SLAM

These slides are based on:
Probabilistic Robotics,
S. Thrun, W. Burgard,
D. Fox, MIT Press, 2005
and
Chang Young Kim's Slides

Lecturers: Dr. Yehuda Elmaliah, Mr. Roi Yehoshua
elmaliahy@colman.ac.il

Agenda

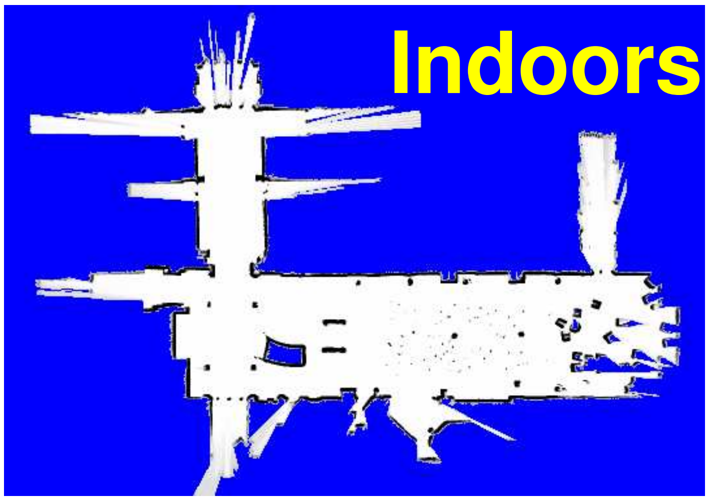
המסלול האקדמי
המכללה למינהל

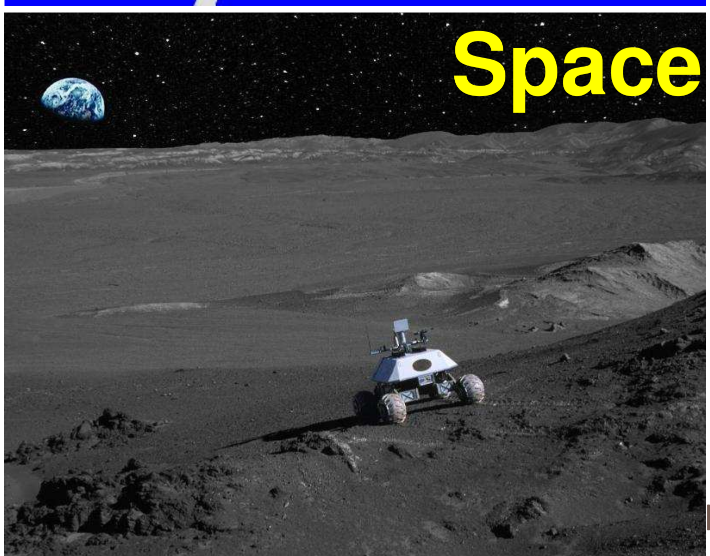
- The SLAM Problem
- SLAM algorithm
- Bayes Filter
- Particle Filters

The SLAM Problem

- A robot is exploring an unknown, static environment
- No map is available and no pose info
- **Given:**
  - The robot's controls
  - Observations of nearby features
- **Estimate:**
  - Map of features
  - Path of the robot

