Probabilistic Robotics*

Non-parametric Bayes Filter Implementations

Particle filters

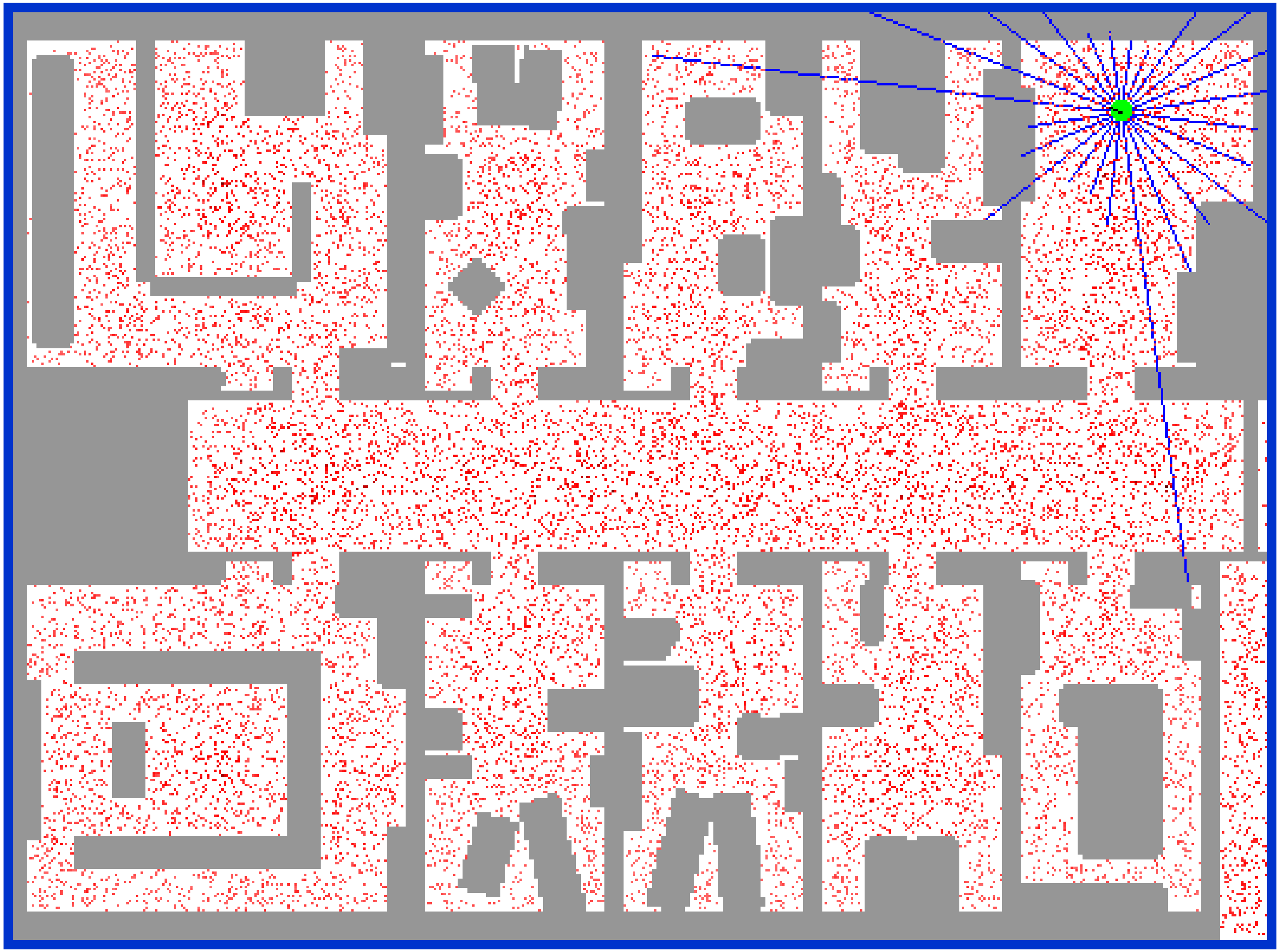
Mohan Sridharan

University of Birmingham, UK

m.sridharan@bham.ac.uk

*Revised original slides that accompany the book by Thrun, Burgard and Fox.

Sample-based Localization (sonar)



Particle Filters

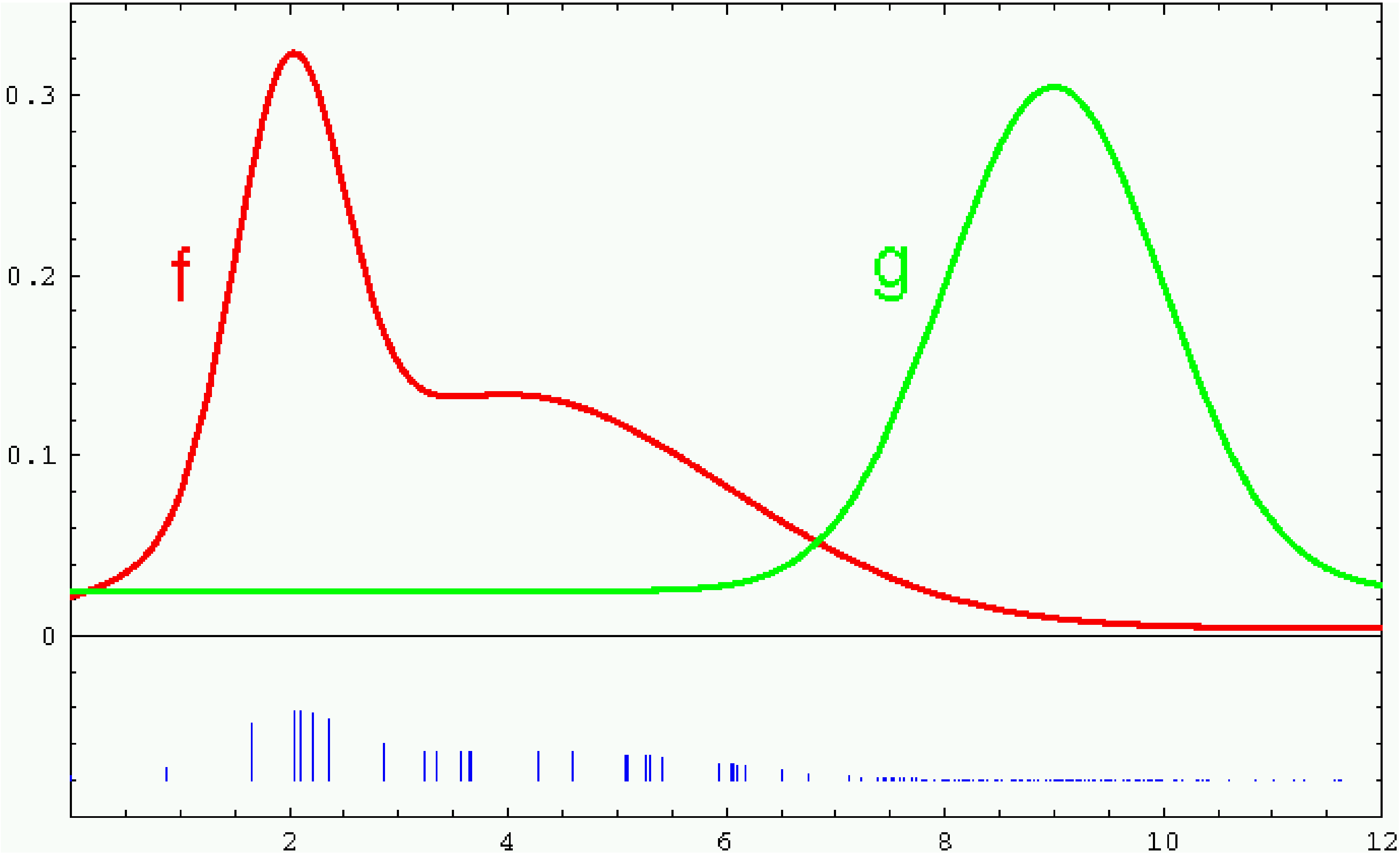
- Represent belief by random samples.
- Estimation of non-Gaussian, nonlinear processes.
- Monte Carlo filter, survival of the fittest, I-condensation, bootstrap filter, particle filter.
- Filtering: [Rubin, 88], [Gordon et al., 93], [Kitagawa 96].
- Computer vision: [Isard and Blake 96, 98].
- Dynamic Bayesian Networks: [Kanazawa et al., 95].

Particle Filter Algorithm (basic)

Algorithm **particle_filter**(χ_{t-1}, u_t, z_t):

1. $\bar{\chi}_t = \chi_t = \emptyset$
2. **For** m = 1… M
3. Sample $x_t^{[m]} \sim p(x_t \mid x_{t-1}^{[m]}, u_t)$
4. $w_t^{[m]} = p(z_t \mid x_t^{[m]})$
5. $\bar{\chi}_t = \bar{\chi}_t + \left\langle x_t^{[m]}, w_t^{[m]} \right\rangle$
6. **For** m=1…M
7. Draw i with probability $\propto w_t^{[i]}$
8. Add $x_t^{[i]}$ to χ_t
9. **Return**

Importance Sampling



Weight samples: $w = f / g$

Importance Sampling

$$f(.) = bel(x_t) = \eta\, p(z \mid x)\overline{bel}(x_t)$$

$$g(.) = \overline{bel}(x_t) = \sum p(x_t \mid u_t, x_{t-1})\, bel(x_{t-1})$$

Function $f(.)$: target distribution.

Function $g(.)$: proposal distribution.

Weights: $w(x) = f(x) / g(x)$

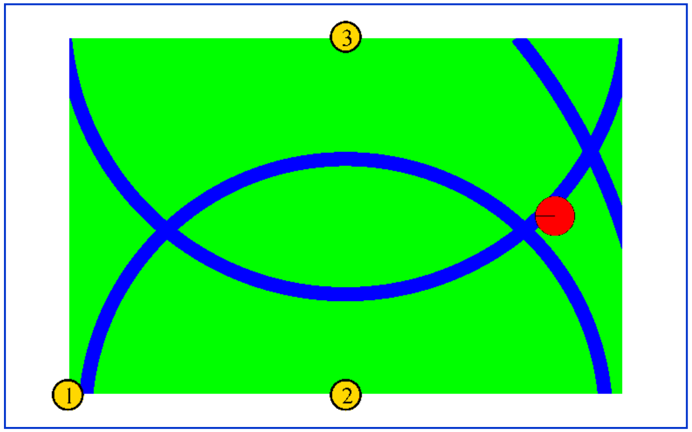
Need: $f(x) > 0 \rightarrow g(x) > 0$

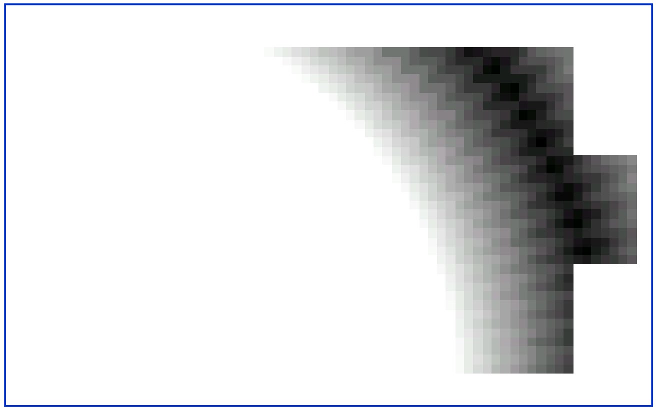
Converges to desired distribution iteratively.

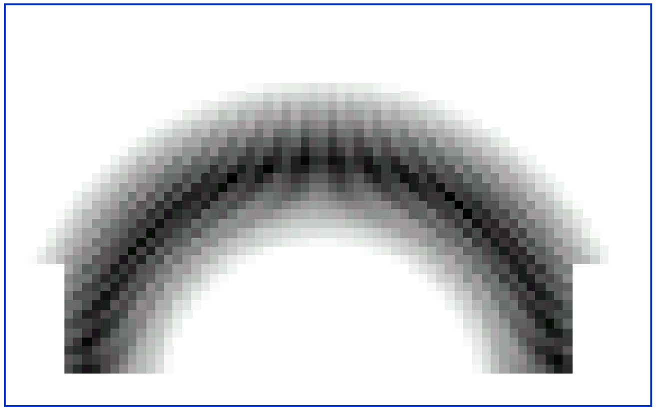
PF derivation (Section 4.3.3, PR)

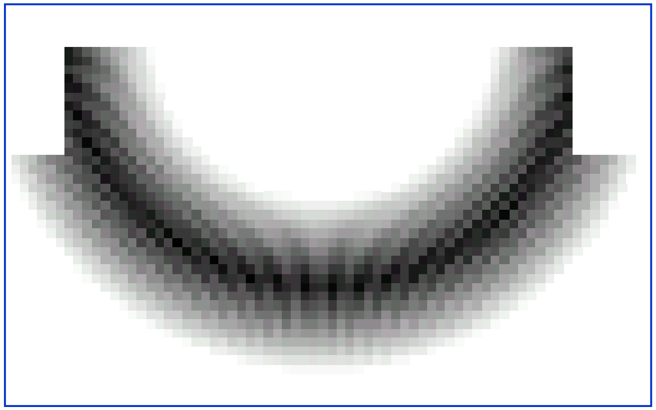
Importance Sampling with Resampling: Landmark Detection Example

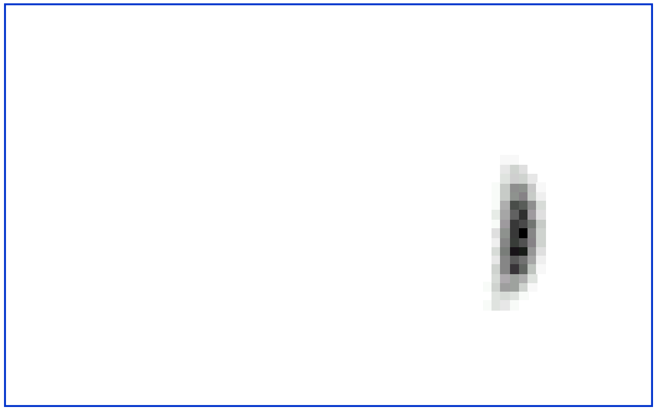
Distributions



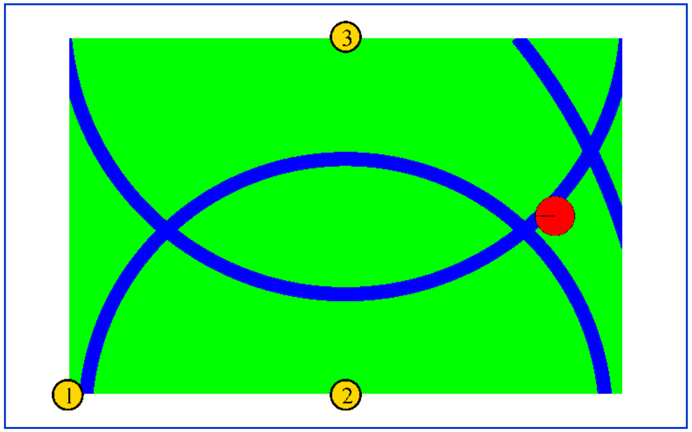


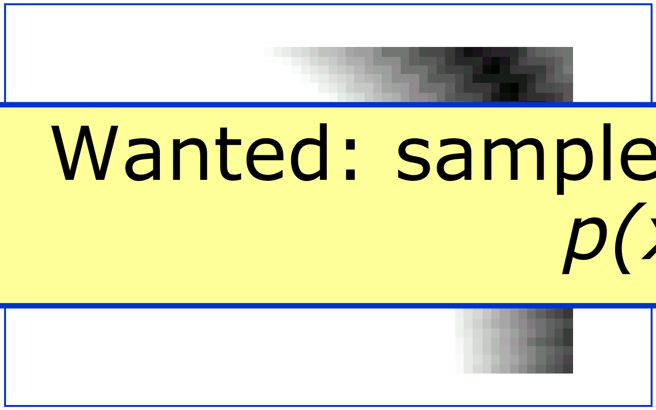


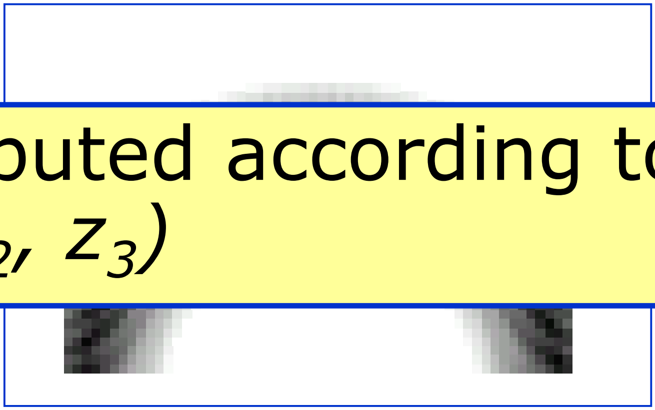




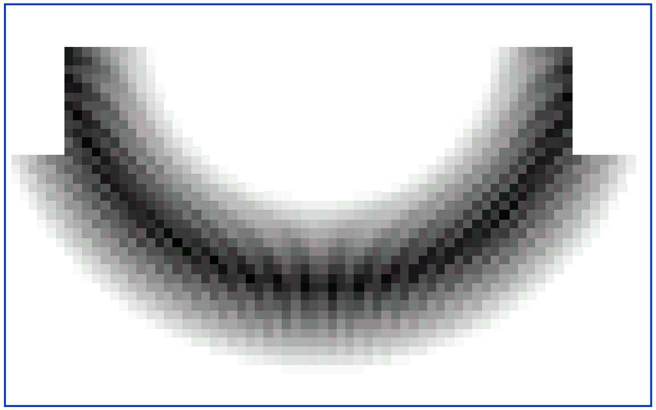
Distributions

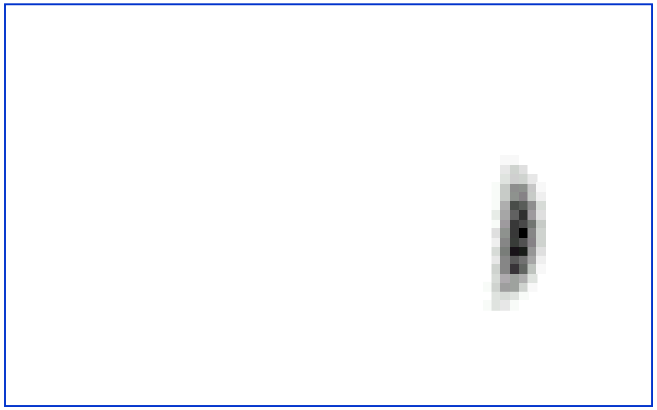






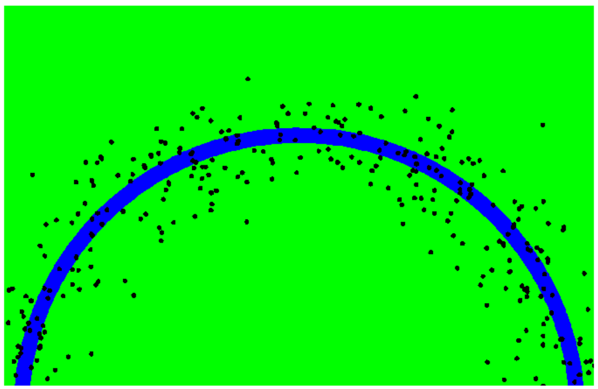
Wanted: samples distributed according to
$p(x|\ z_1, z_2, z_3)$

