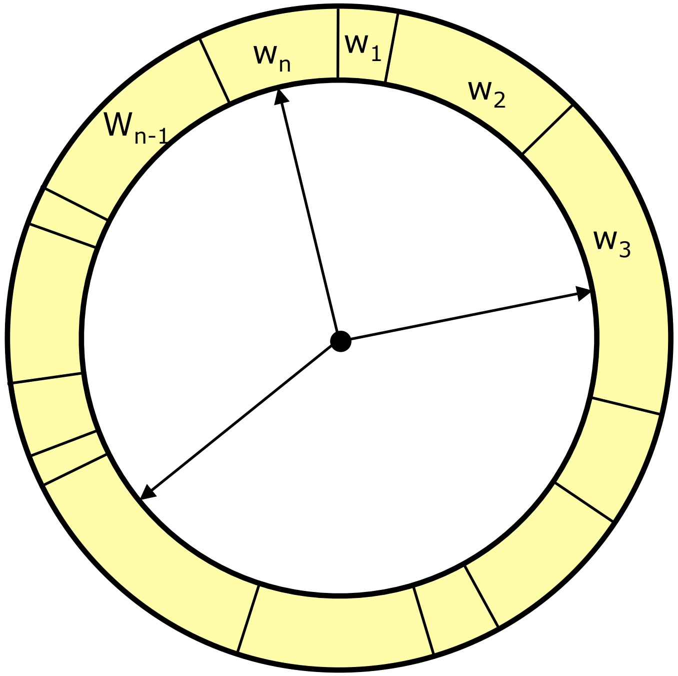
Application: Particle Filter for Localization (Known Map)

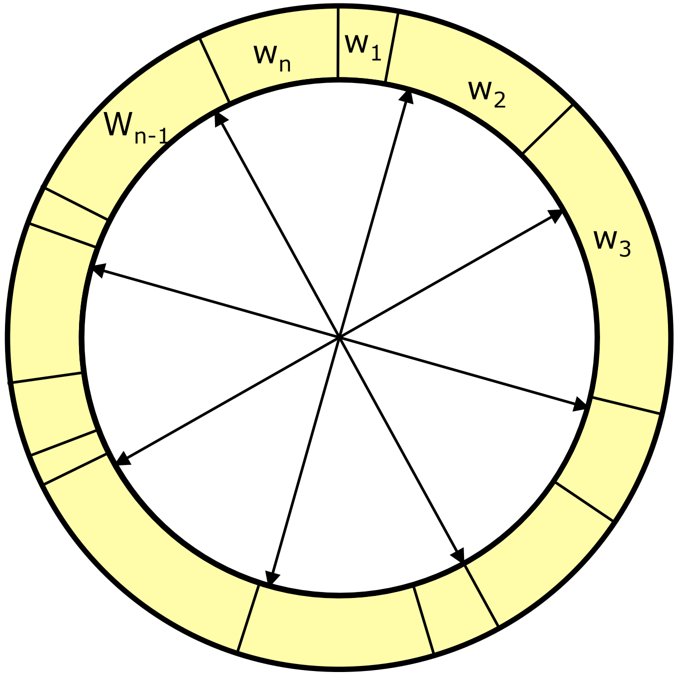


Resampling

- Survival of the fittest: “Replace unlikely samples by more likely ones”
- “Trick” to avoid that many samples cover unlikely states
- Needed as we have a limited number of samples

Resampling





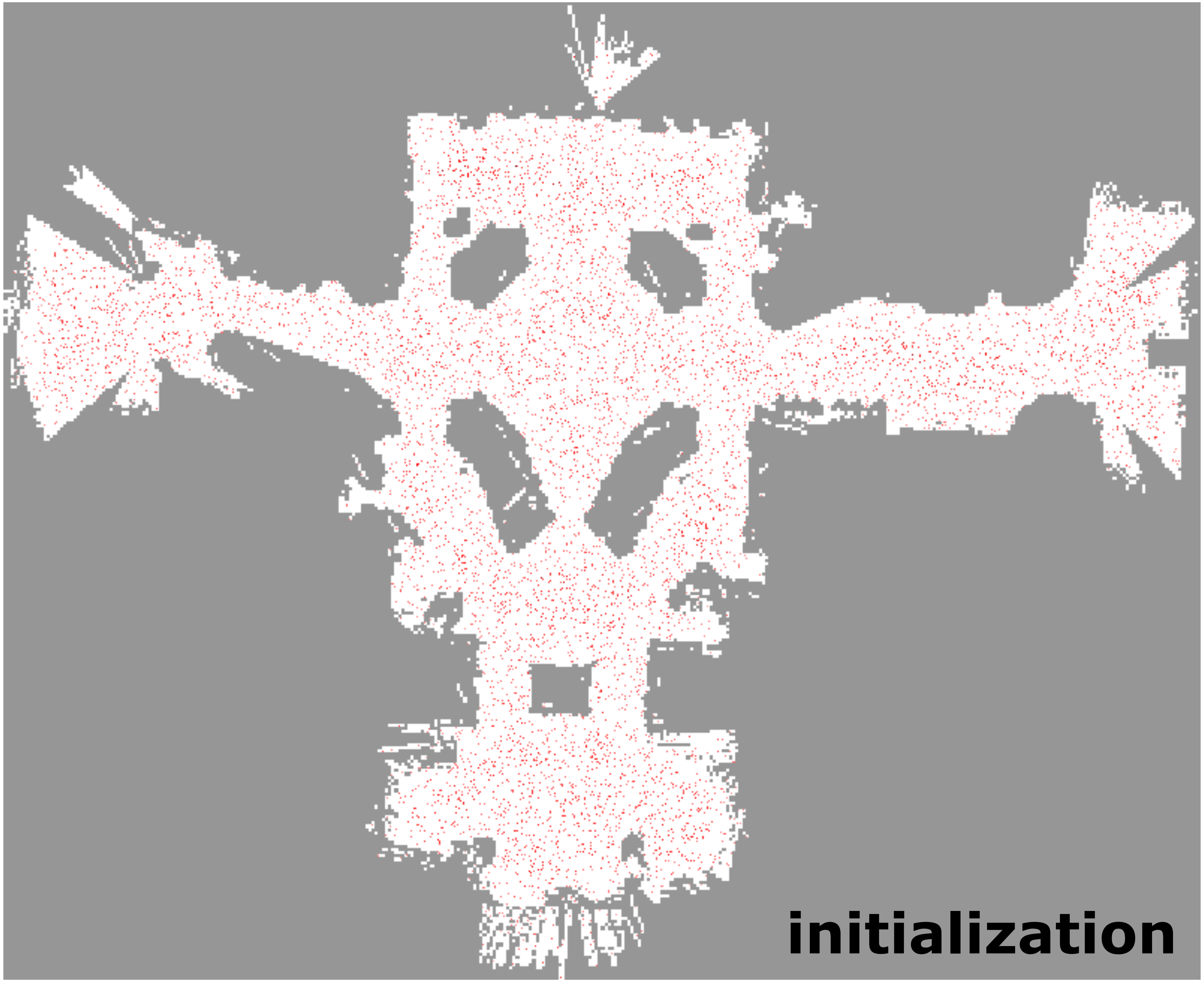
- Roulette wheel
- Binary search
- O(n log n)