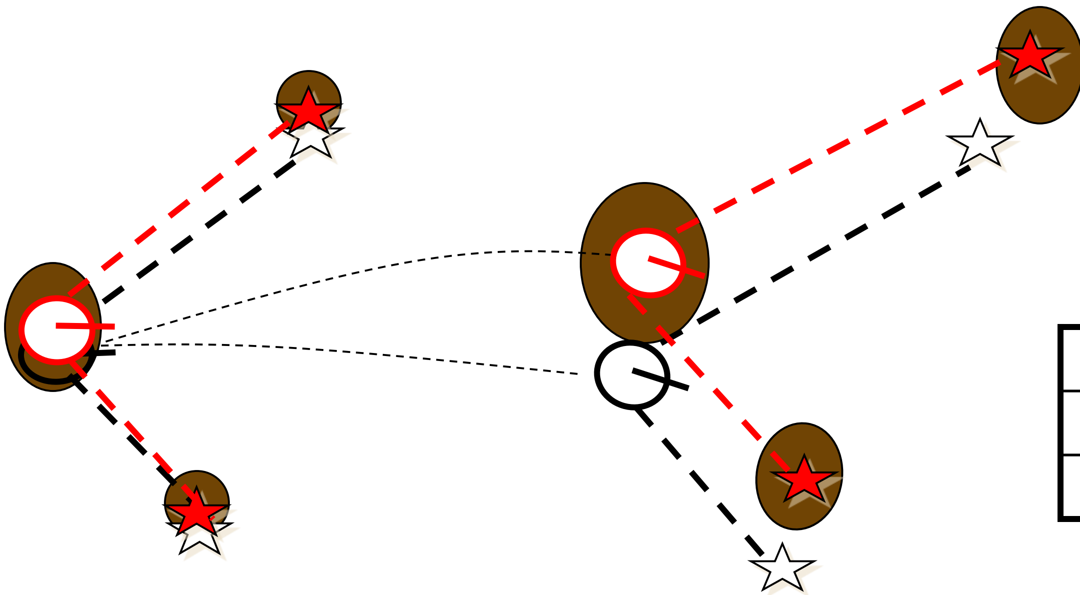
SLAM Applications





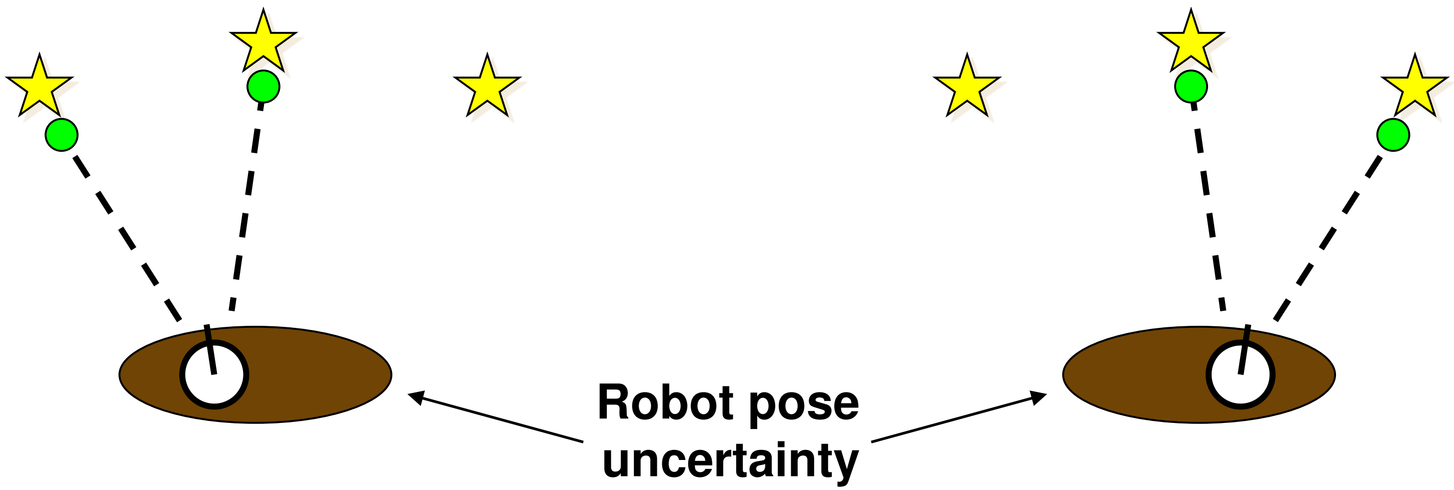
Why is SLAM a Hard Problem?

SLAM: robot path and map are both **unknown**



	Robot	Landmark
Estimated		
True		

Why is SLAM a Hard Problem?



- In the real world, the mapping between observations and landmarks is unknown
- Picking wrong data associations can have catastrophic consequences
- Pose error correlates data associations

Terminology

- Robot State (or pose): $x_t = [x, y, \theta]$
  - Position and heading
  - $x_{1:t} = \{x_1, \ldots, x_t\}$
- Robot Controls: u_t
  - Robot motion and manipulation
  - $u_{1:t} = \{u_1, \ldots, u_t\}$
- Sensor Measurements: z_t
  - Range scans, images, etc.
  - $z_{1:t} = \{z_1, \ldots, z_t\}$
- Landmark or Map:
  - Landmarks or Map