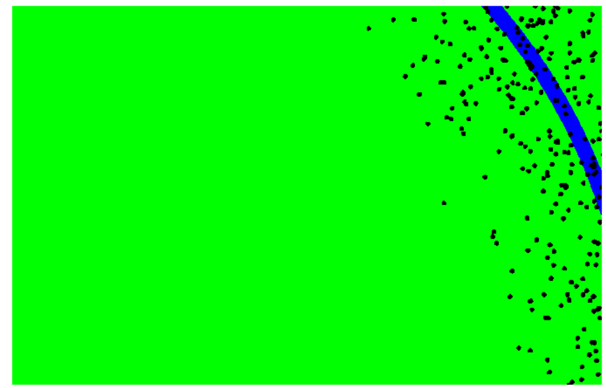


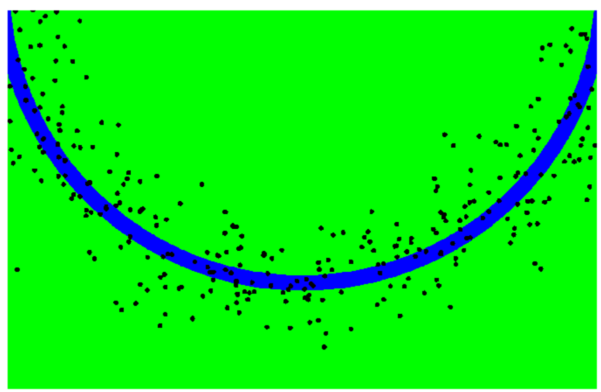


This is Easy!

We can draw samples from $p(x|z_l)$ by adding noise to the detection parameters.







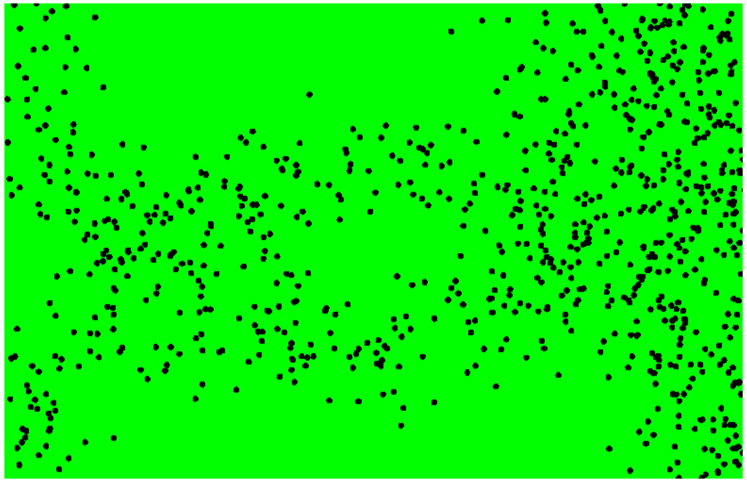
Importance Sampling with Resampling

$$\text{Target distribution f}: p(x \mid z_1, z_2, \ldots, z_n) = \frac{\prod_k p(z_k \mid x) \quad p(x)}{p(z_1, z_2, \ldots, z_n)}$$

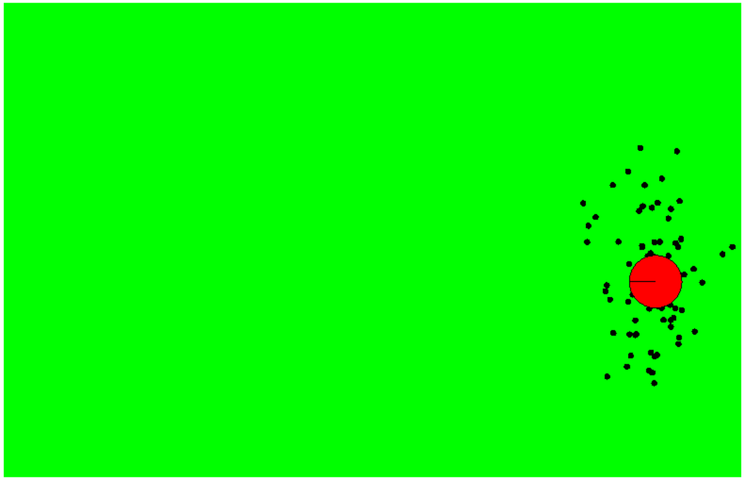
$$\text{Sampling distribution g}: p(x \mid z_l) = \frac{p(z_l \mid x)\, p(x)}{p(z_l)}$$

$$\text{Importance weights w}: \frac{f}{g} = \frac{p(x \mid z_1, z_2, \ldots, z_n)}{p(x \mid z_l)} = \frac{p(z_l) \prod_{k \neq l} p(z_k \mid x)}{p(z_1, z_2, \ldots, z_n)}$$

Importance Sampling with Resampling

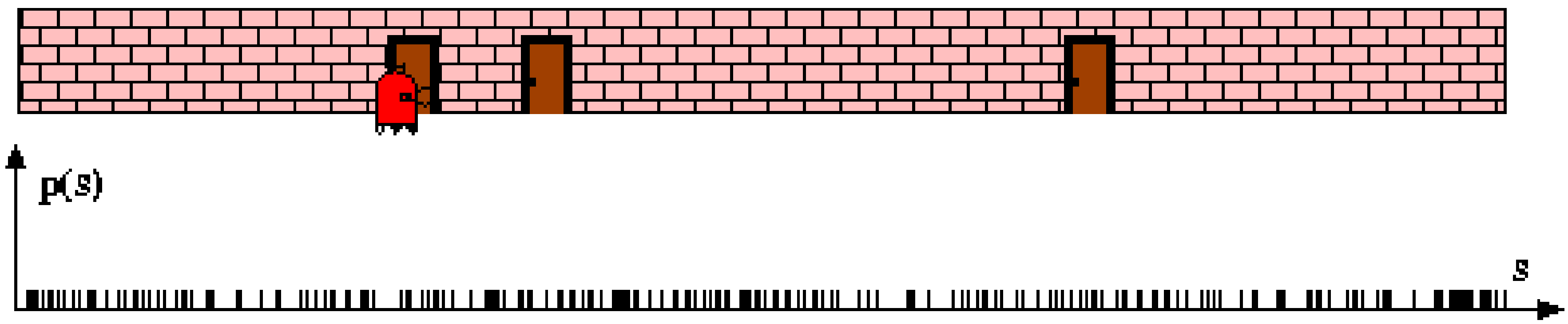


Weighted samples



After resampling

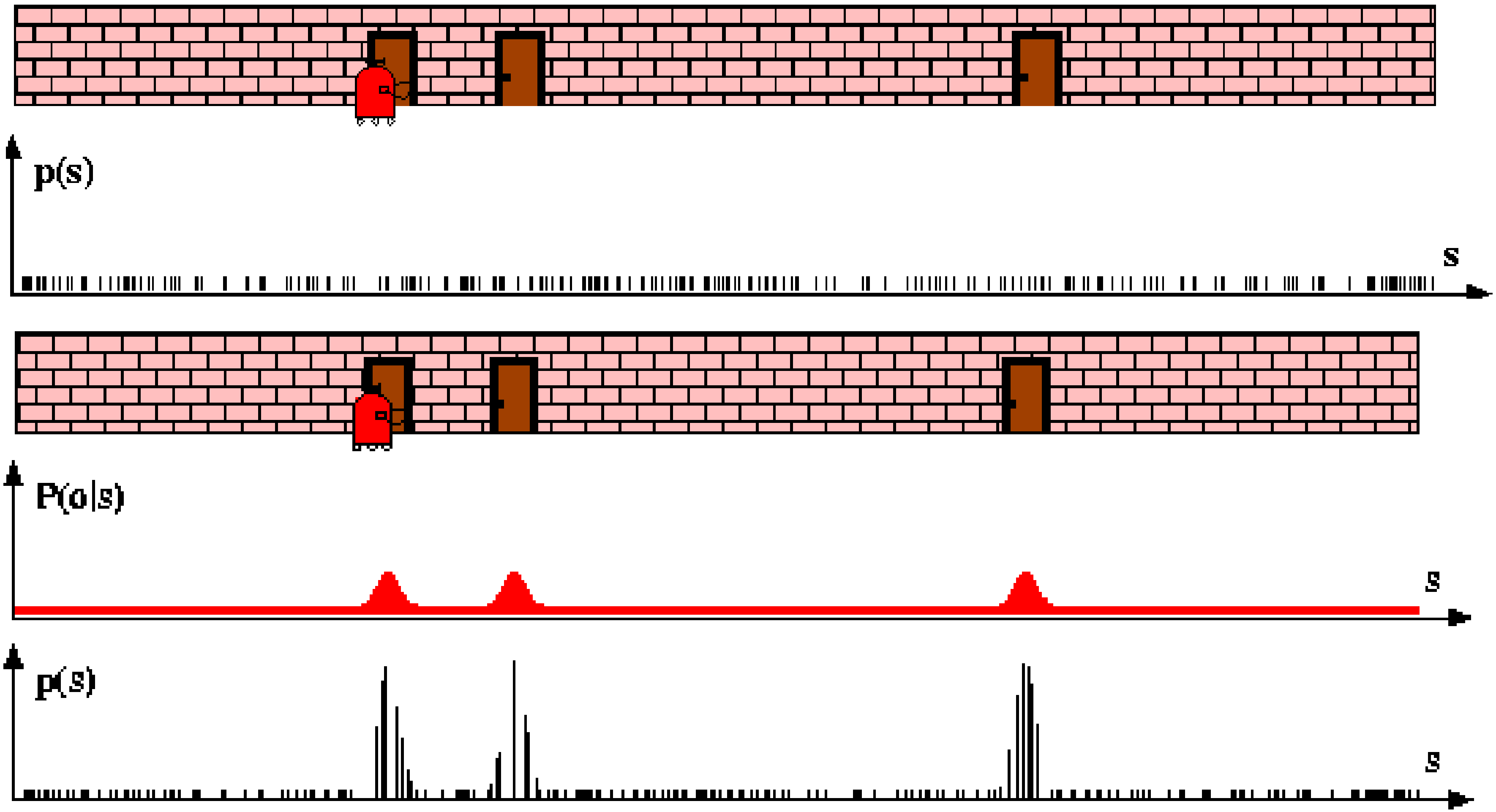
Particle Filters



Sensor Information: Importance Sampling

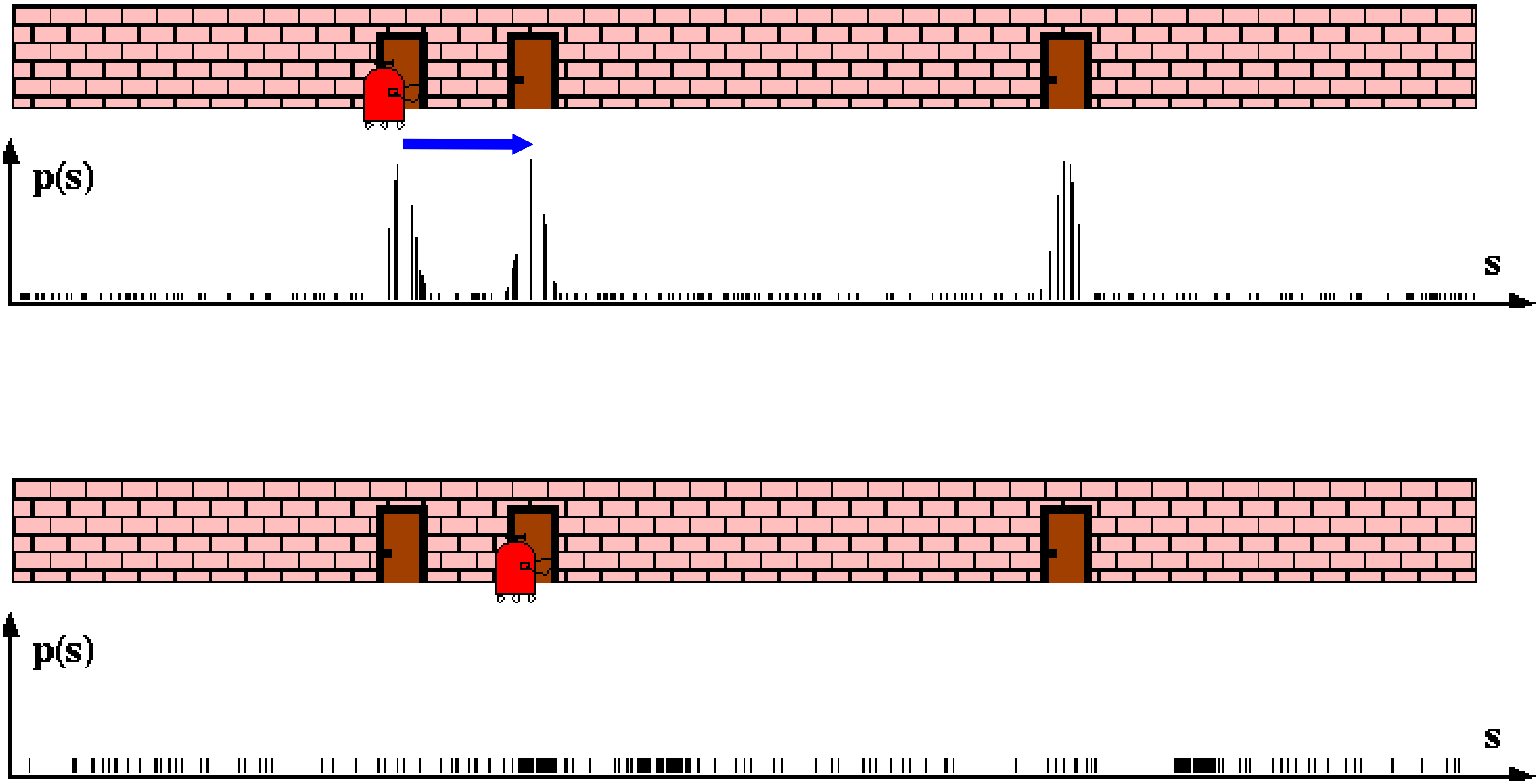
$$Bel(x) \leftarrow \alpha\ p(z \mid x)\ Bel^{-}(x)$$

$$w \leftarrow \frac{\alpha\ p(z \mid x)\ Bel^{-}(x)}{Bel^{-}(x)} = \alpha\ p(z \mid x)$$