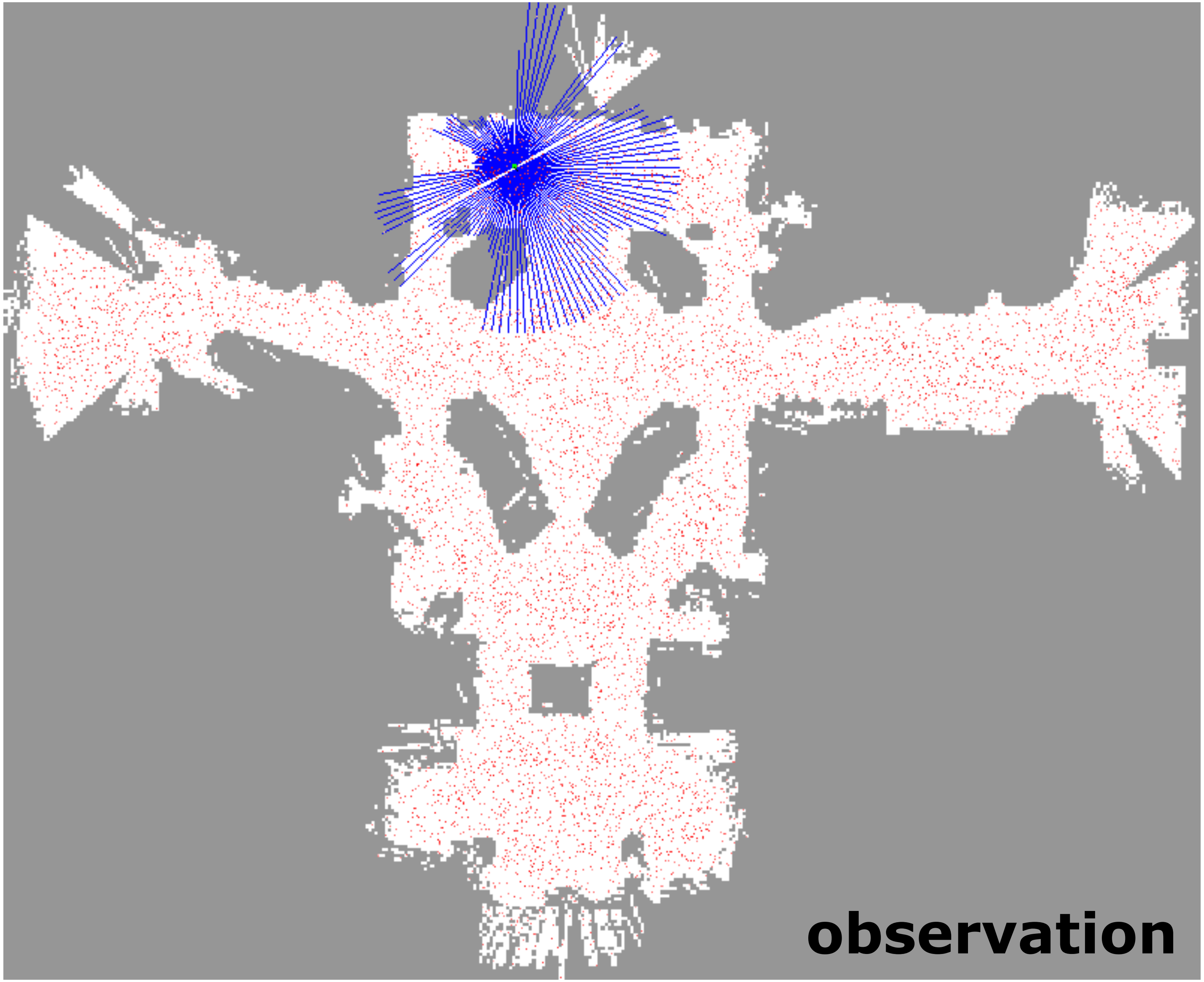
- Stochastic universal sampling
- Low variance
- O(n)