$$m = \{m_1, \ldots, m_n\} \text{ or } l = \{l_1, \ldots, l_n\}$$

Terminology

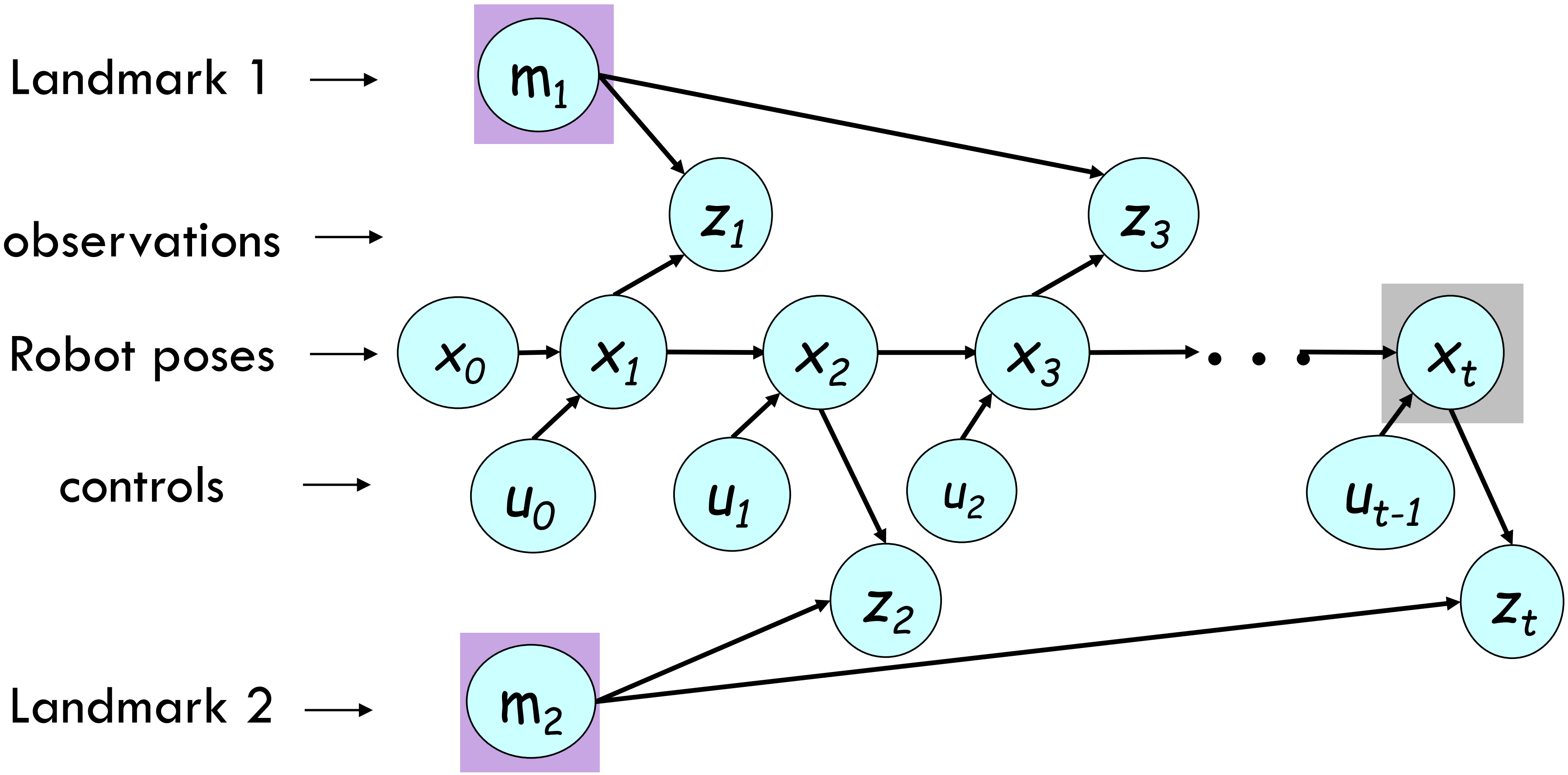
- Observation model: $P(z_t \mid x_t)$ or $P(z_t \mid x_t, m)$
  - The probability of a measurement z_t given that the robot is at position x_t *and map m*
- Motion model: $P(x_t \mid x_{t-1}, u_t)$
  - The posterior probability that action u_t carries the robot from x_{t-1} to x_t.

המסלול האקדמי
המכללה למינהל

Terminology

- Belief: $bel(x_t)$
  - Posterior probability
  - Conditioned on available data
  - $bel(x_t) = p(x_t \mid z_{1..t}, u_{1..t})$
- Prediction: $\overline{bel}(x_t)$
  - Estimate before measurement data
  - $\overline{bel}(x_t) = p(x_t \mid z_{1..t-1}, u_{1..t})$

Graphical Model of SLAM



- Need to compute: $p(x_t, m \mid z_{1:t}, u_{1:t})$

Bayes Filter

- Bayes filter is a recursive algorithm for calculating the belief at time *t* from the belief at time *t* – 1
- **Prediction:**

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

- **Update:**

$$\mathrm{bel}(x_t, m) = \eta\, p(z_t \mid x_t, m)\, \overline{\mathrm{bel}}(x_t, m)$$

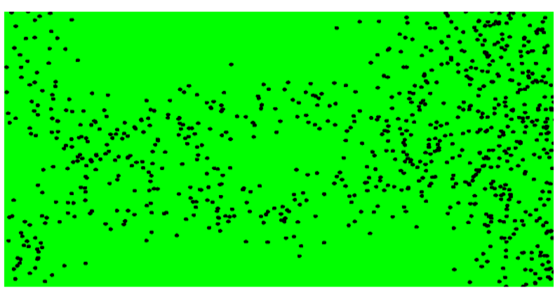
Normalization factor

Different SLAM Techniques

- EKF (Extended Kalman Filter) SLAM
- Particle Filter SLAM
- Fast-SLAM (Rao-Blackwellisation)
- Graph-SLAM, SEIFs