Robot Motion

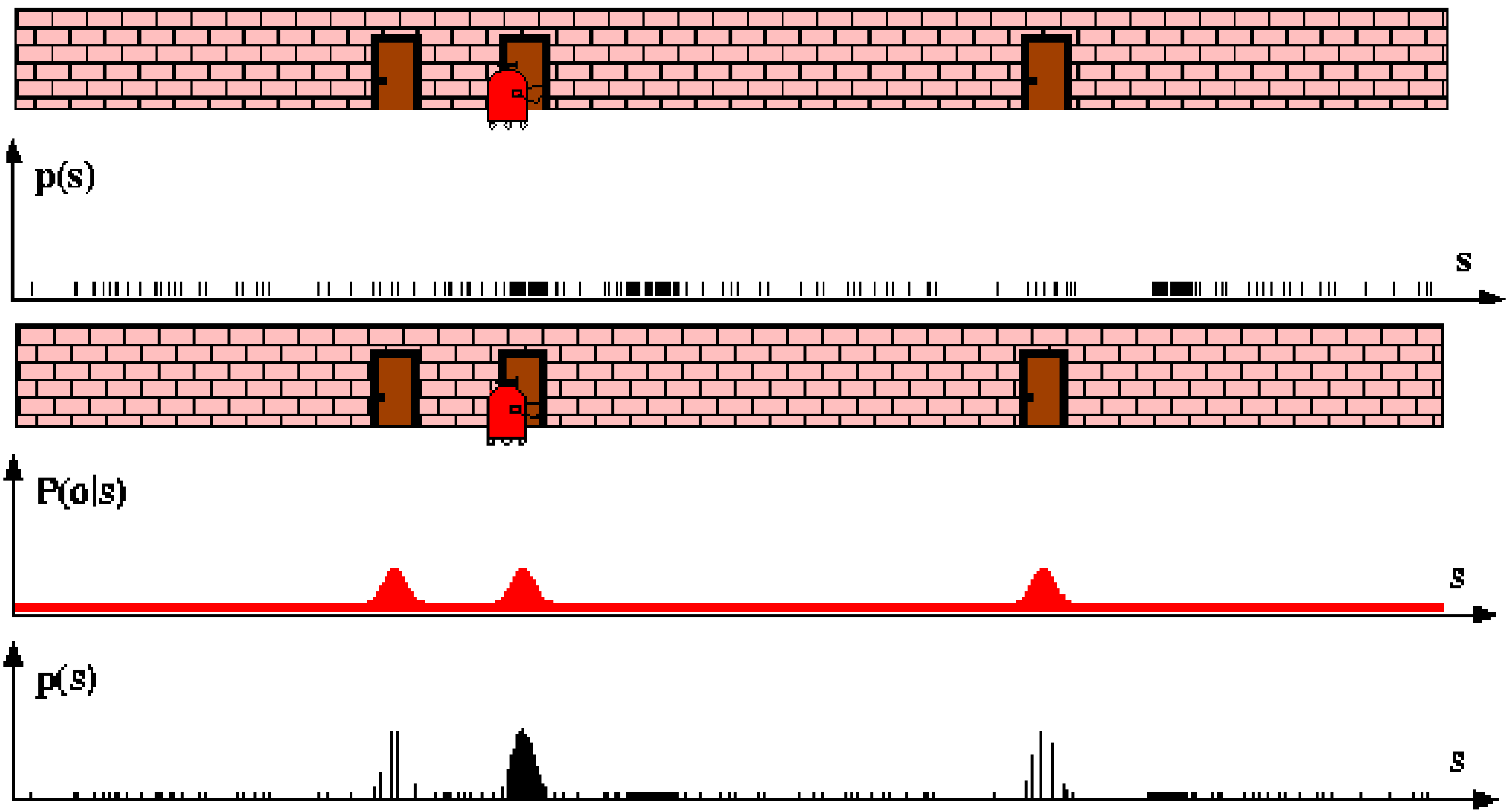
$$Bel^{-}(x) \leftarrow \int p(x \mid u, x')\, Bel(x')\, \mathrm{d}x'$$



Sensor Information: Importance Sampling

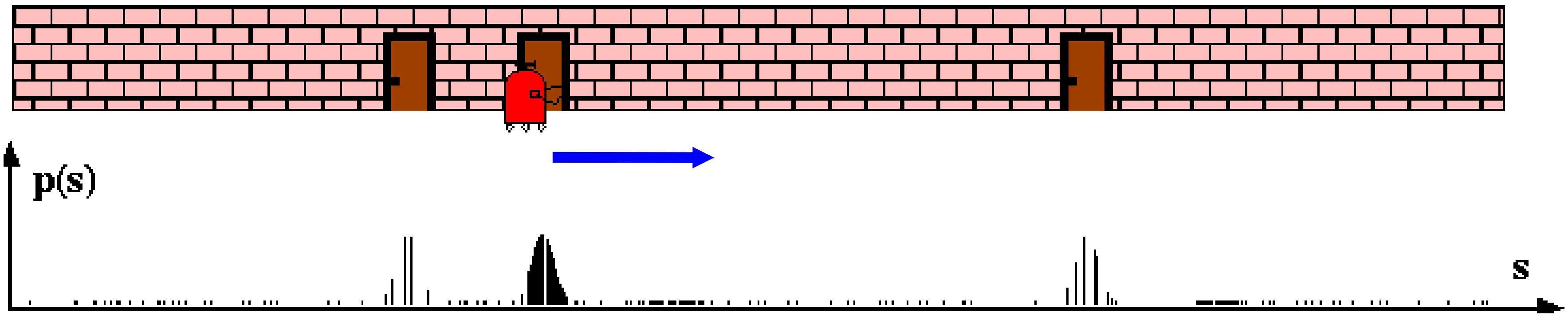
$$Bel(x) \leftarrow \alpha\ p(z \mid x)\, Bel^{-}(x)$$

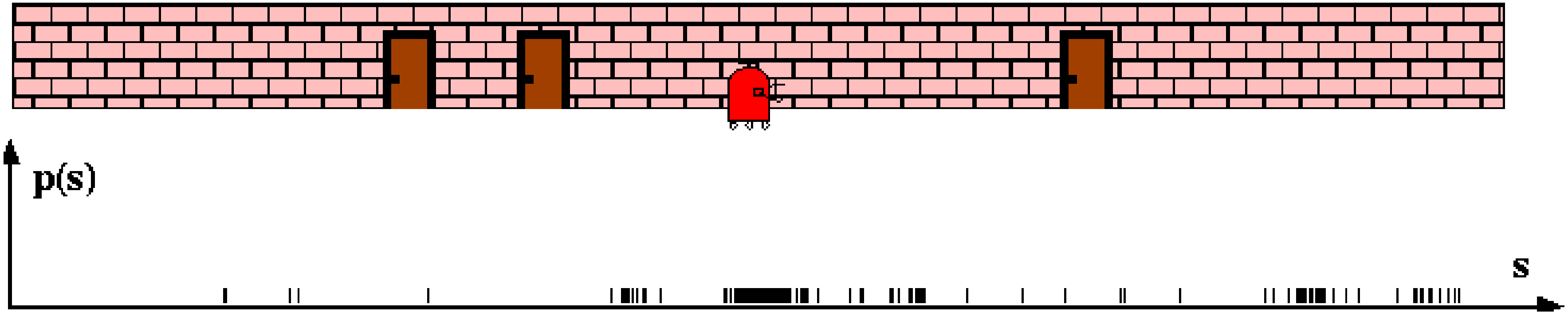
$$w \leftarrow \frac{\alpha\ p(z \mid x)\, Bel^{-}(x)}{Bel^{-}(x)} = \alpha\ p(z \mid x)$$



Robot Motion

$$Bel^{-}(x) \leftarrow \int p(x \mid u, x')\, Bel(x')\ \mathrm{d}x'$$





Particle Filter Algorithm

$$Bel\,(x_t) = \eta\ p(z_t \mid x_t) \int p(x_t \mid x_{t-1}, u_t)\ Bel\,(x_{t-1})\ dx_{t-1}$$

draw x^i_{t-1} from $Bel(x_{t-1})$

draw x^i_t from $p(x_t \mid x^i_{t-1}, u_t)$

Importance factor for x^i_t:

$$w^i_t = \frac{\text{target distribution}}{\text{proposal distribution}}$$

$$= \frac{\eta\ p(z_t \mid x_t)\ p(x_t \mid x_{t-1}, u_t)\ Bel\,(x_{t-1})}{p(x_t \mid x_{t-1}, u_t)\ Bel\,(x_{t-1})}$$

$$\propto p(z_t \mid x_t)$$

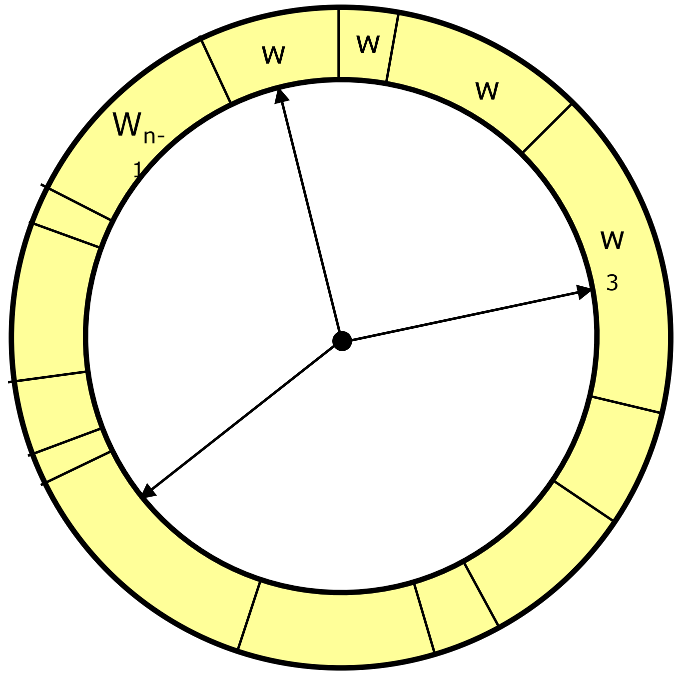
Resampling

- **Given**: Set S of weighted samples.

- **Wanted** : Random sample, where the probability of drawing x_i is given by w_i.

- Typically done M times with replacement to generate new sample set S'.

Resampling



- Roulette wheel
- Binary search, n log n

- Stochastic universal sampling
- Systematic resampling
- Linear time complexity
- Easy to implement, low variance

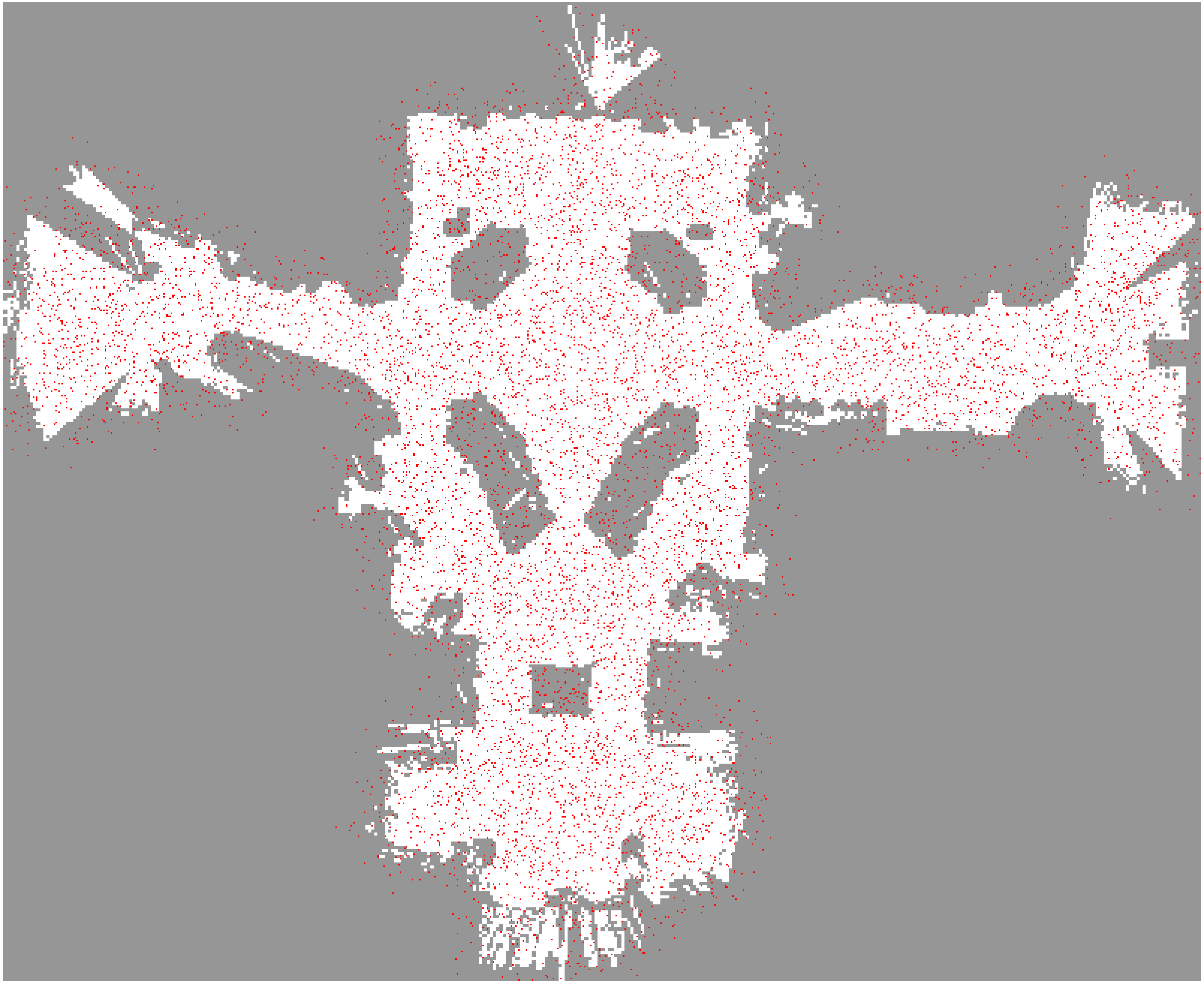
Resampling Algorithm

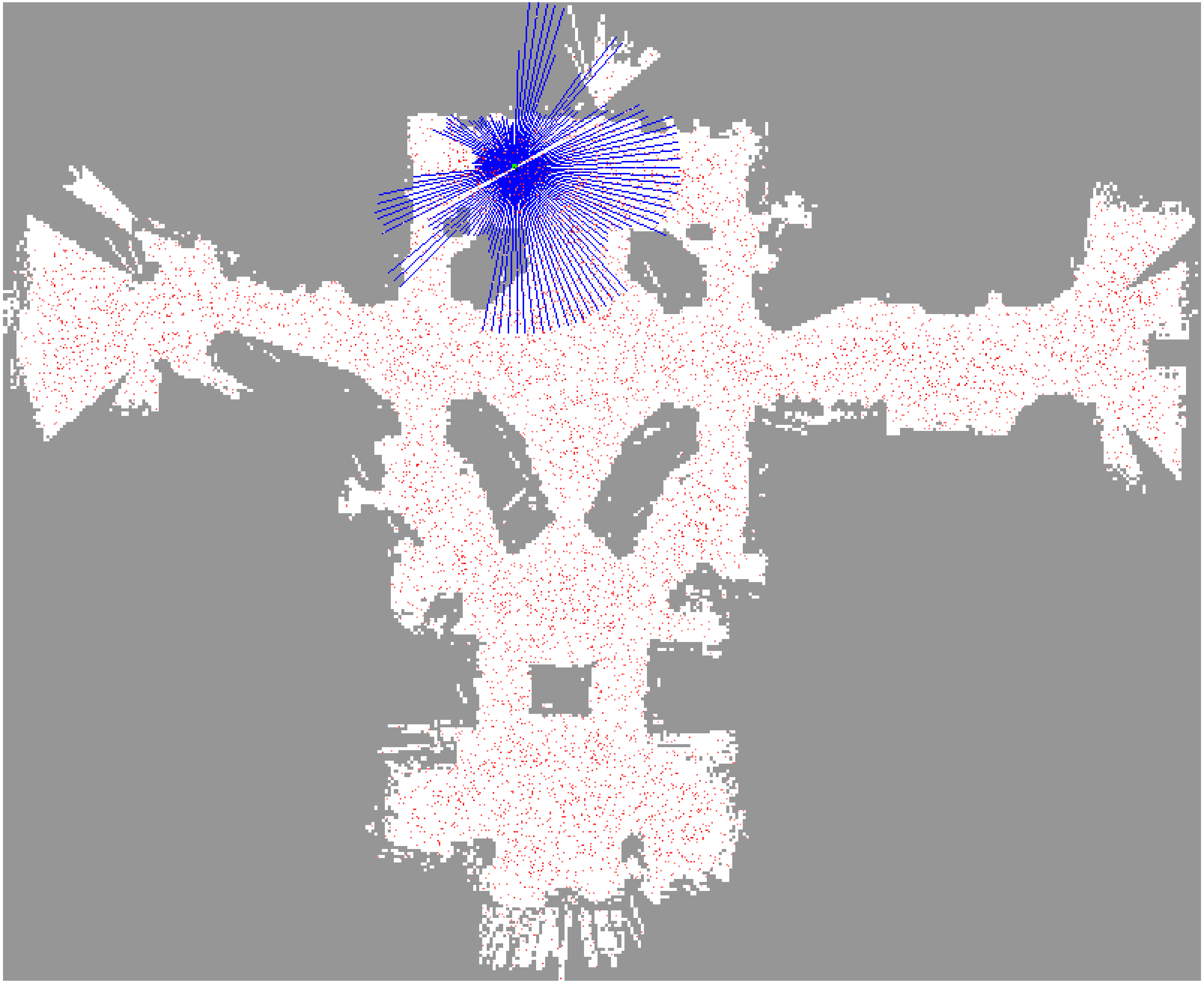
1. Algorithm **systematic_resampling**(*S*, *M*):