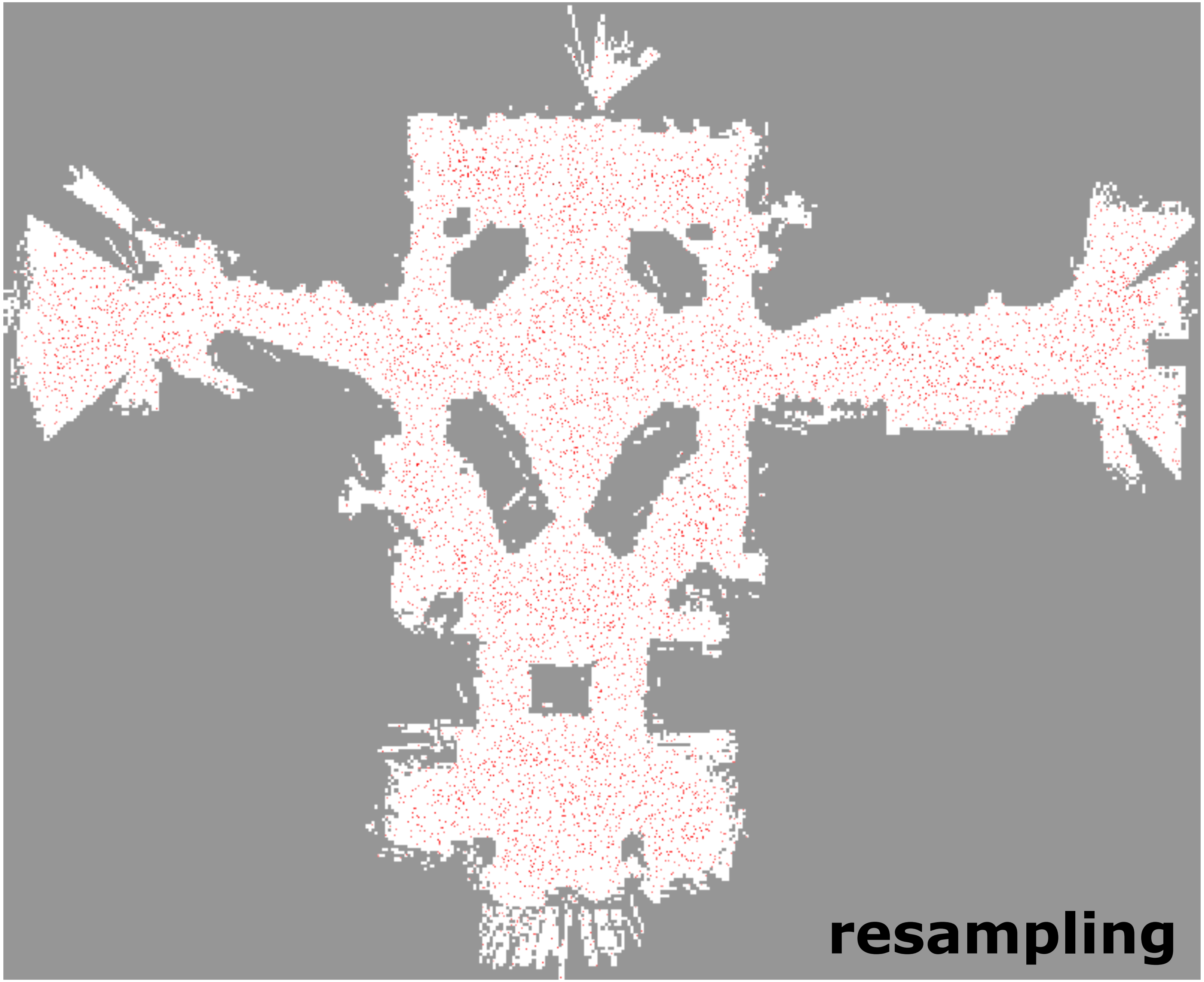
Low Variance Resampling

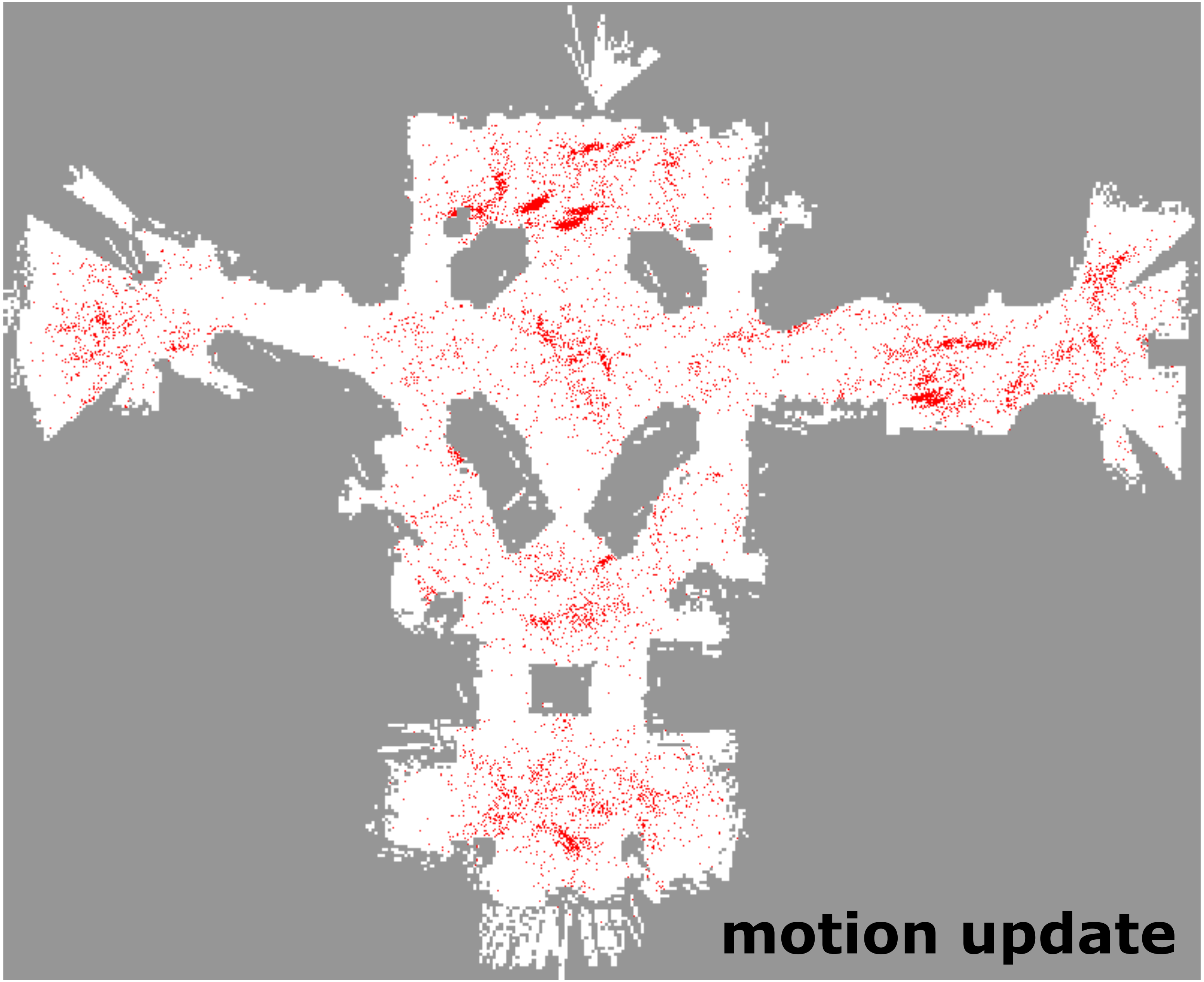
Low_variance_resampling$(\mathcal{X}_t, \mathcal{W}_t)$**:**

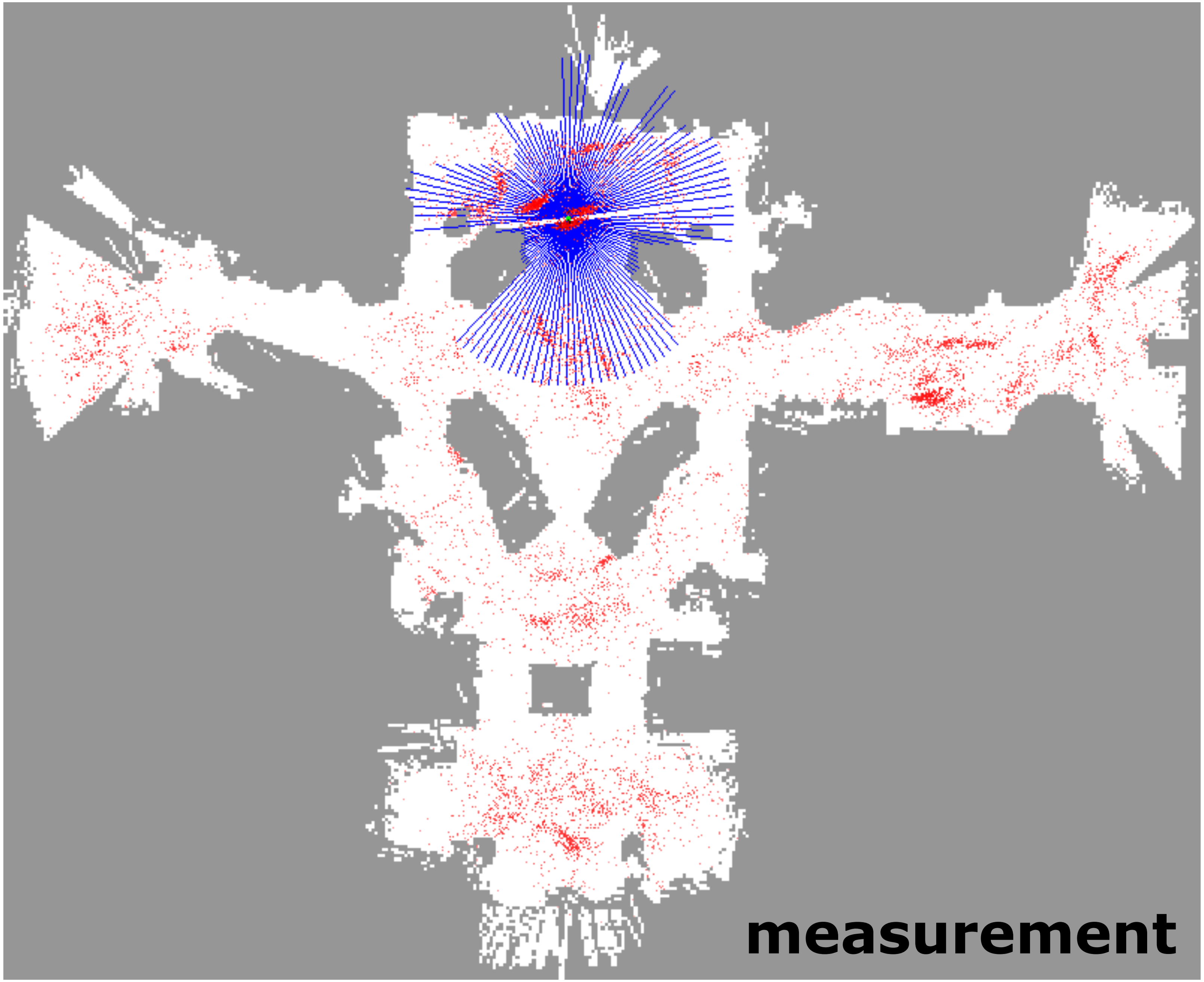
$\bar{\mathcal{X}}_t = \emptyset$
$r = \mathrm{rand}(0; M^{-1})$
$c = w_t^{[1]}$
$i = 1$
for $m = 1$ *to* M *do*
  $U = r + (m-1) \cdot M^{-1}$
  while $U > c$
    $i = i + 1$
    $c = c + w_t^{[i]}$
  endwhile
  add $x_t^{[i]}$ to $\bar{\mathcal{X}}_t$
endfor
return $\bar{\mathcal{X}}_t$