Particle Filters

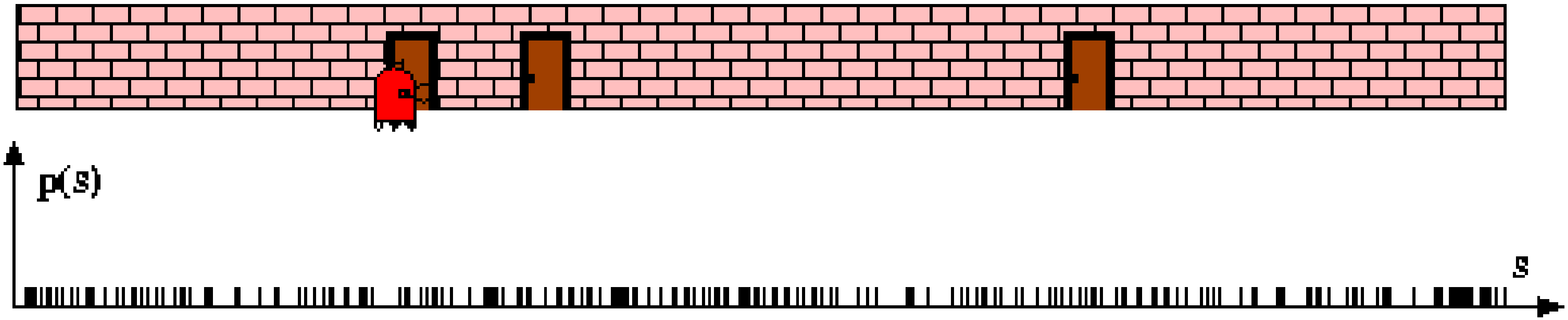
- Represent belief by random samples
- Estimation of non-Gaussian, nonlinear processes
- Sampling Importance Resampling (SIR) principle
  - Draw the new generation of particles
  - Assign an importance weight to each particle
  - Resampling