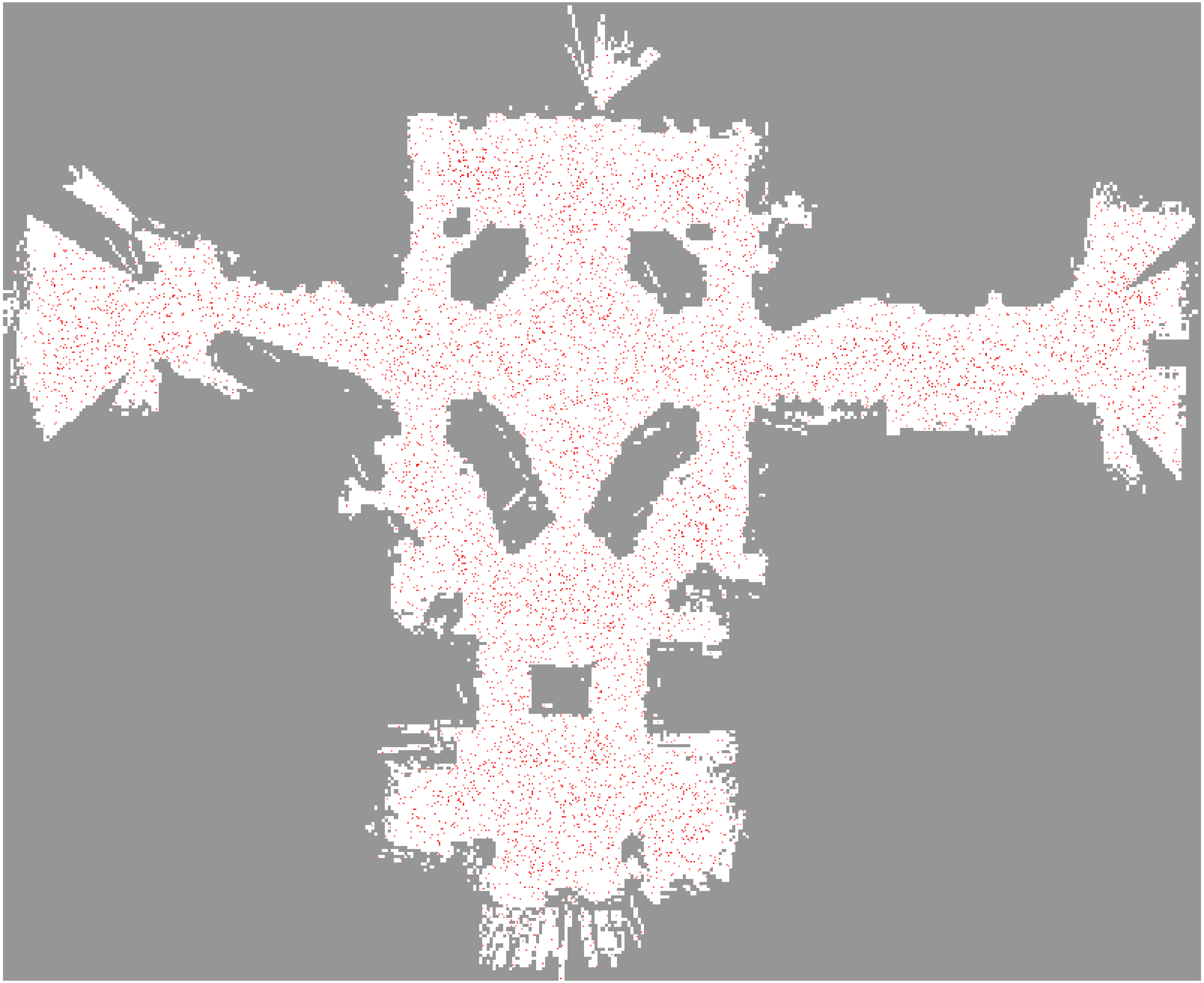
1. $S' = \emptyset, c_1 = w^1$
2. **For** $i = 2 \ldots M$ — ***Generate cdf***
3. $\quad c_i = c_{i-1} + w^i$
4. $u_1 \sim U(0, M^{-1}], i = 1$ — ***Initialize threshold***

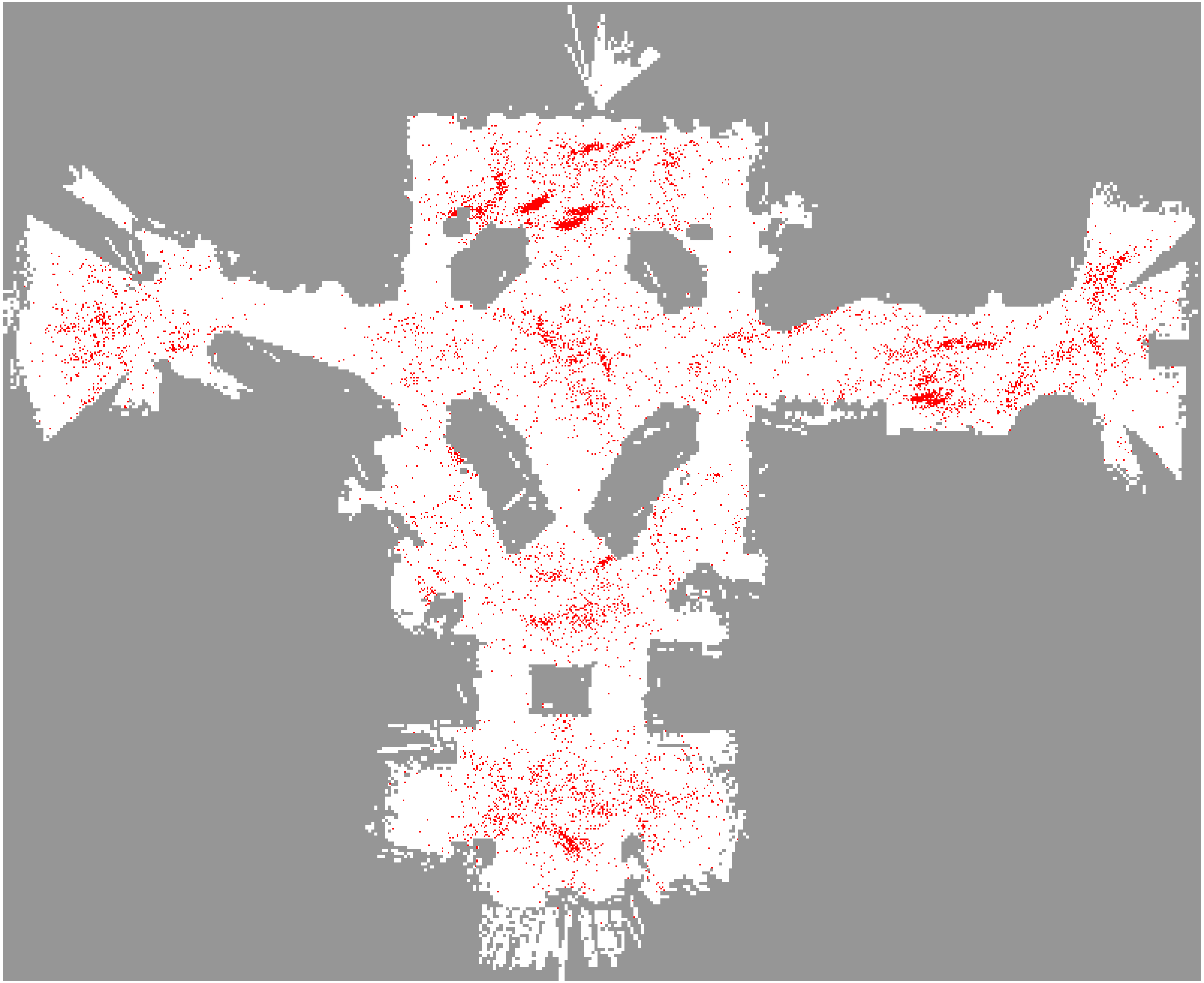
1. **For** $j = 1 \ldots M$ — ***Draw samples …***
2. $\quad$ **While** $(u_j > c_i)$ — ***Skip until next threshold reached***
3. $\qquad i = i + 1$
4. $\quad S' = S' \cup \left\{ < x^i, M^{-1} > \right\}$ — ***Insert***
5. $\quad u_{j+1} = u_j + M^{-1}$ — ***Increment threshold***
6. **Return** *S’*

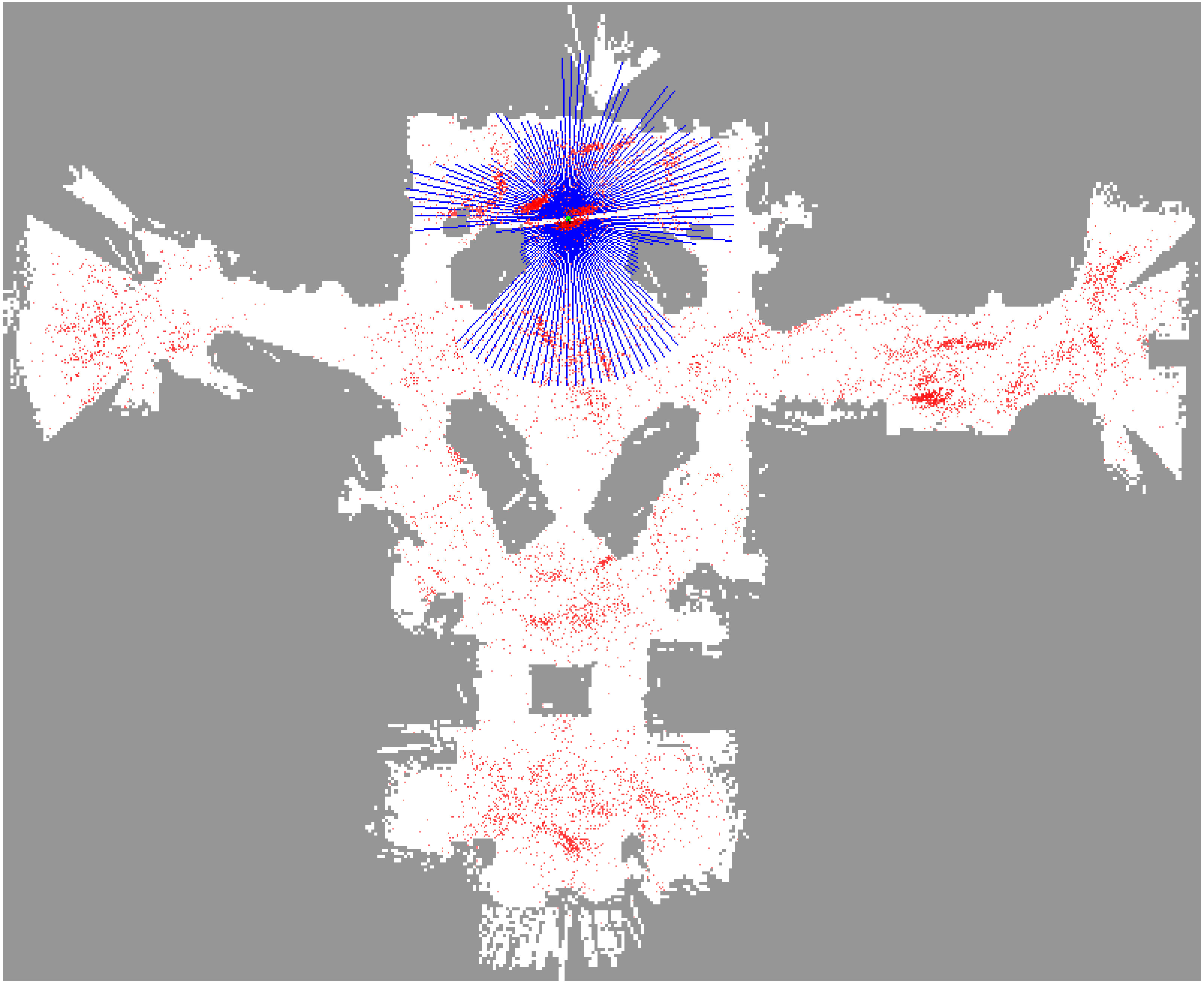
Also called **stochastic universal sampling**