initialization

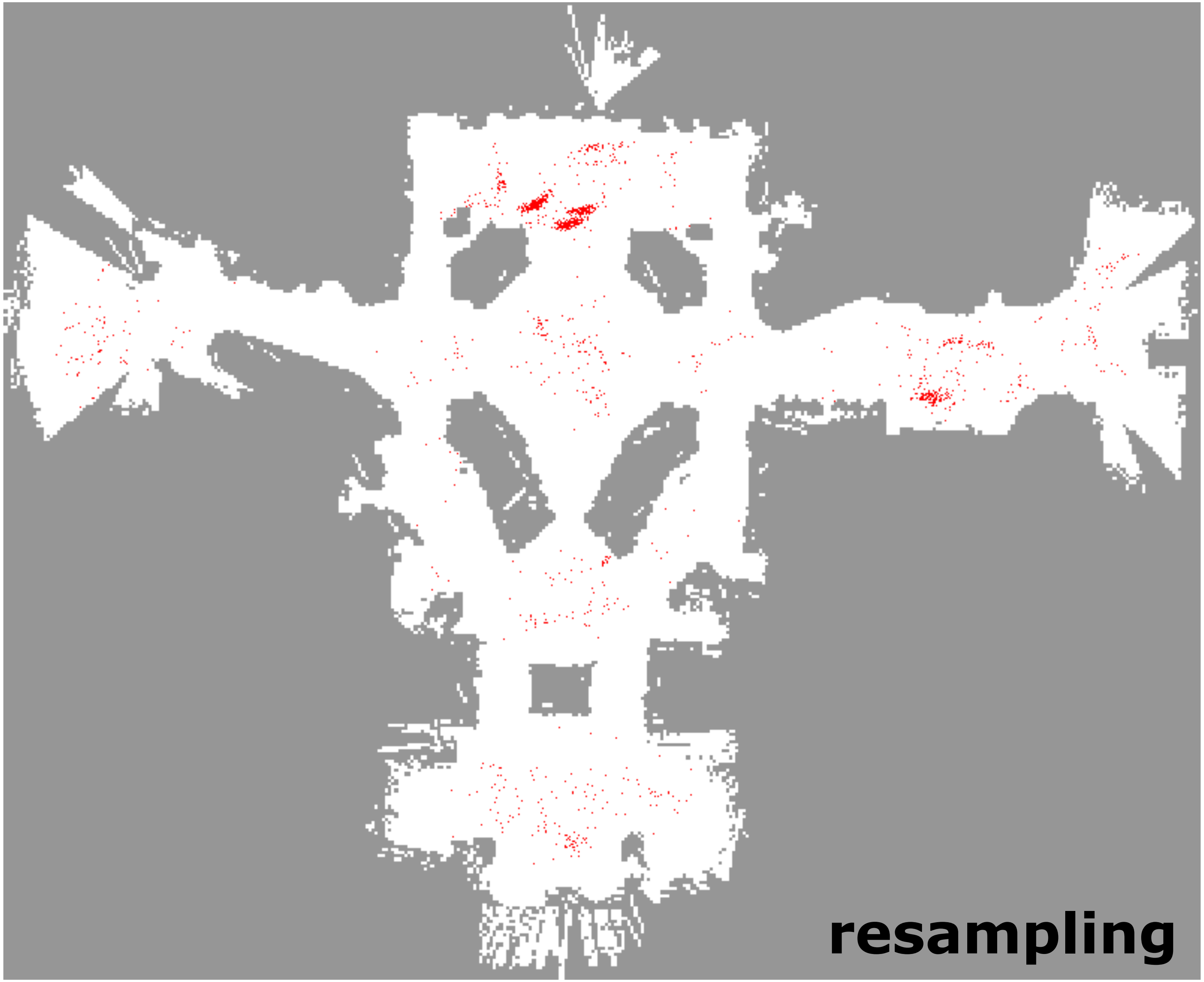

observation

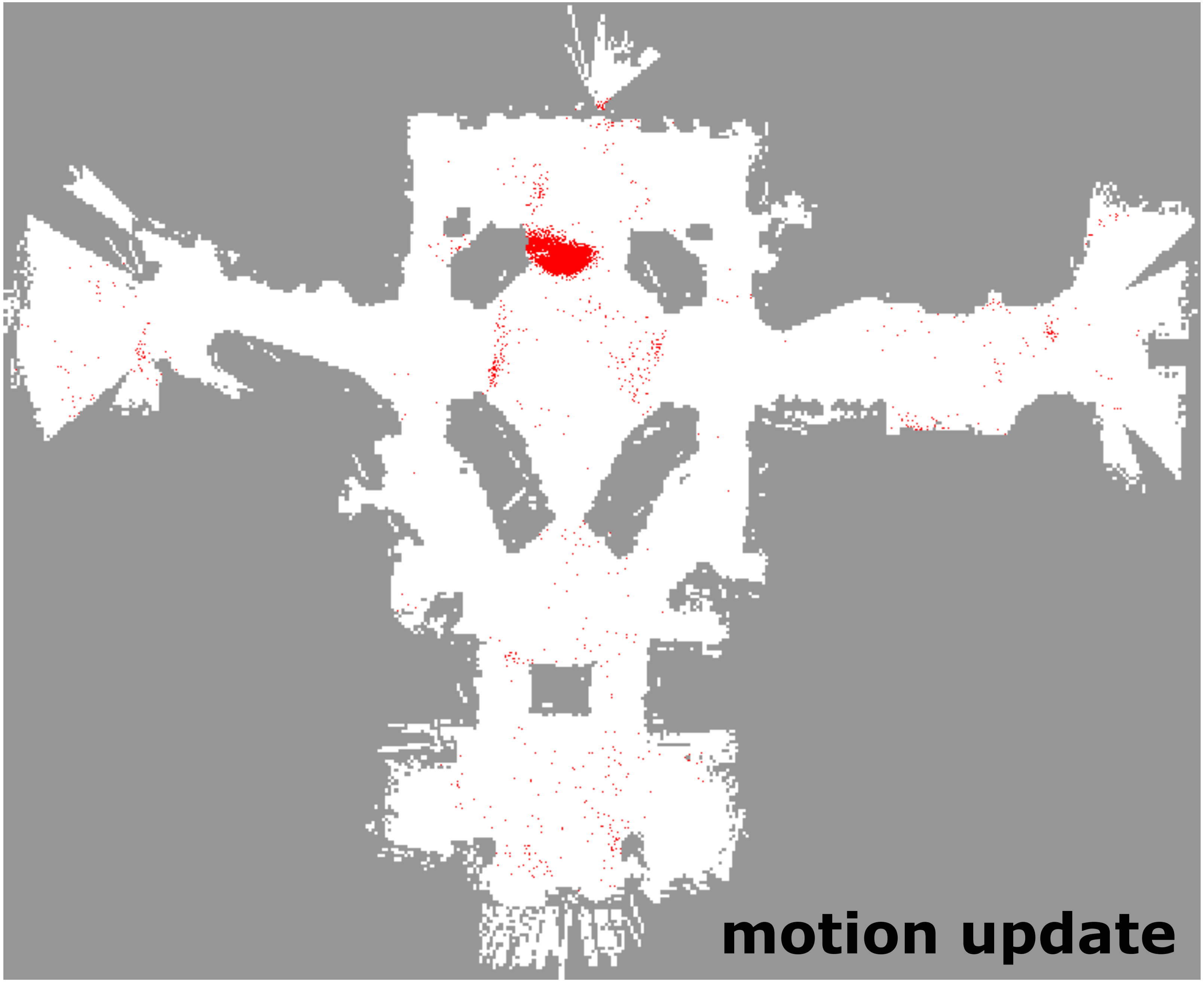

resampling

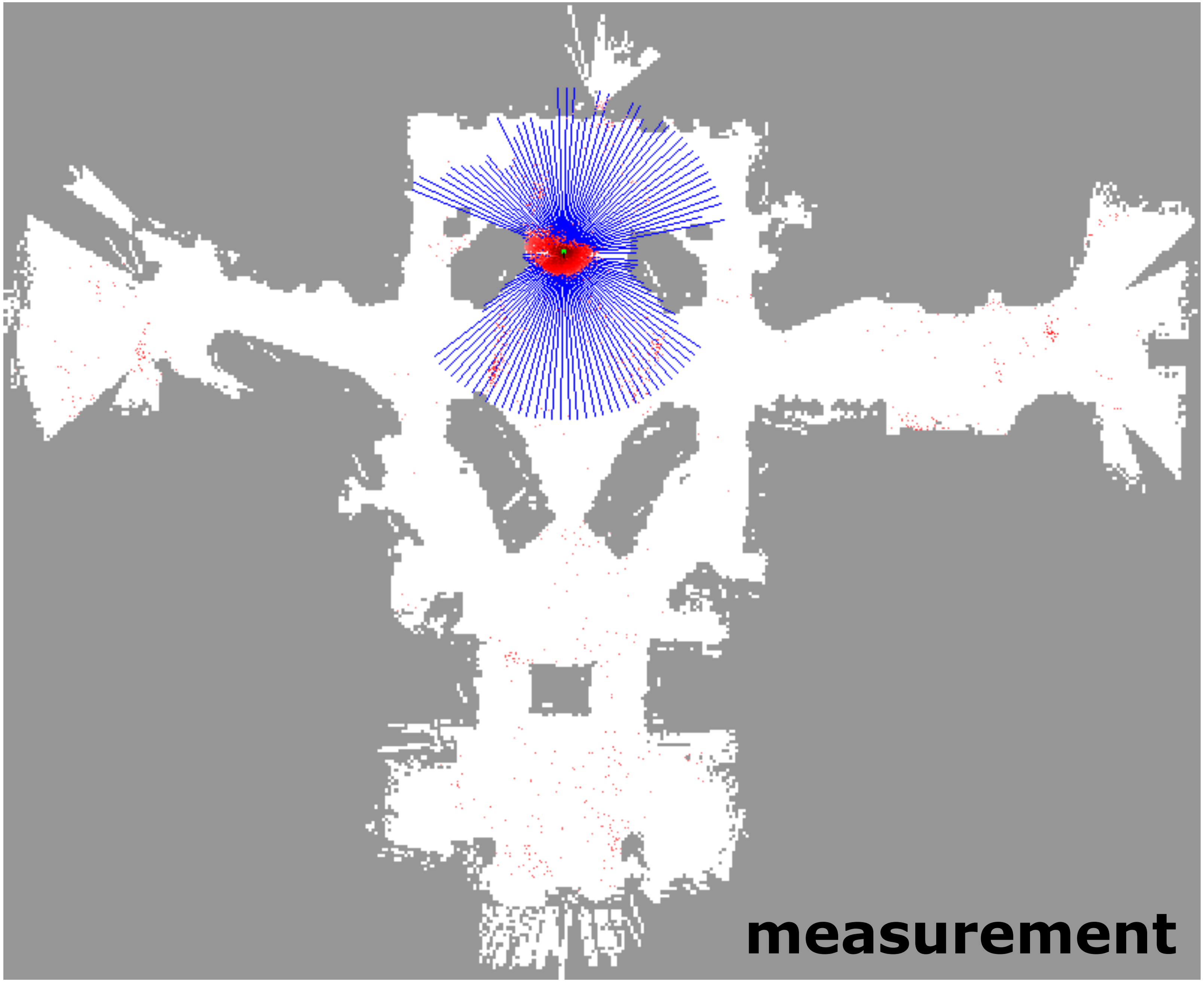

motion update


measurement

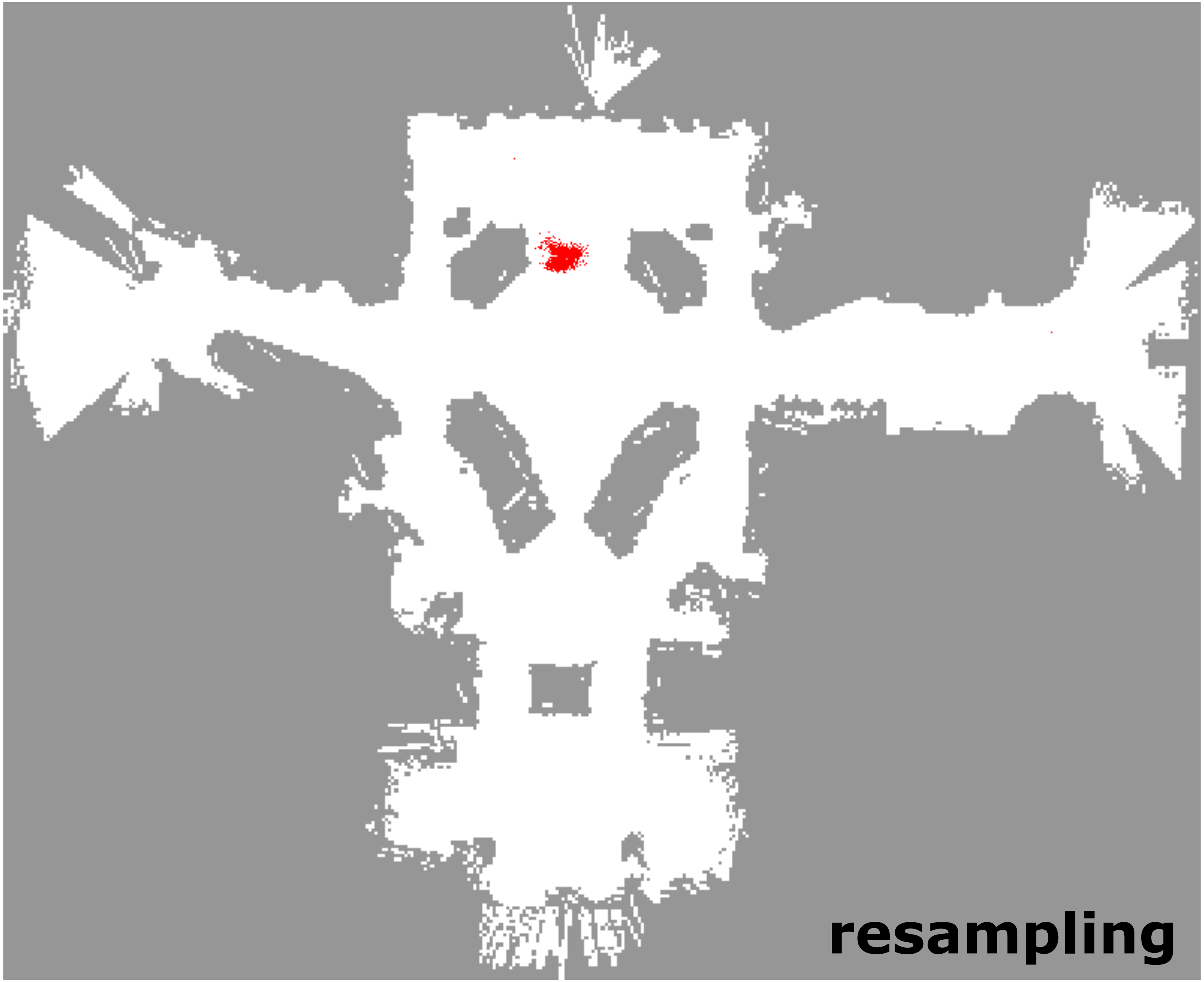

weight update

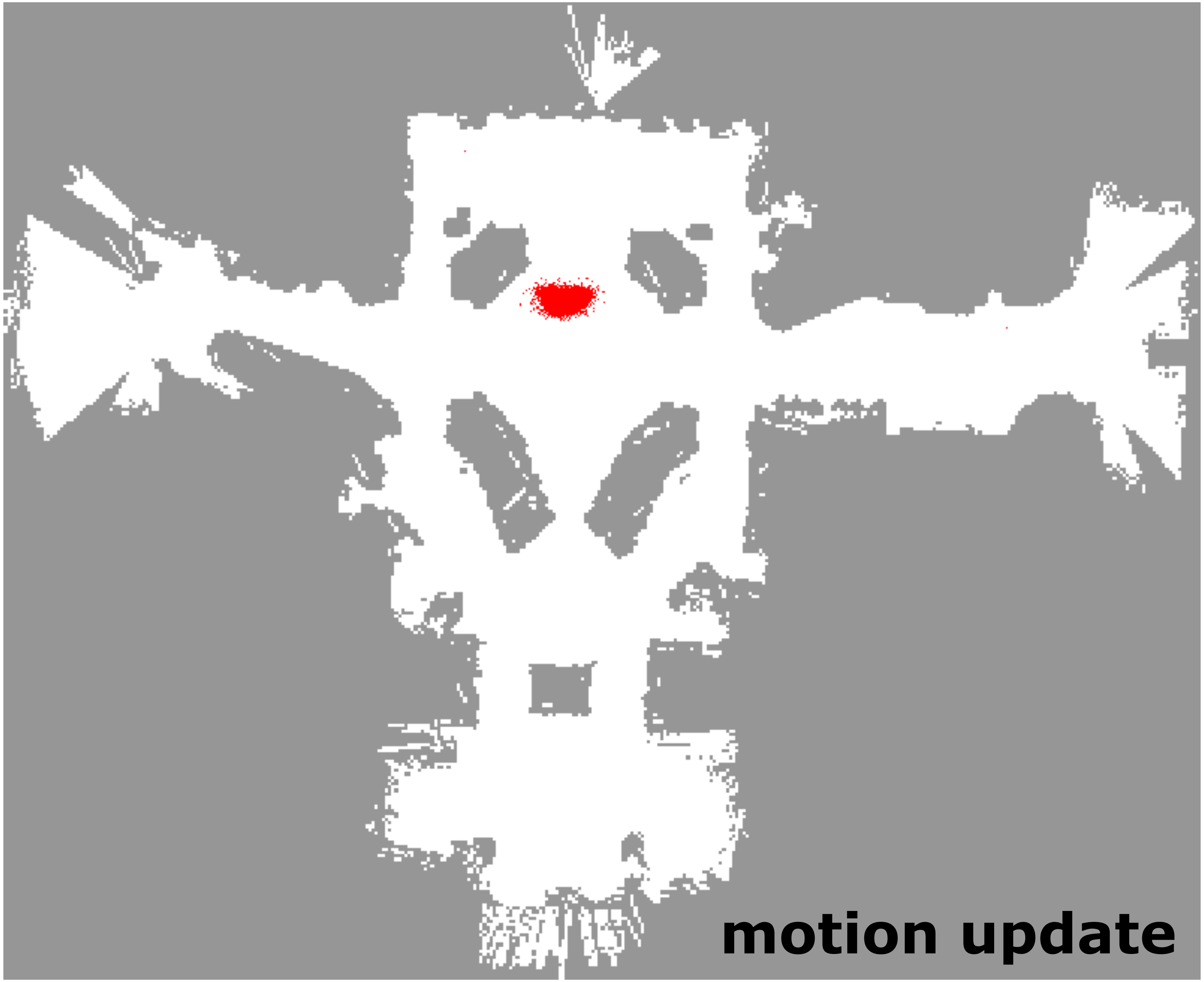

resampling