Weighted samples

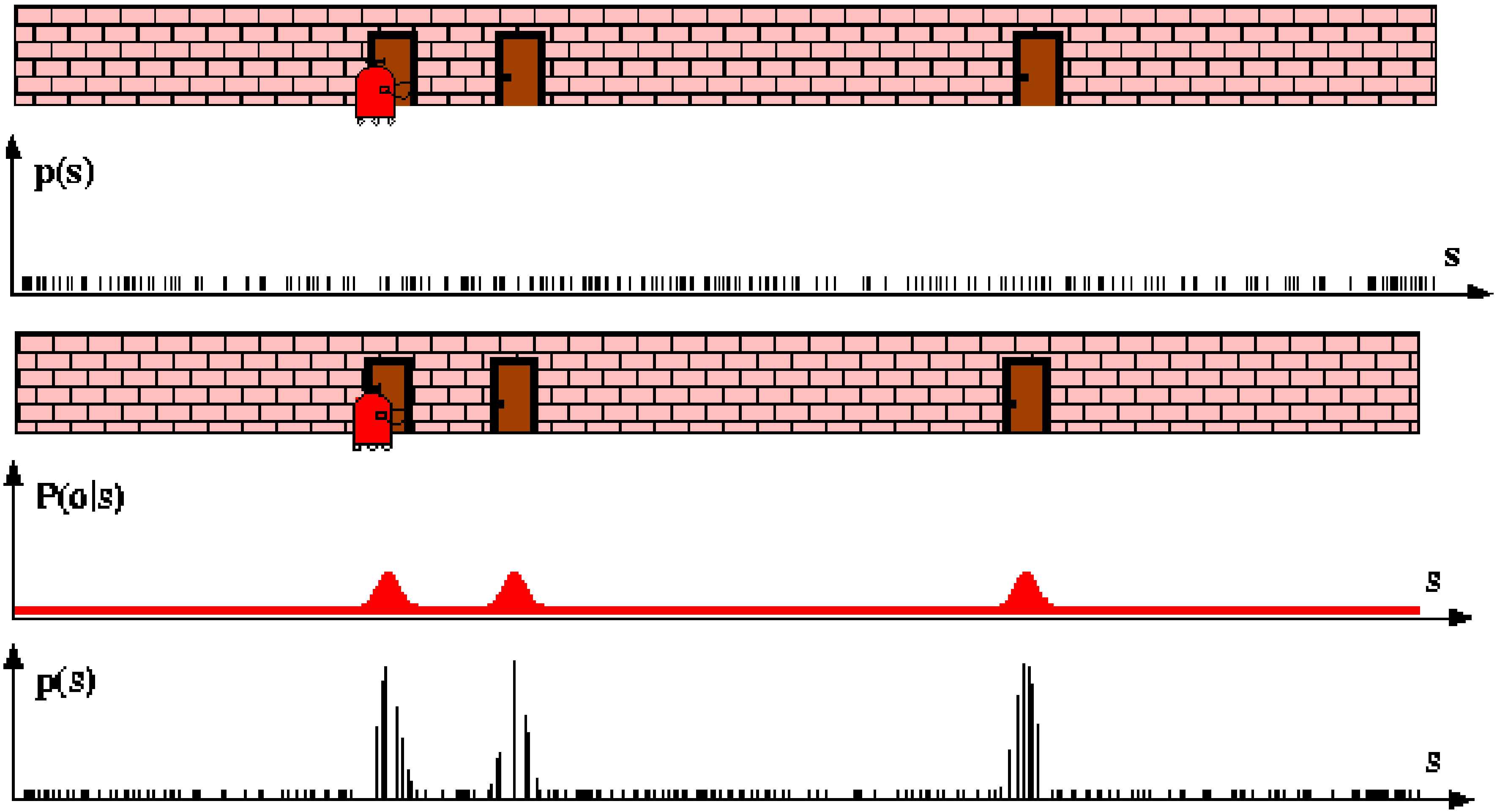
After resampling

Particle Filters



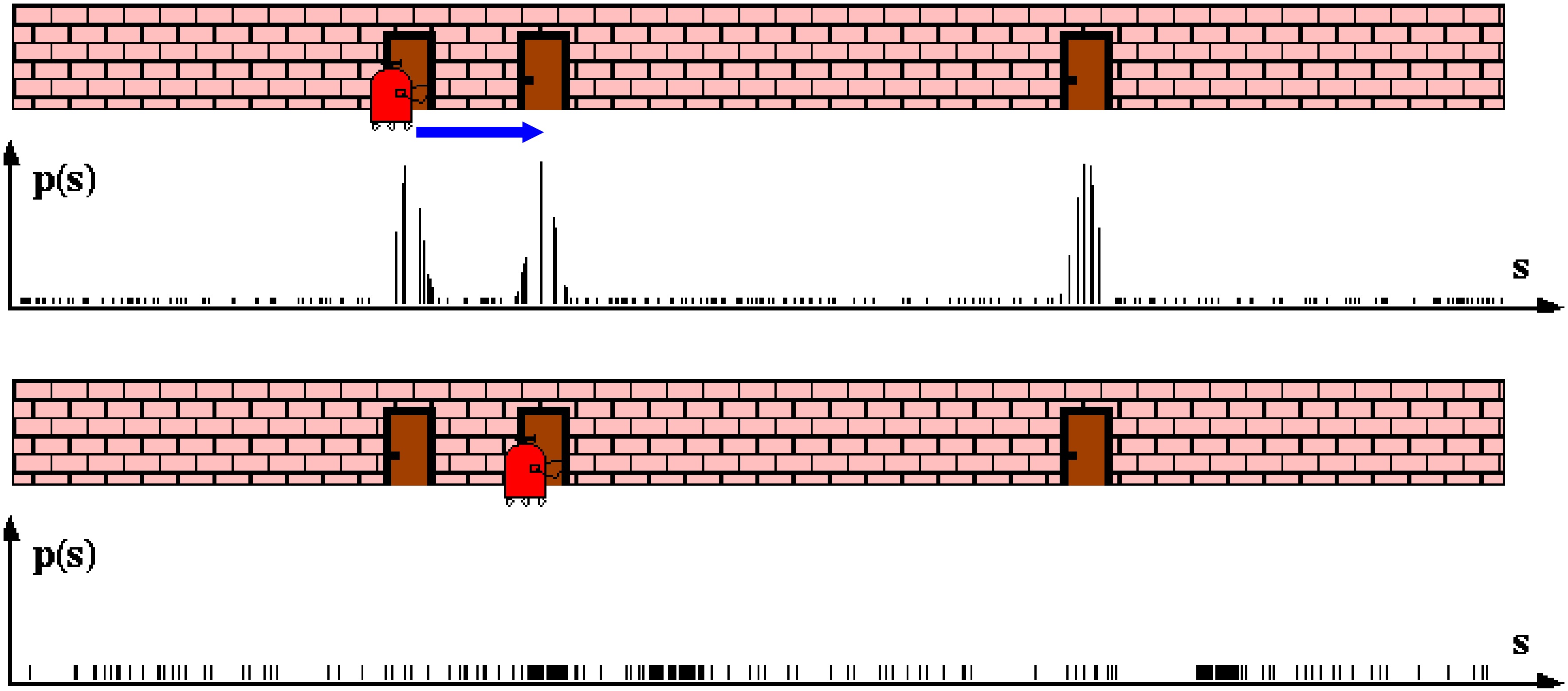
Sensor Information – Importance Sampling