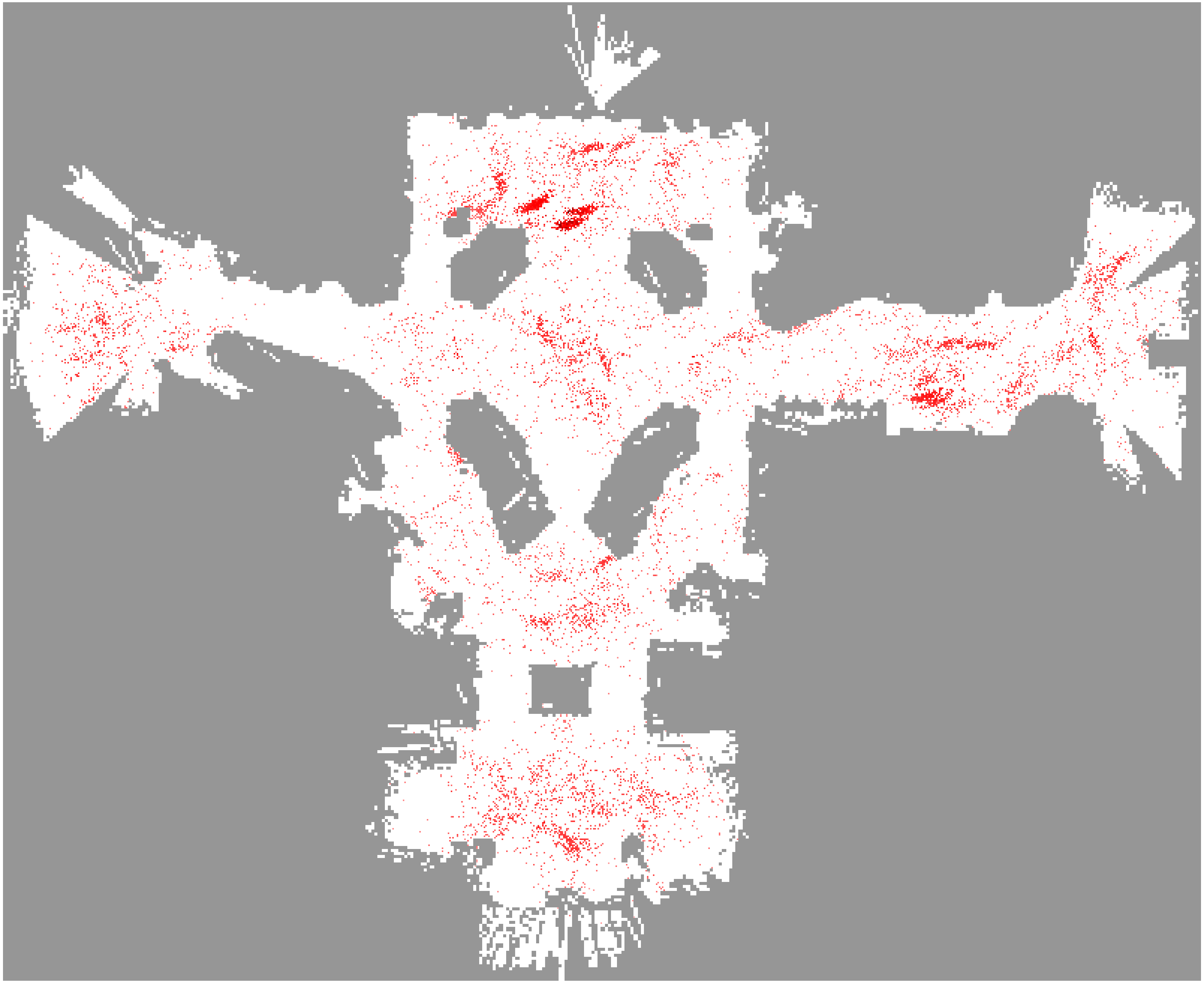


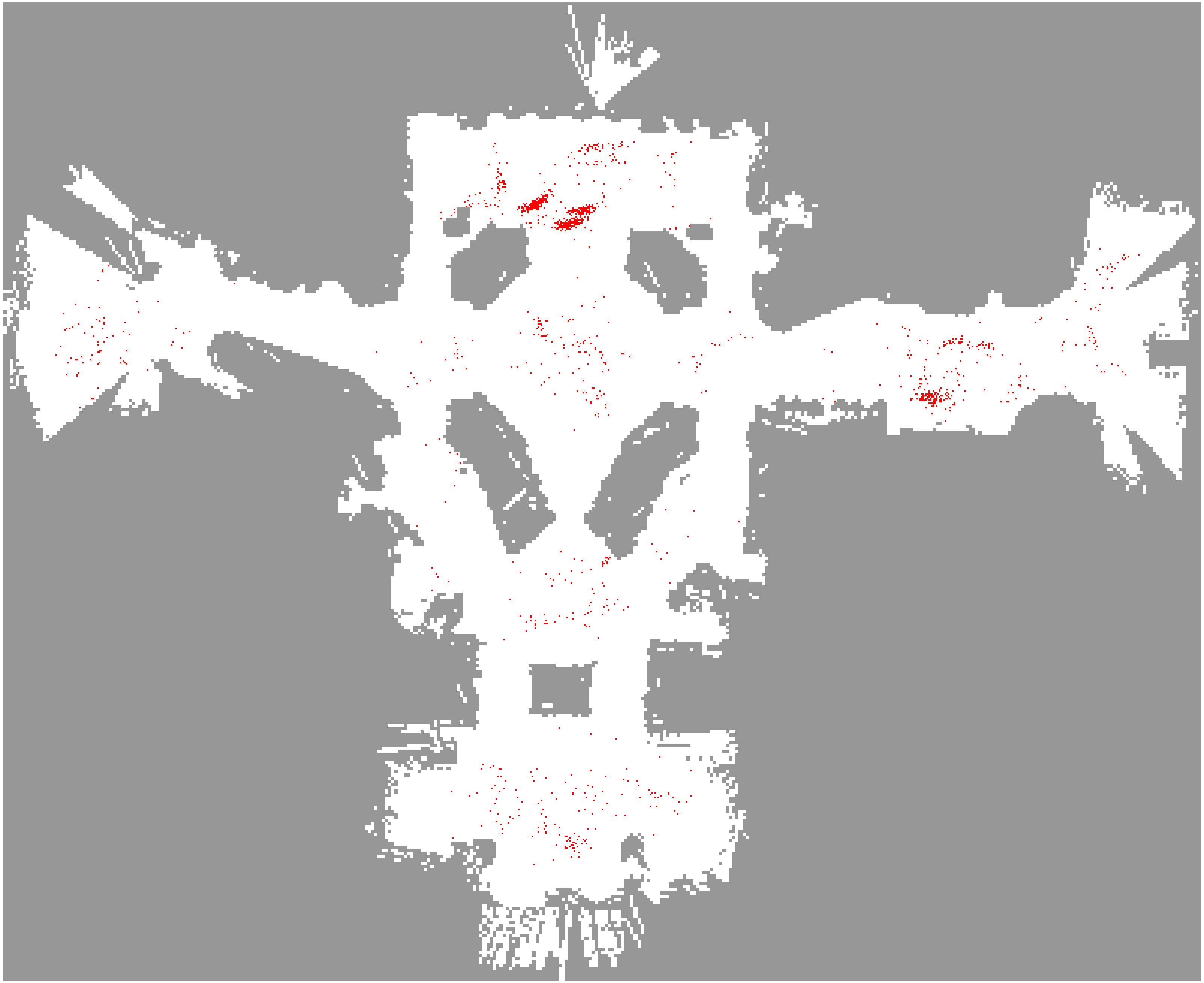


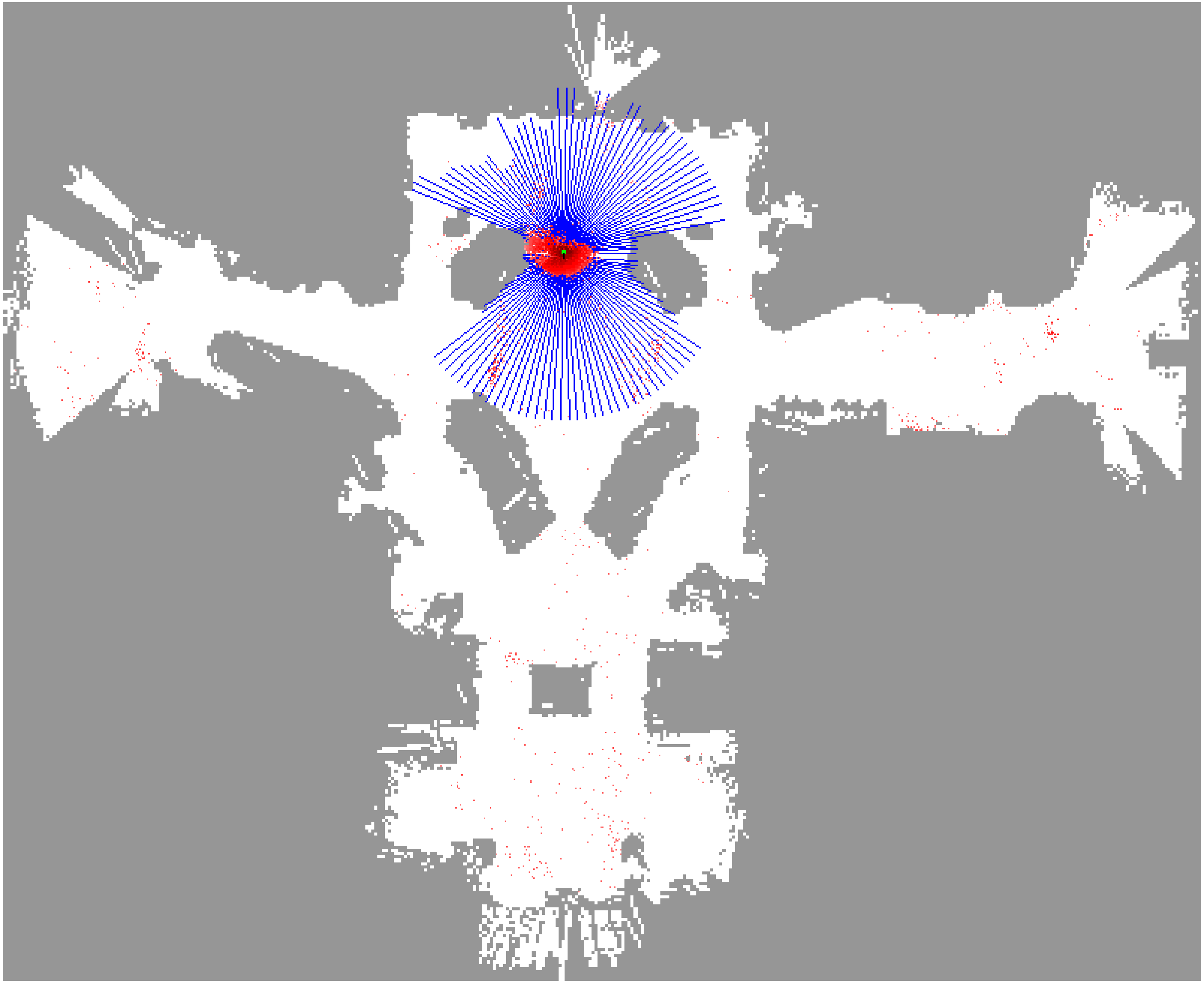


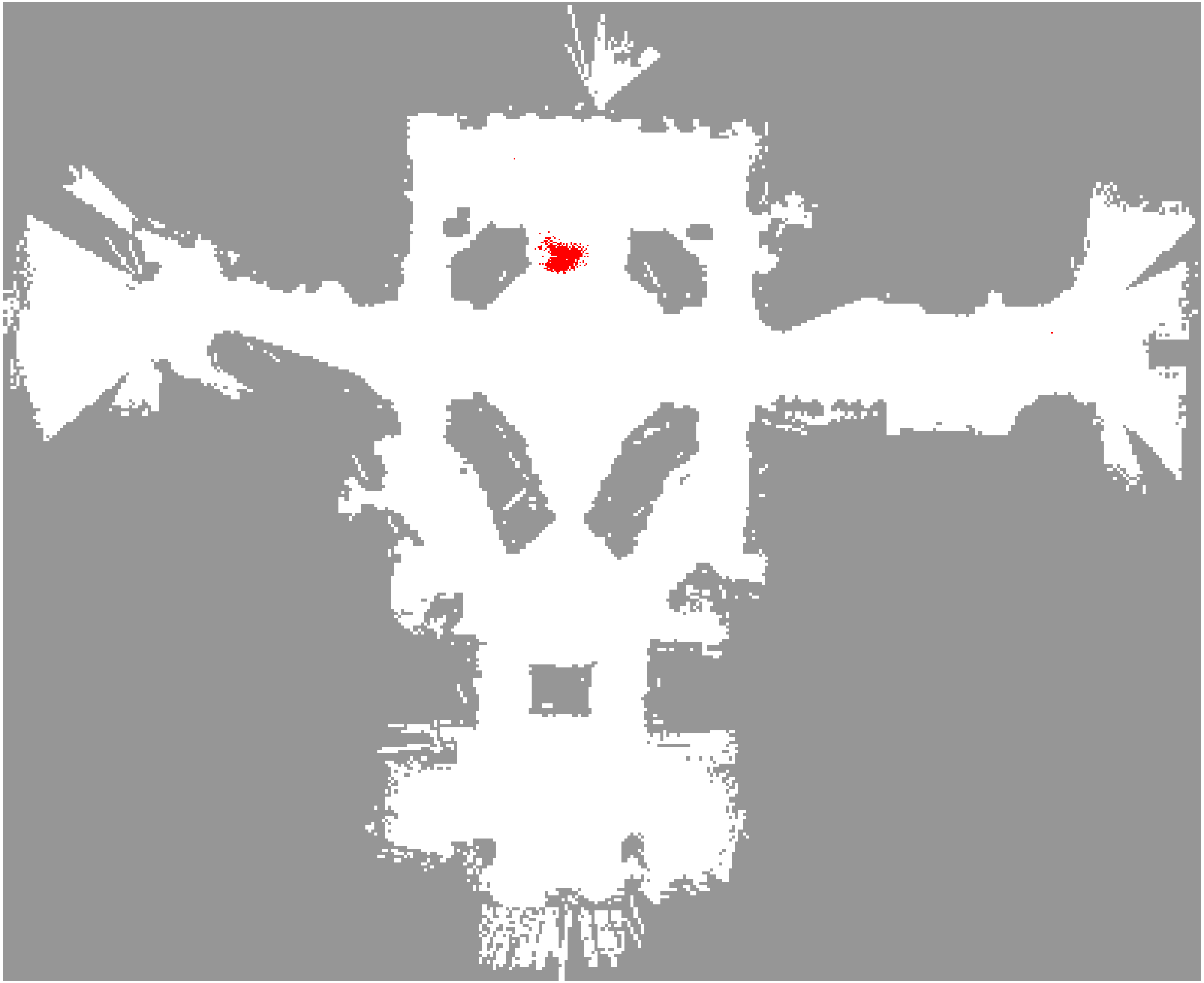


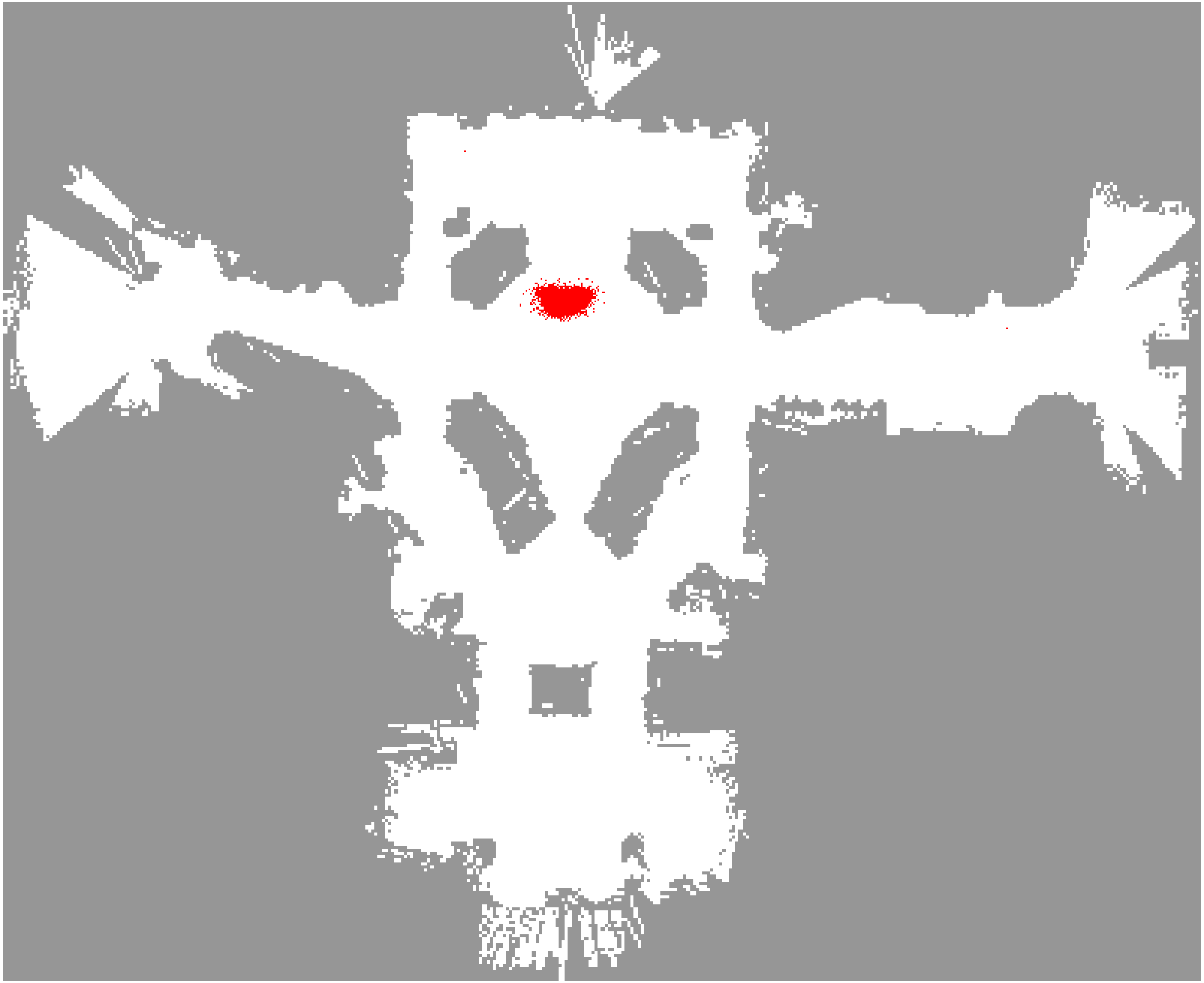


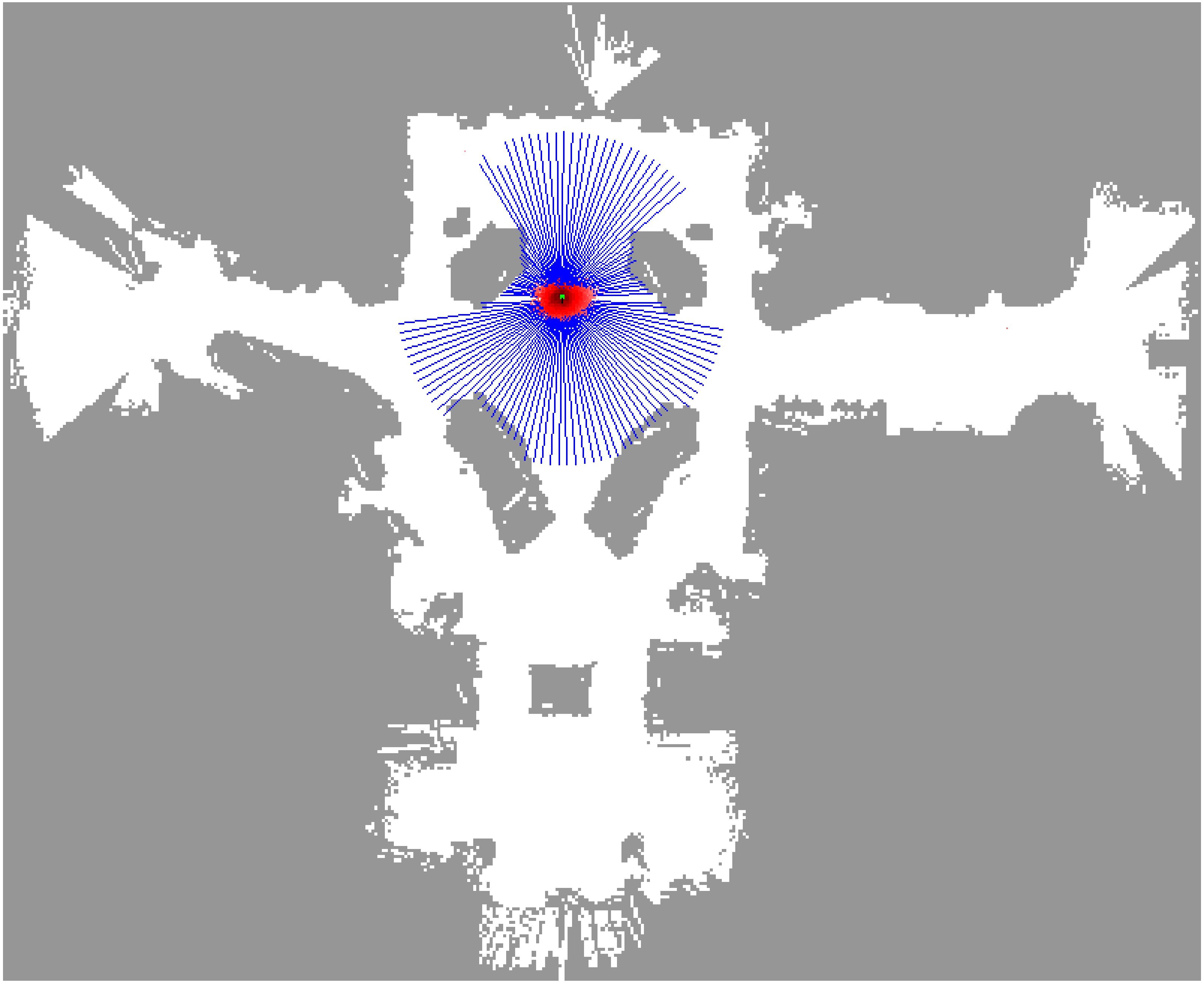