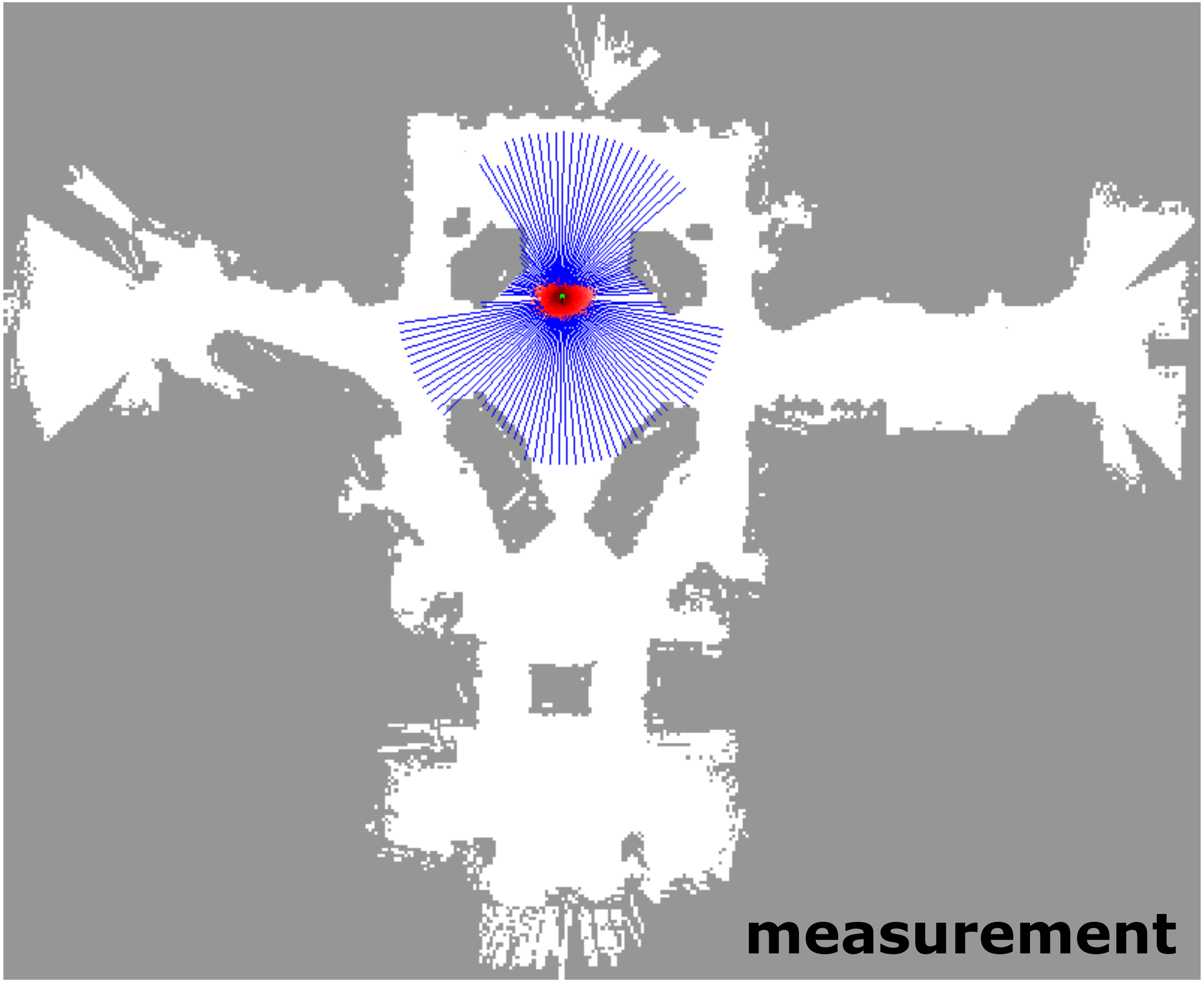

motion update

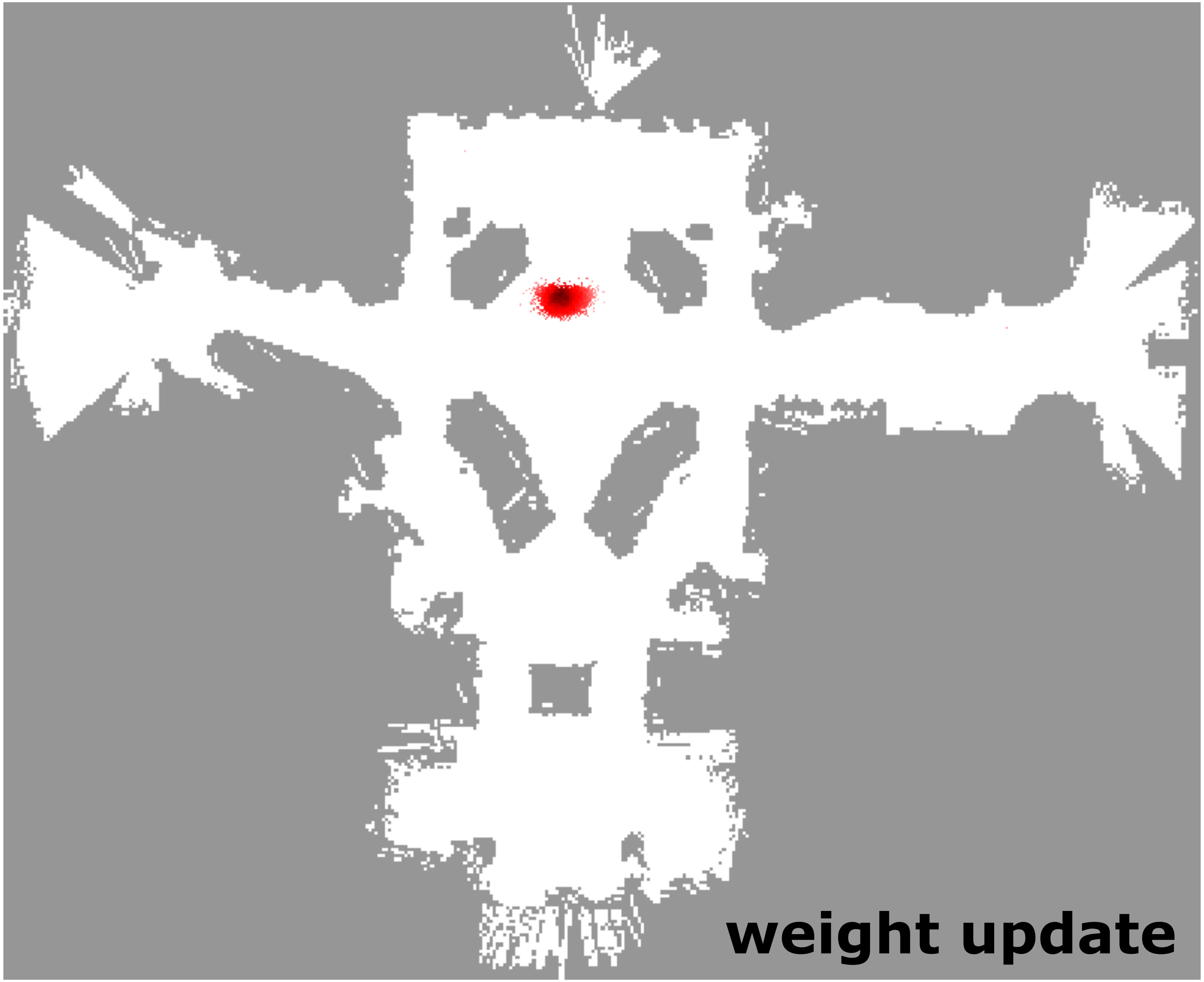

measurement

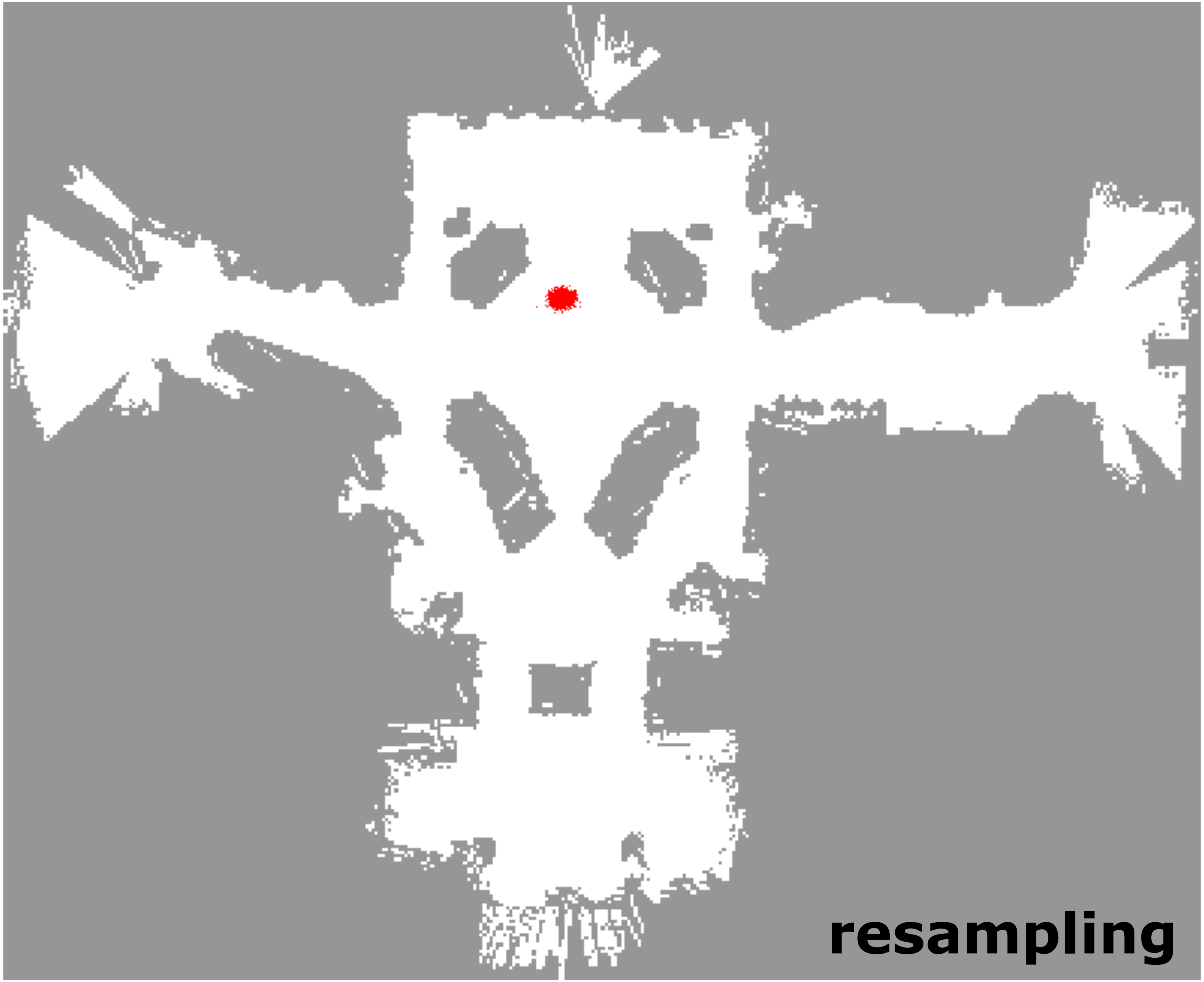

weight update

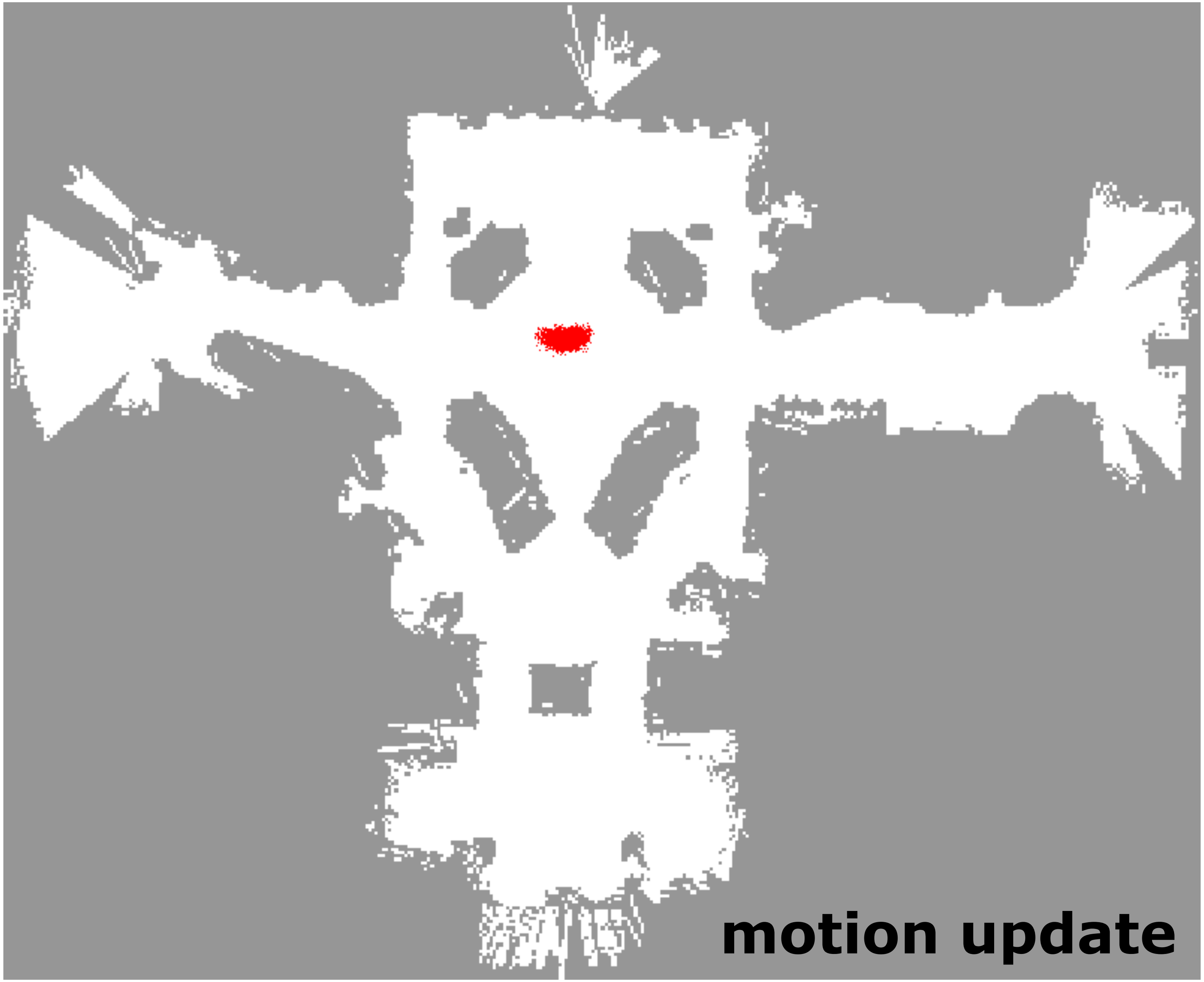

resampling

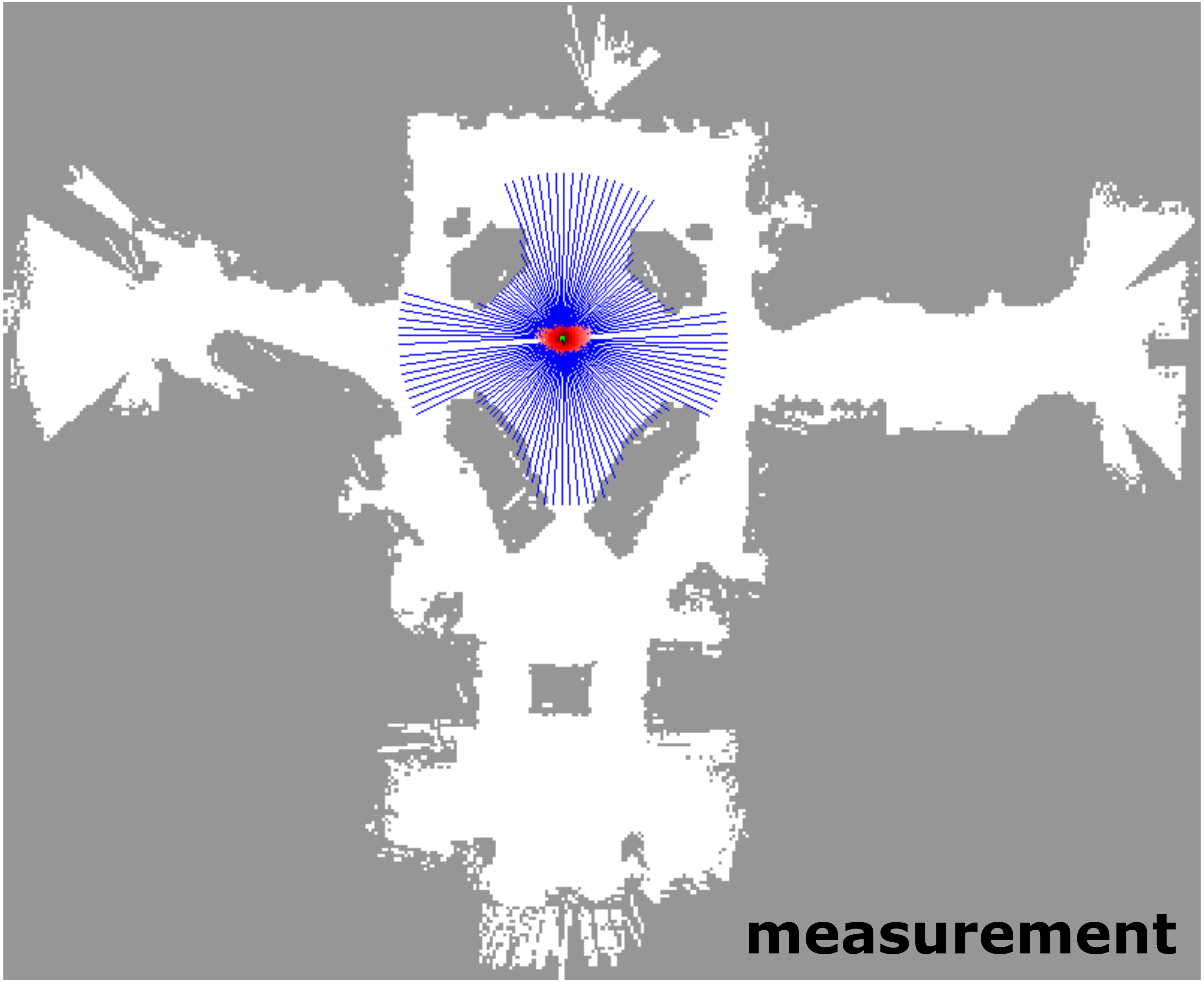

motion update


measurement


weight update


resampling


motion update


measurement

Summary – Particle Filters

- Particle filters are non-parametric, recursive Bayes filters
- Posterior is represented by a set of weighted samples
- Not limited to Gaussians
- Proposal to draw new samples
- Weight to account for the differences between the proposal and the target
- Work well in low-dimensional spaces

Summary – PF Localization

- Particles are propagated according to the motion model
- They are weighted according to the likelihood of the observation
- Called: Monte-Carlo localization (MCL)
- MCL is the gold standard for mobile robot localization today

Literature

On Monte Carlo Localization

- Thrun et al. “Probabilistic Robotics”, Chapter 8.3

On the particle filter

- Thrun et al. “Probabilistic Robotics”, Chapter 3

On motion and observation models

- Thrun et al. “Probabilistic Robotics”, Chapters 5 & 6

Key Questions

- What does our map look like?
- What kind of sensor do we use?
- How to obtain a sensor model?
- How to describe the motion?