$$\boxed{w_t = p(z_t \mid x_t)}$$



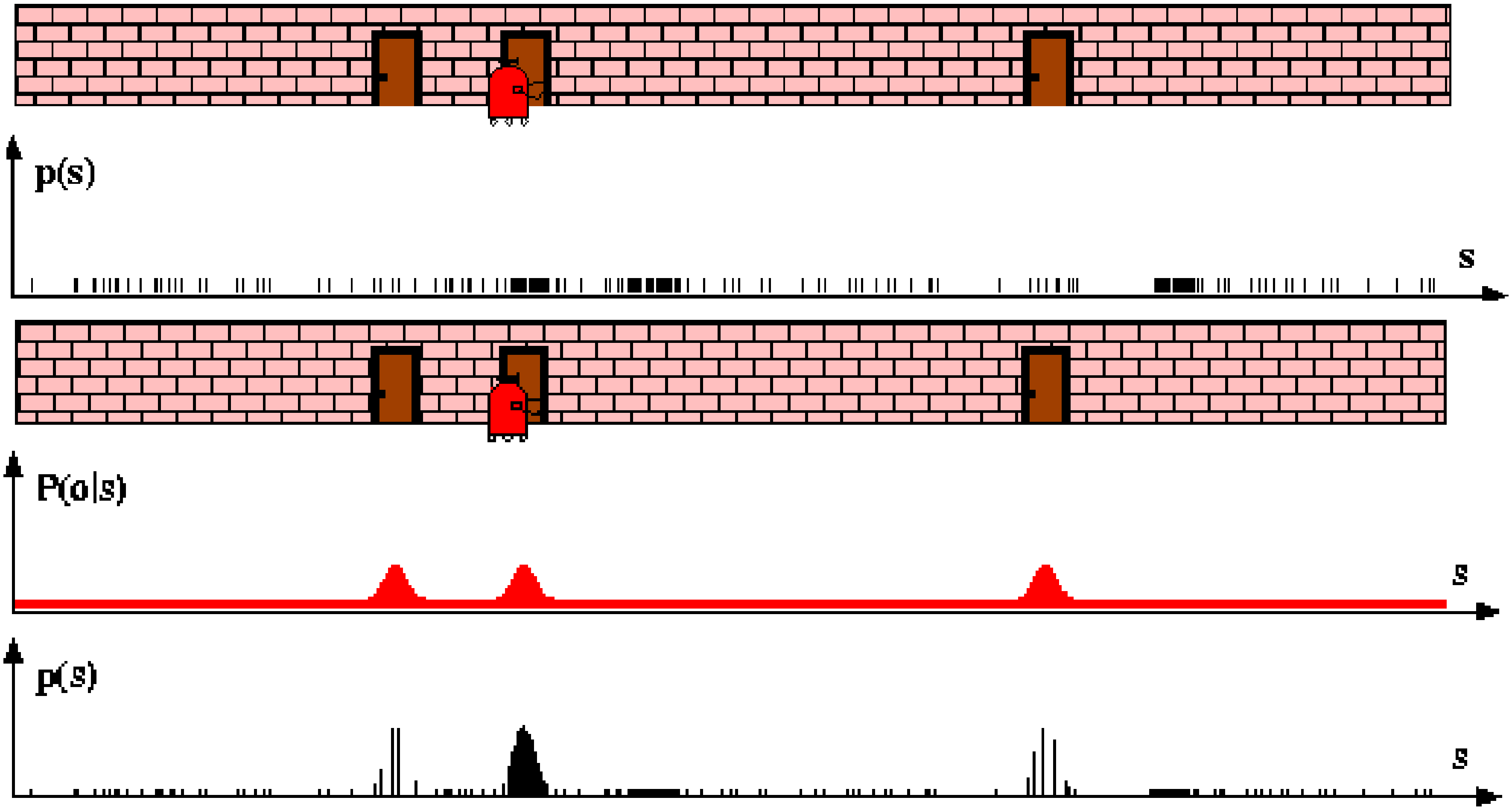
Robot Motion

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



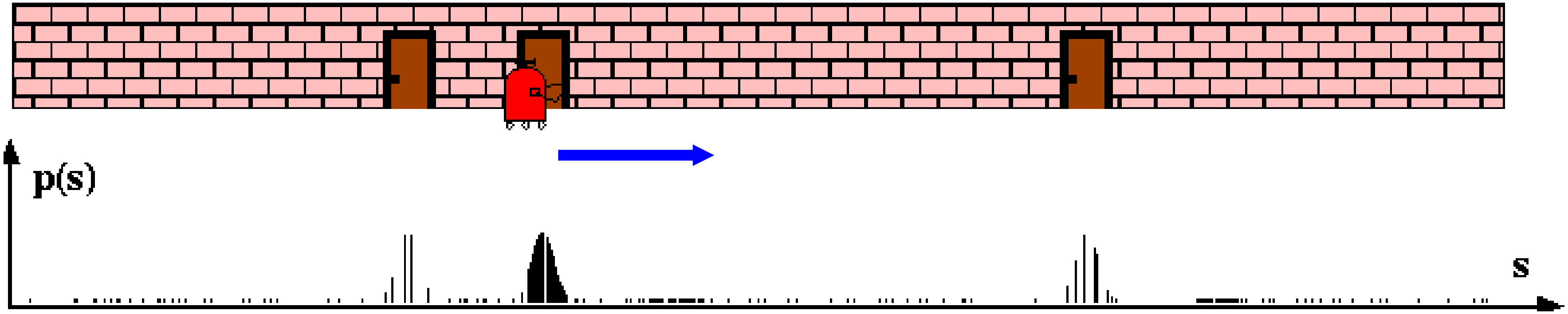
Sensor Information – Importance Sampling

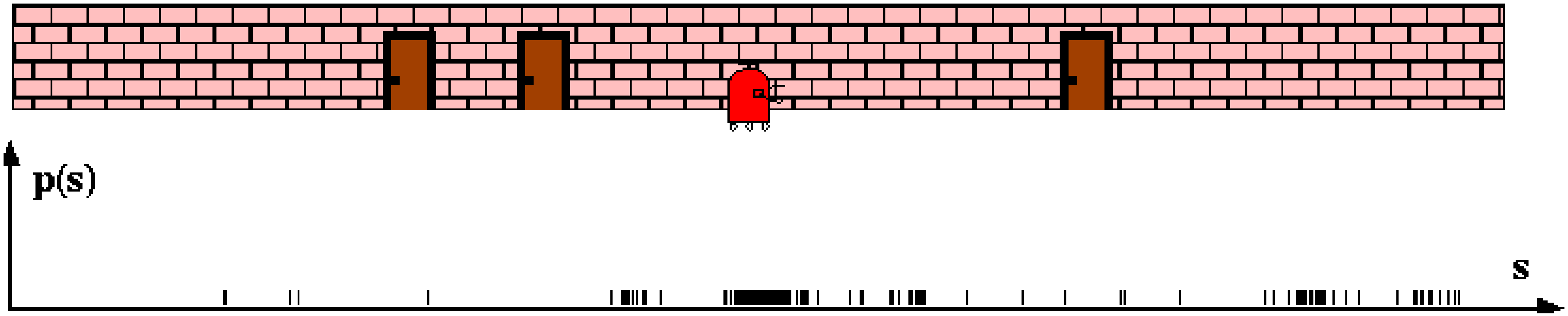
$$w_t = p(z_t \mid x_t)$$



Robot Motion

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





Particle Filter Algorithm

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

draw x^i_{t-1} from $Bel(\mathbf{x}_{t-1})$

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

Importance factor for x^i_t:

$$w^i_t = \frac{\text{target distribution}}{\text{proposal distribution}} = \frac{\eta\, p(z_t \mid x_t)\, p(x_t \mid x_{t-1}, u_{t-1})\, Bel(x_{t-1})}{p(x_t \mid x_{t-1}, u_{t-1})\, Bel(x_{t-1})} \propto p(z_t \mid x_t)$$