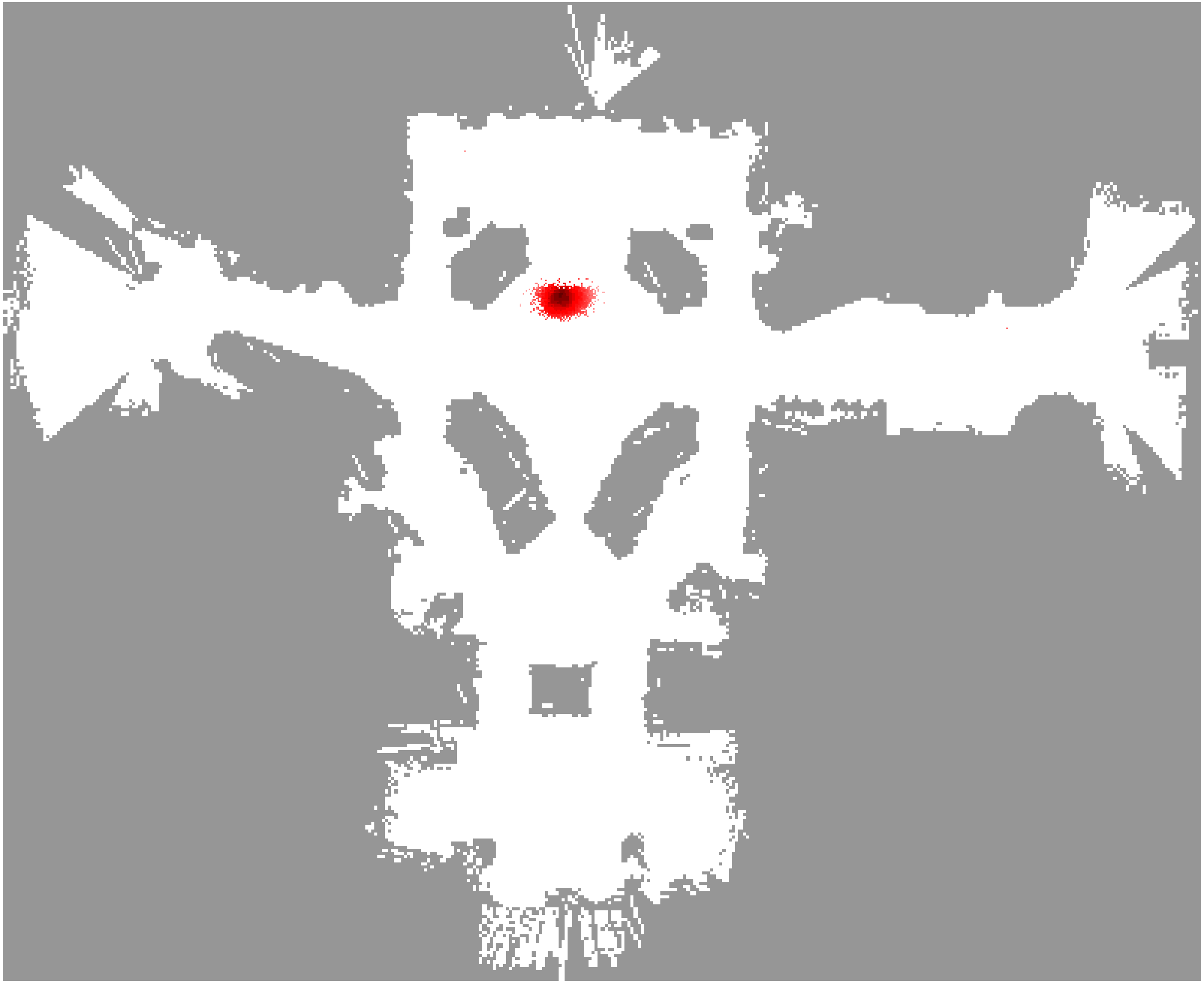


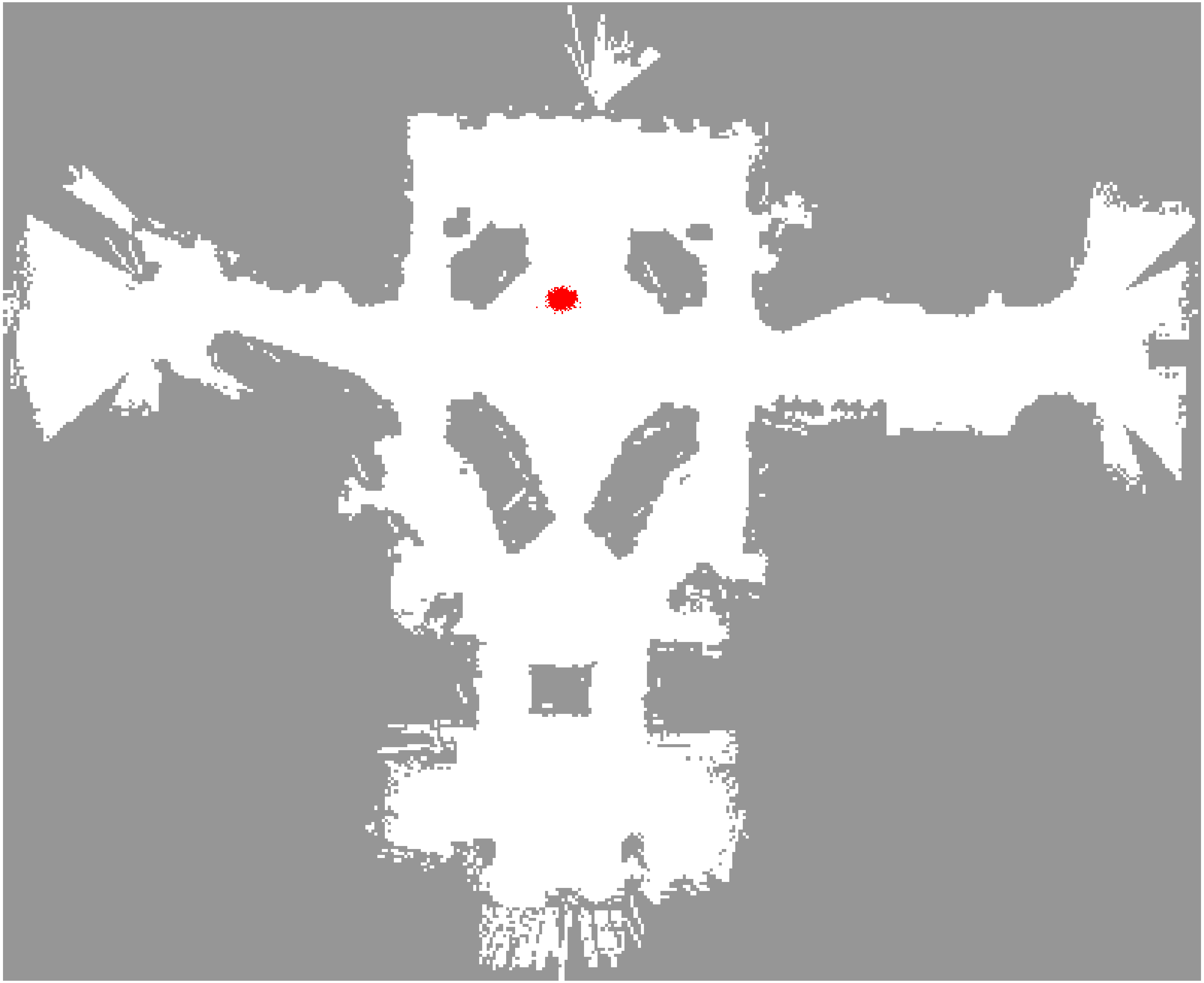


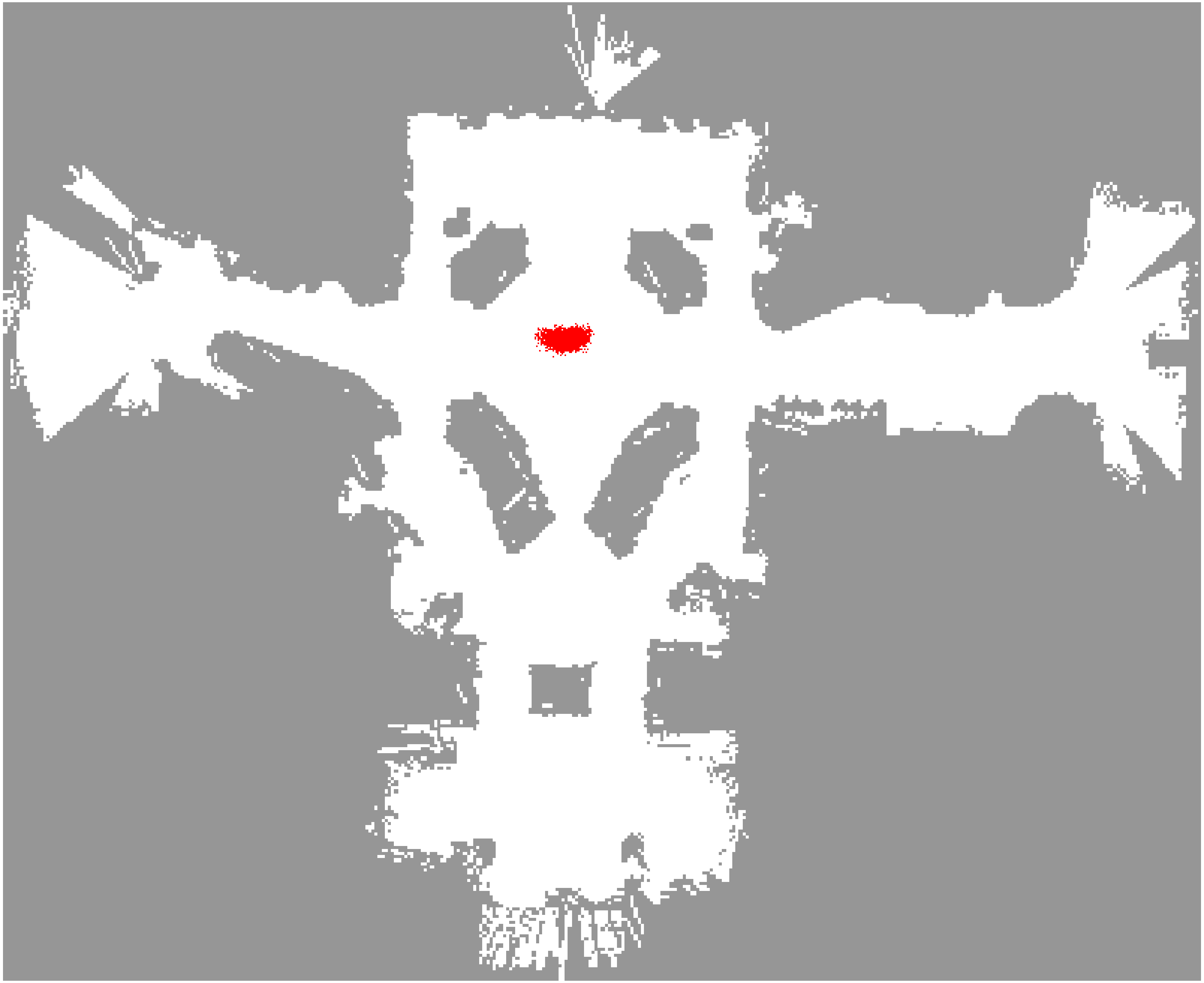


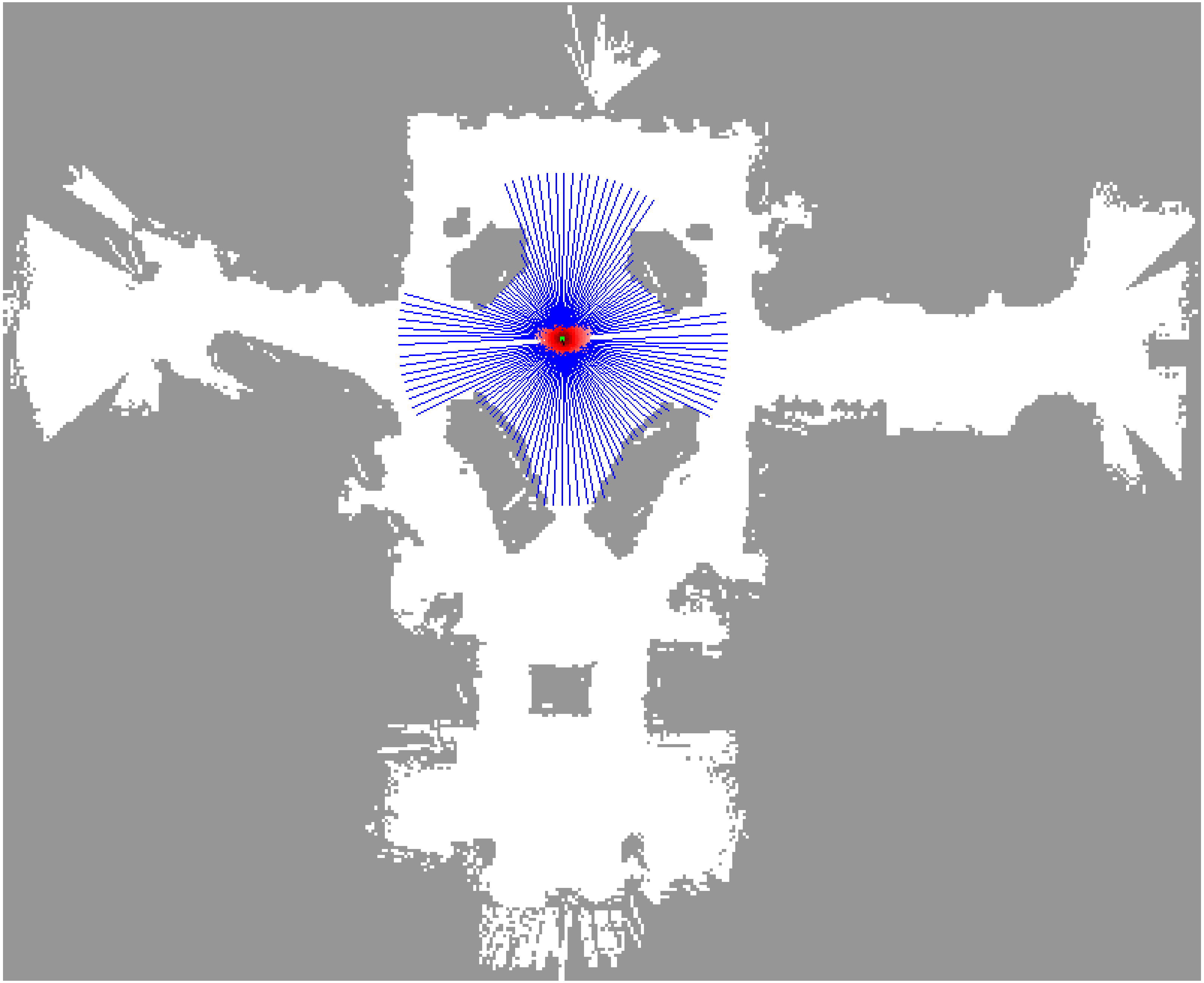


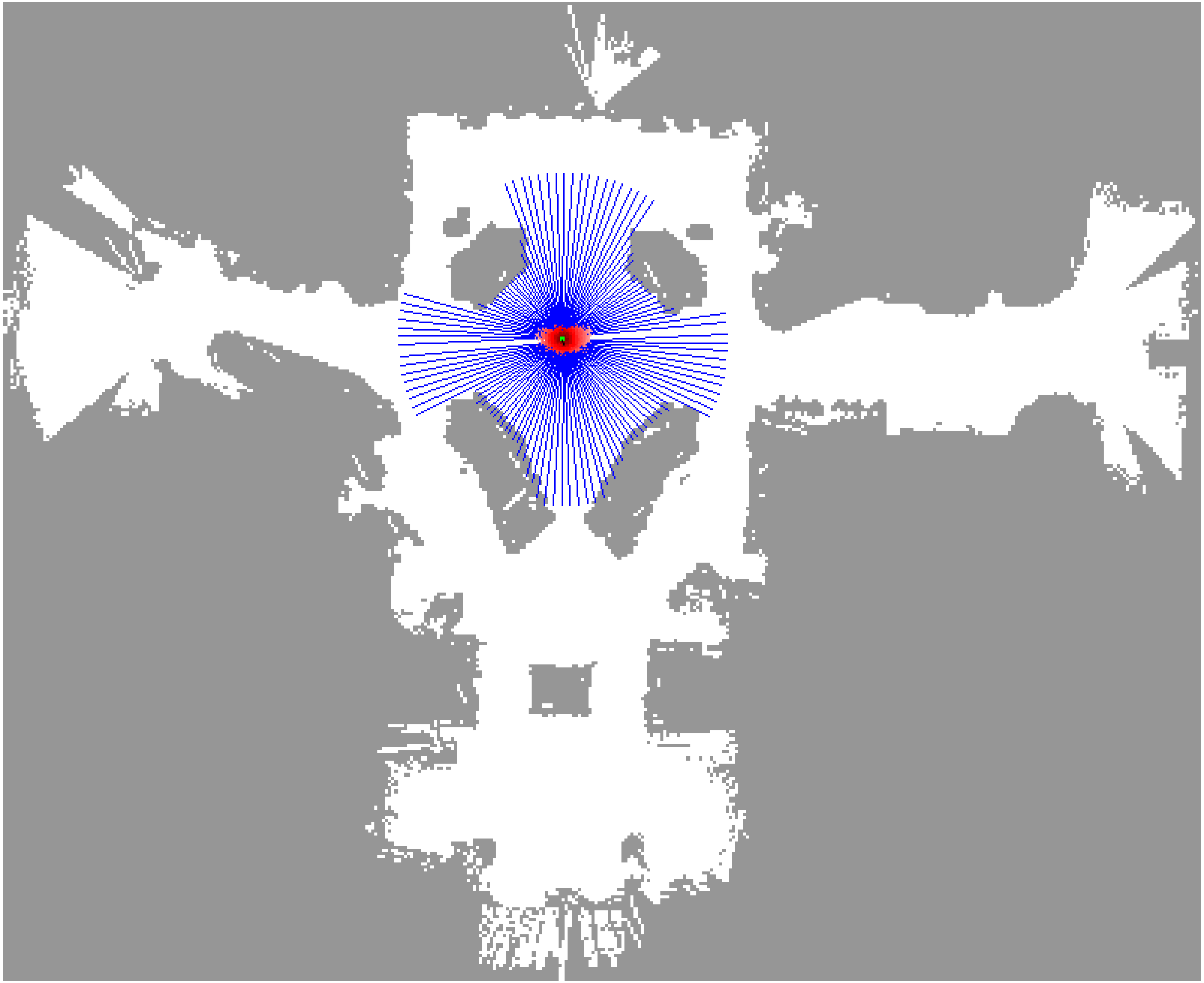




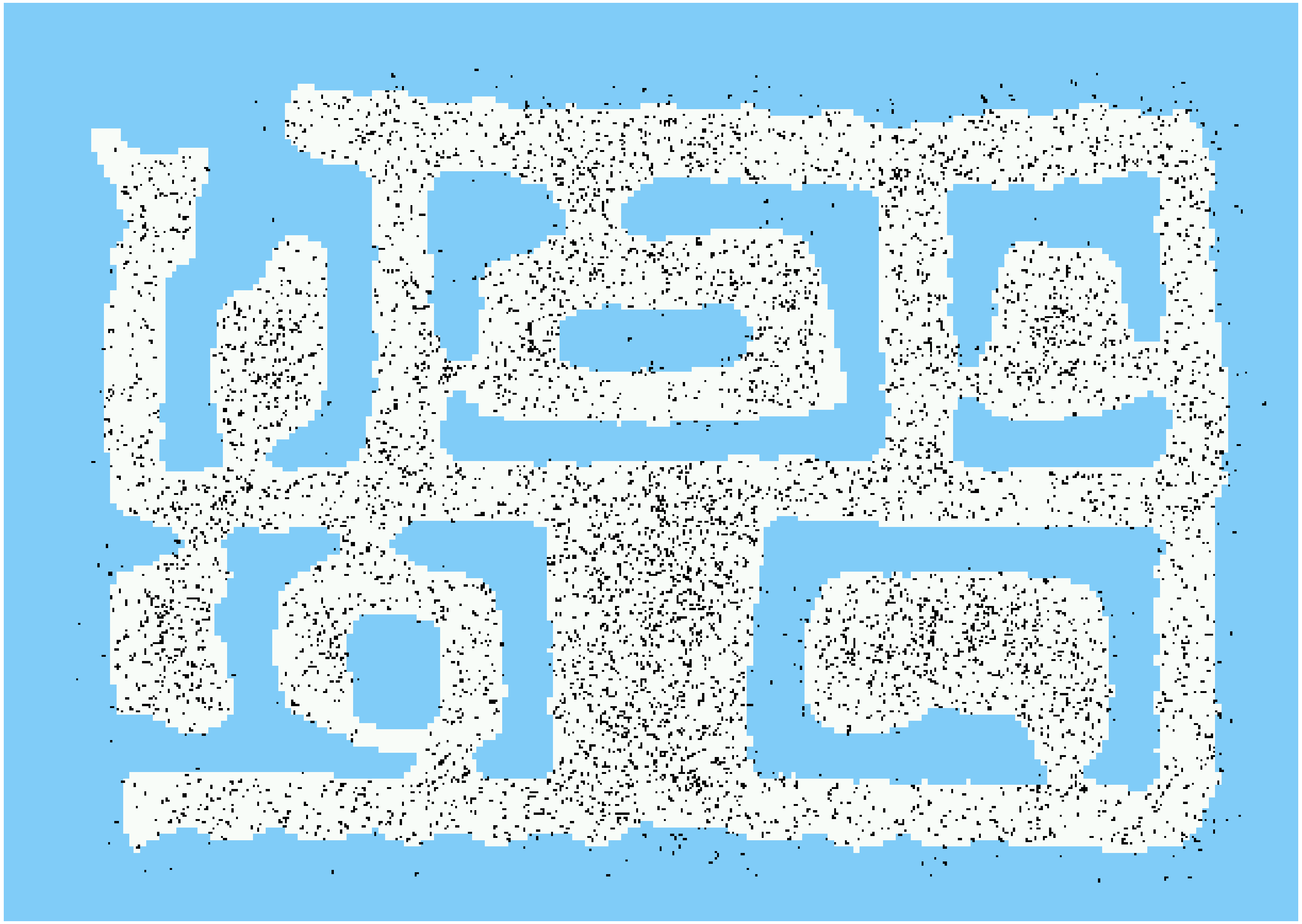




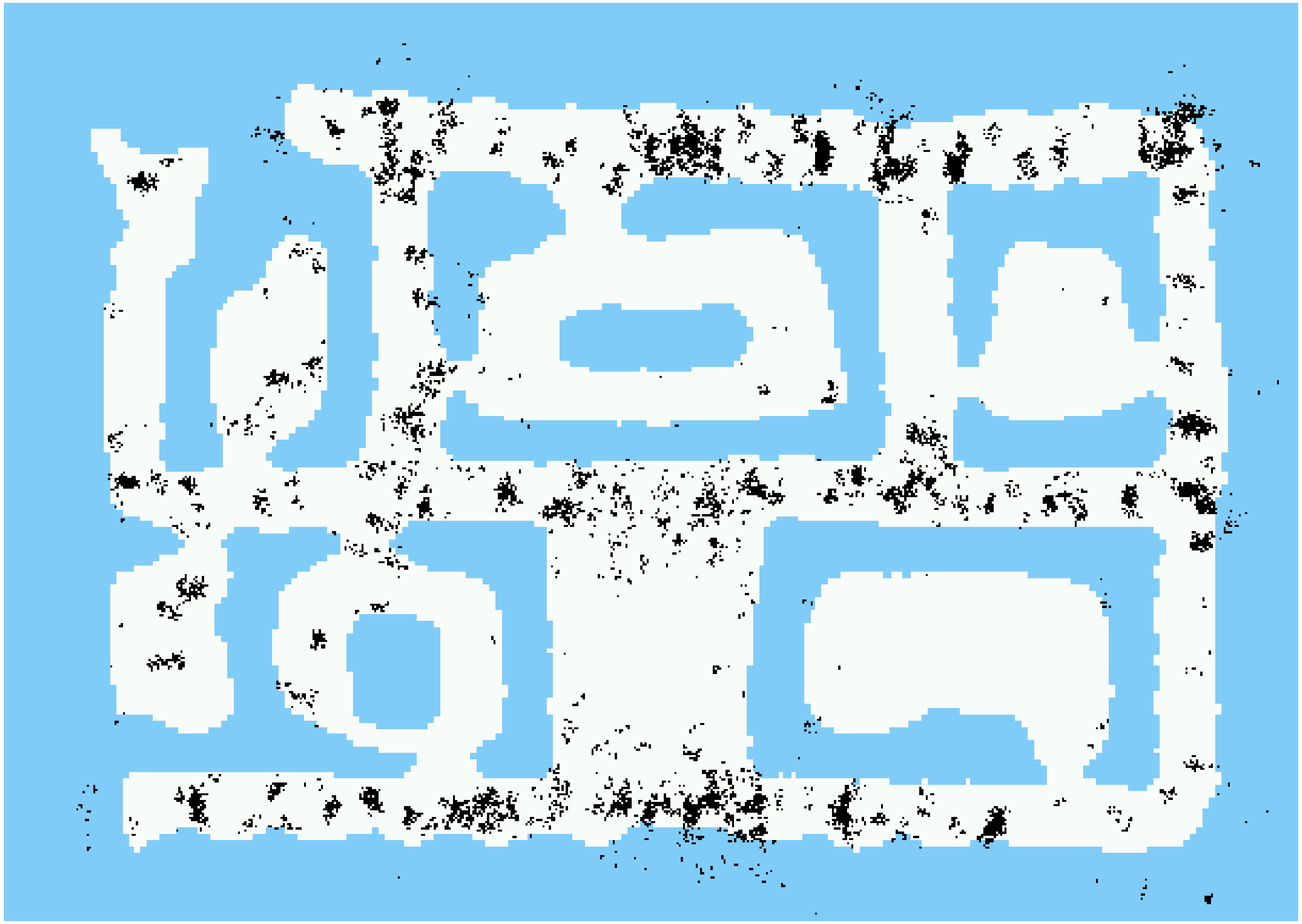




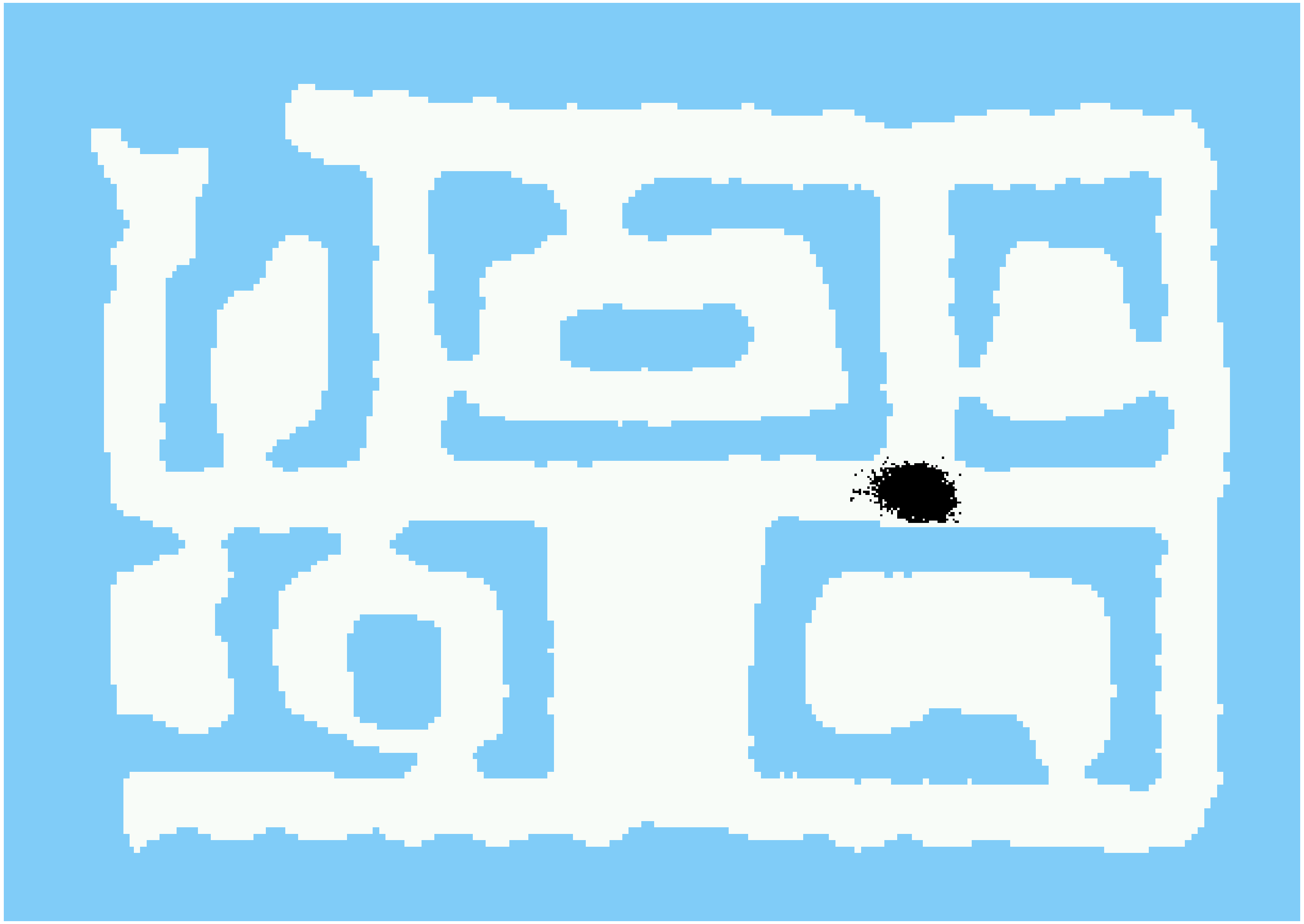
Initial Distribution



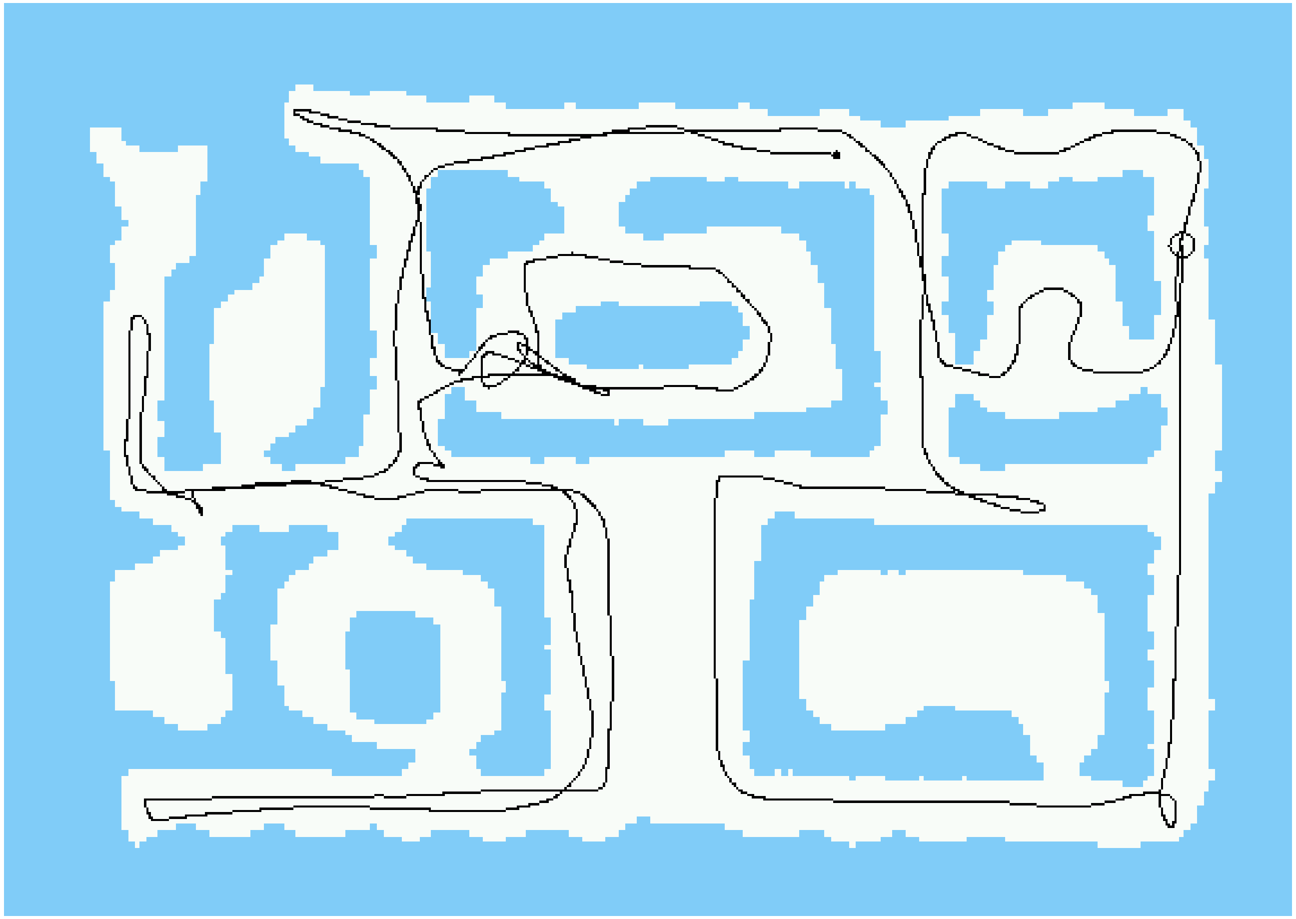
After Incorporating Ten Ultrasound Scans



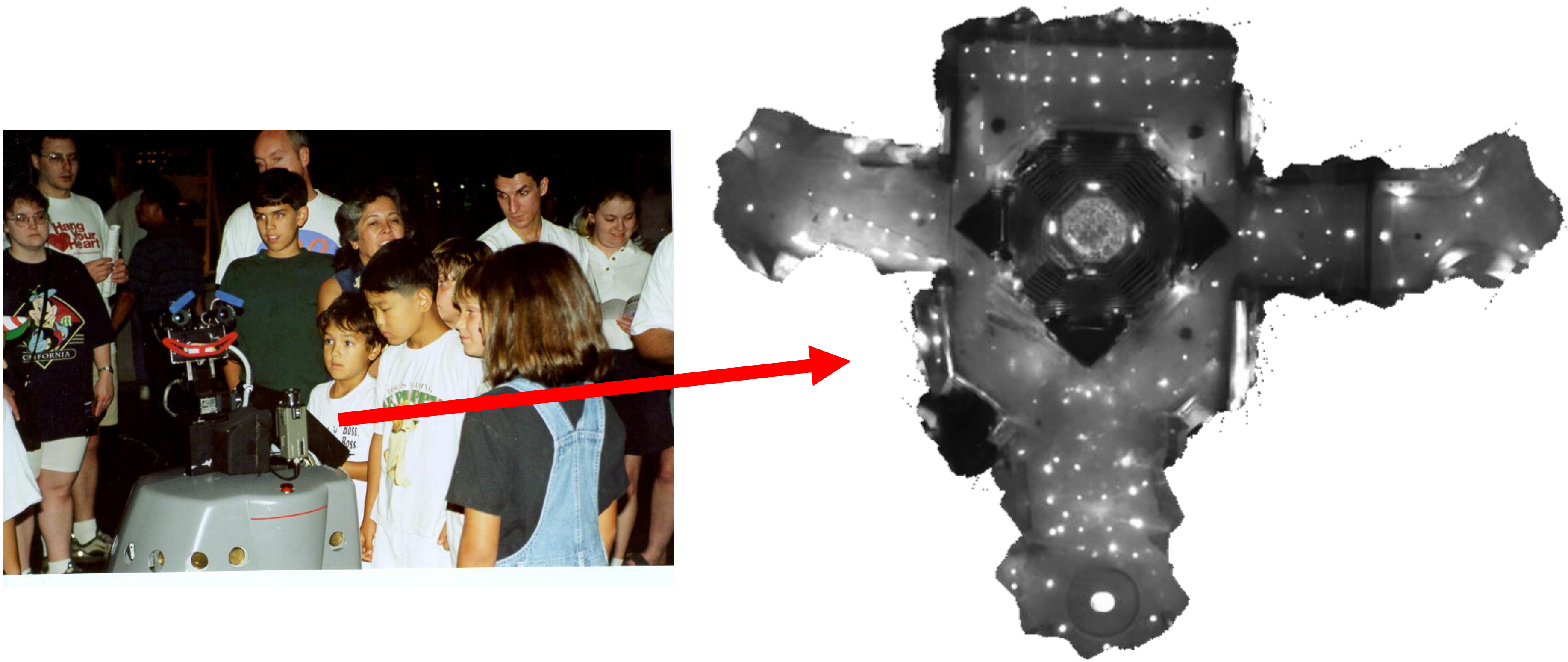
After Incorporating 65 Ultrasound Scans



Estimated Path

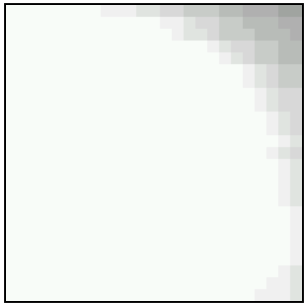


Using Ceiling Maps for Localization

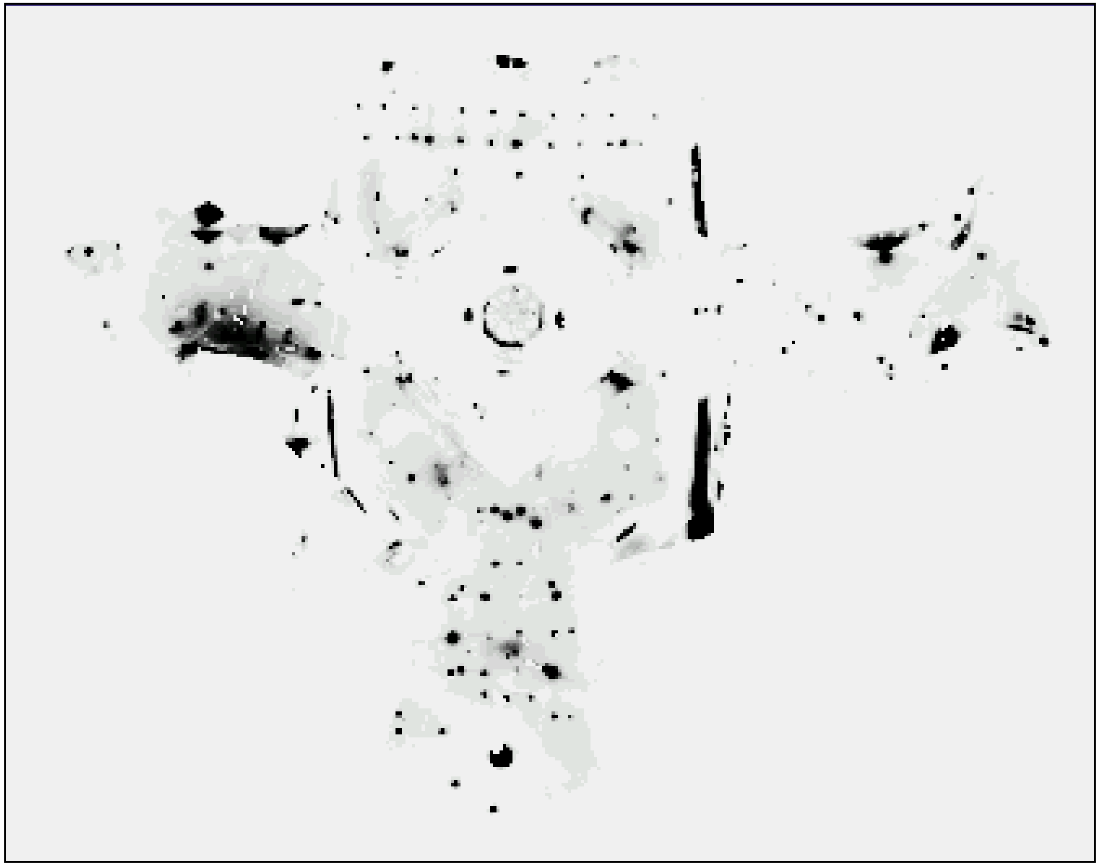


Under a Light

Measurement z:

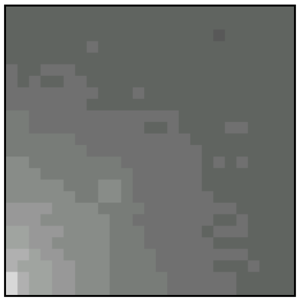


***P(z|x)*:**

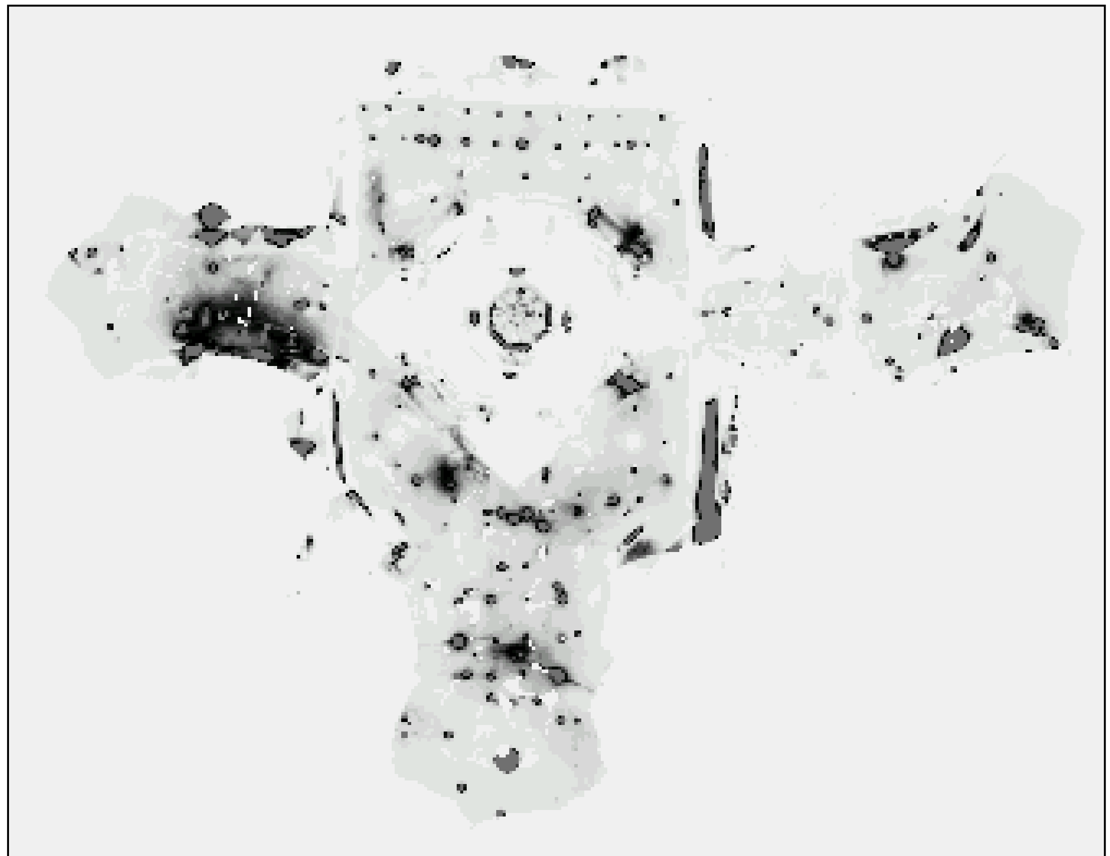


Next to a Light

Measurement z:



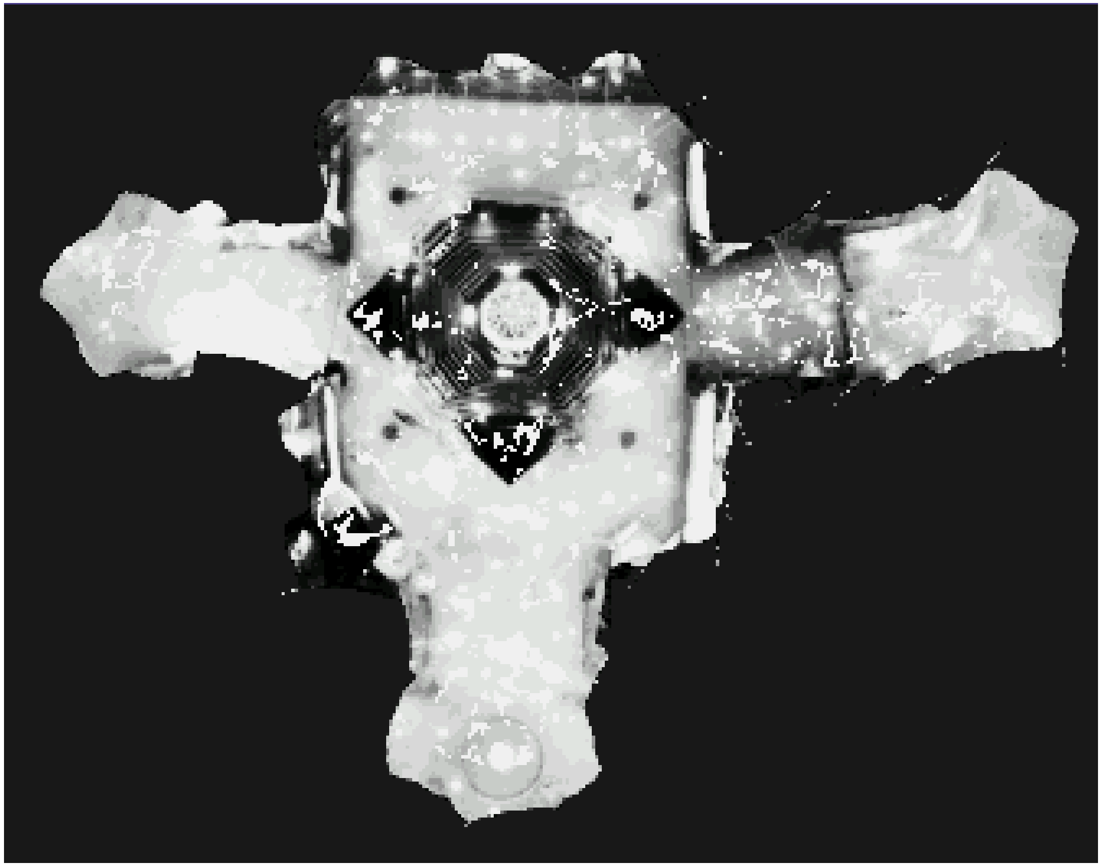
***P(z|x)*:**



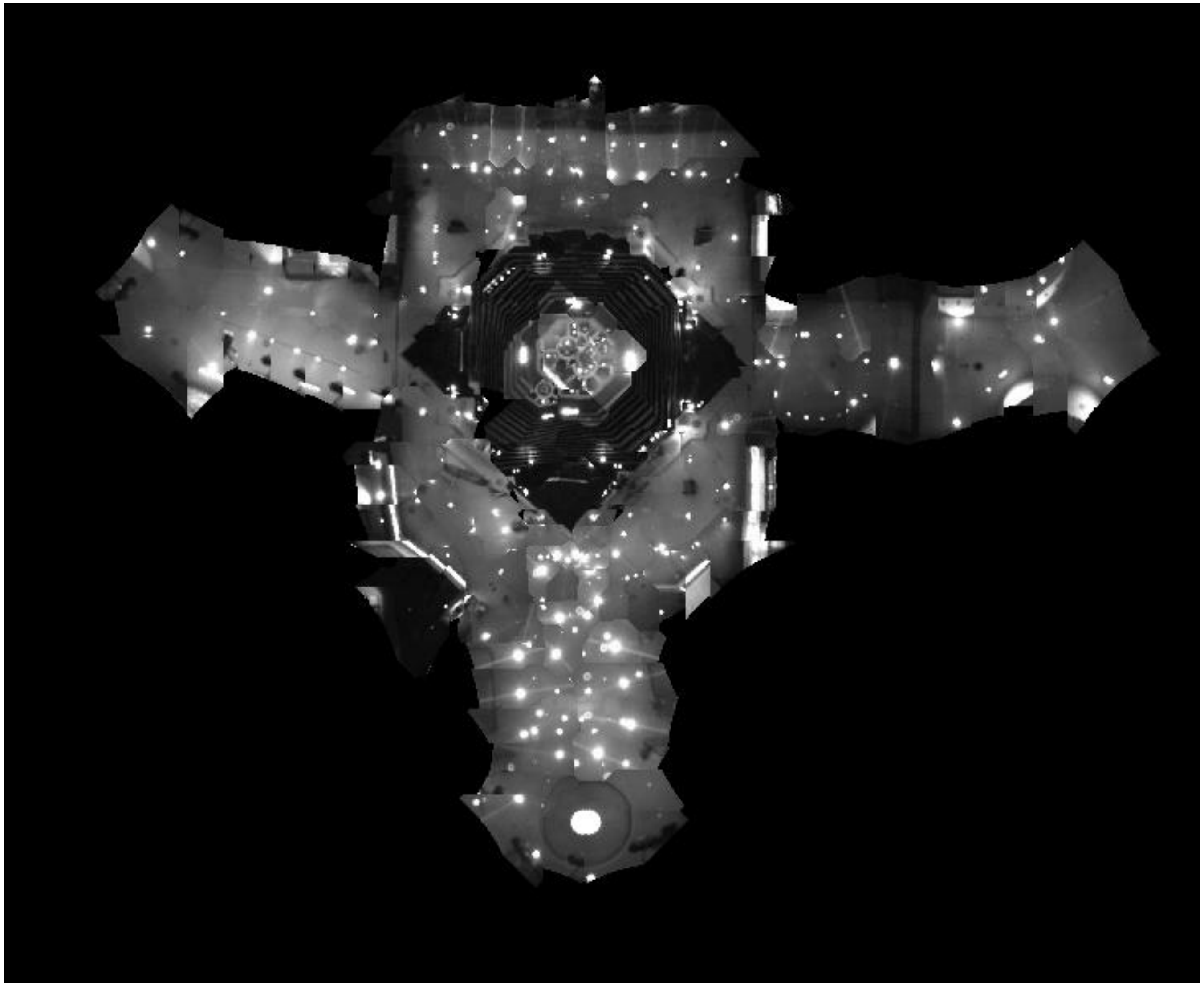
Elsewhere

Measurement z:

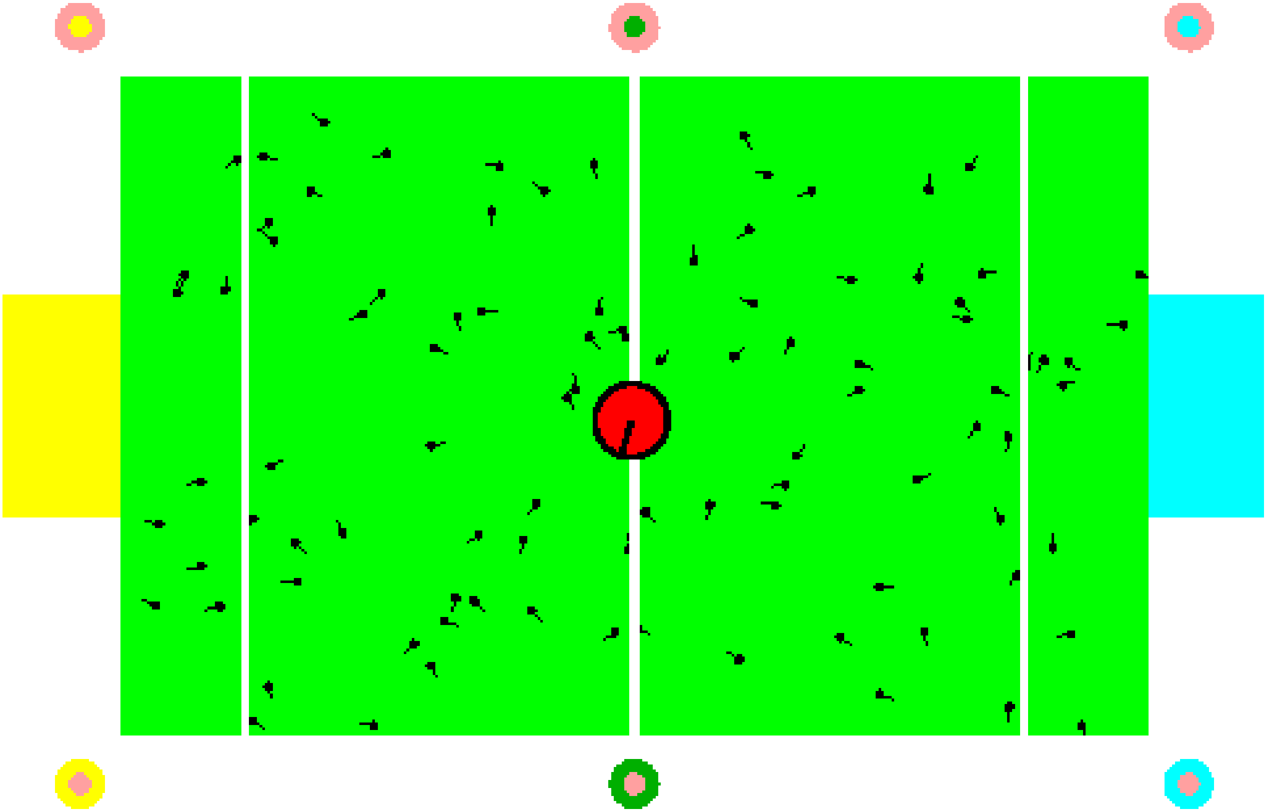
***P(z|x)*:**



Global Localization Using Vision



Localization for AIBO robots



Limitations

- The approach described so far is able to:
  - Track the pose of a mobile robot.
  - Globally localize the robot.
- Can amplify sampling variance, i.e., variability from original distribution due to random sampling.
- *Sampling bias* and *particle deprivation*.
- How can we deal with localization errors, e.g., the *kidnapped robot problem*?

Approaches

- Randomly insert samples;
  - Robot can be “*teleported* “ at any point in time ☺
- Insert random samples proportional to the average likelihood of the particles:
  - Robot has been teleported with higher probability when the likelihood of its observations drops.

Summary

- Particle filters instance of recursive Bayesian filtering.
- Represent the posterior by a set of weighted samples.
- In the context of localization, particles are propagated according to the motion model.
- Particles are then weighted according to the likelihood of the observations.
- During re-sampling, new particles are drawn with probability proportional to the weights.

What Next?

- SLAM!
- EKF-SLAM and Fast-SLAM.
- Probabilistic sequential decision making.