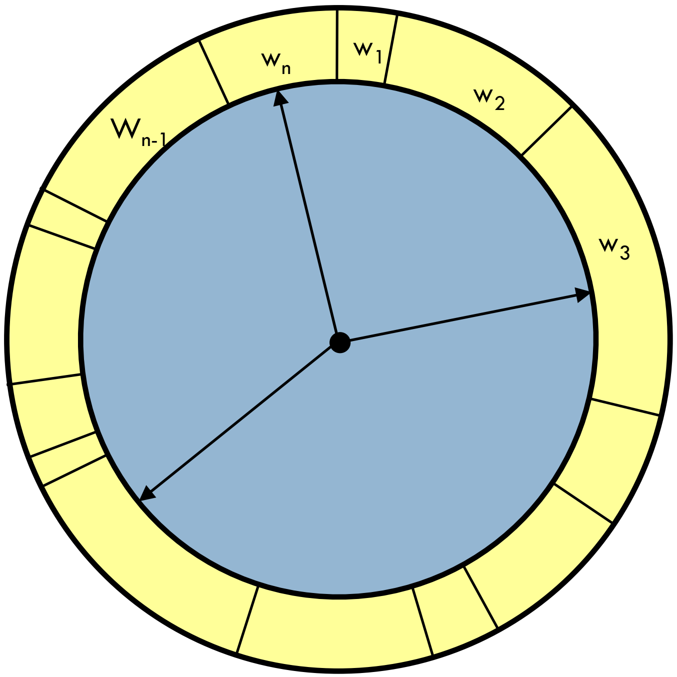
Particle Filter Pseudo-Code

1. Algorithm **particle_filter**(S_{t-1}, u_{t-1} z_t):
2. $S_t = \emptyset, \quad \eta = 0$
3. **For** $i = 1 \ldots n$ *Generate new samples*
4. Sample index $j(i)$ from the discrete distribution given by w_{t-1}
5. Sample x_t^i from $p(x_t \mid x_{t-1}, u_{t-1})$ using $x_{t-1}^{j(i)}$ and u_{t-1}
6. $w_t^i = p(z_t \mid x_t^i)$ *Compute importance weight*
7. $\eta = \eta + w_t^i$ *Update normalization factor*
8. $S_t = S_t \cup \{< x_t^i, w_t^i >\}$ *Insert*
9. **For** $i = 1 \ldots n$
10. $w_t^i = w_t^i / \eta$ *Normalize weights*

Resampling

- **Given**: Set *S* of weighted samples
- **Wanted**: Random sample, where the probability of drawing x_i is given by w_i
- Typically done *n* times with replacement to generate new sample set *S'*

Resampling – Roulette Wheel

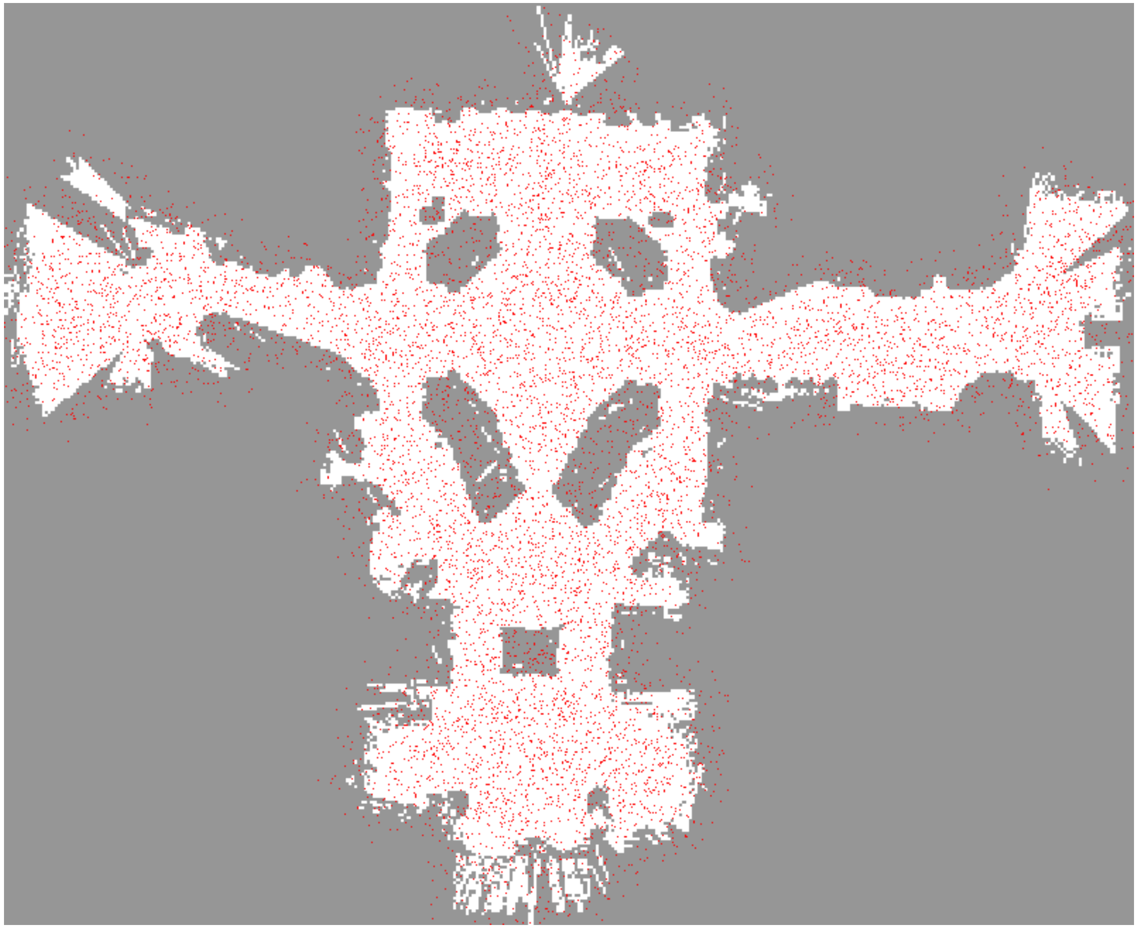


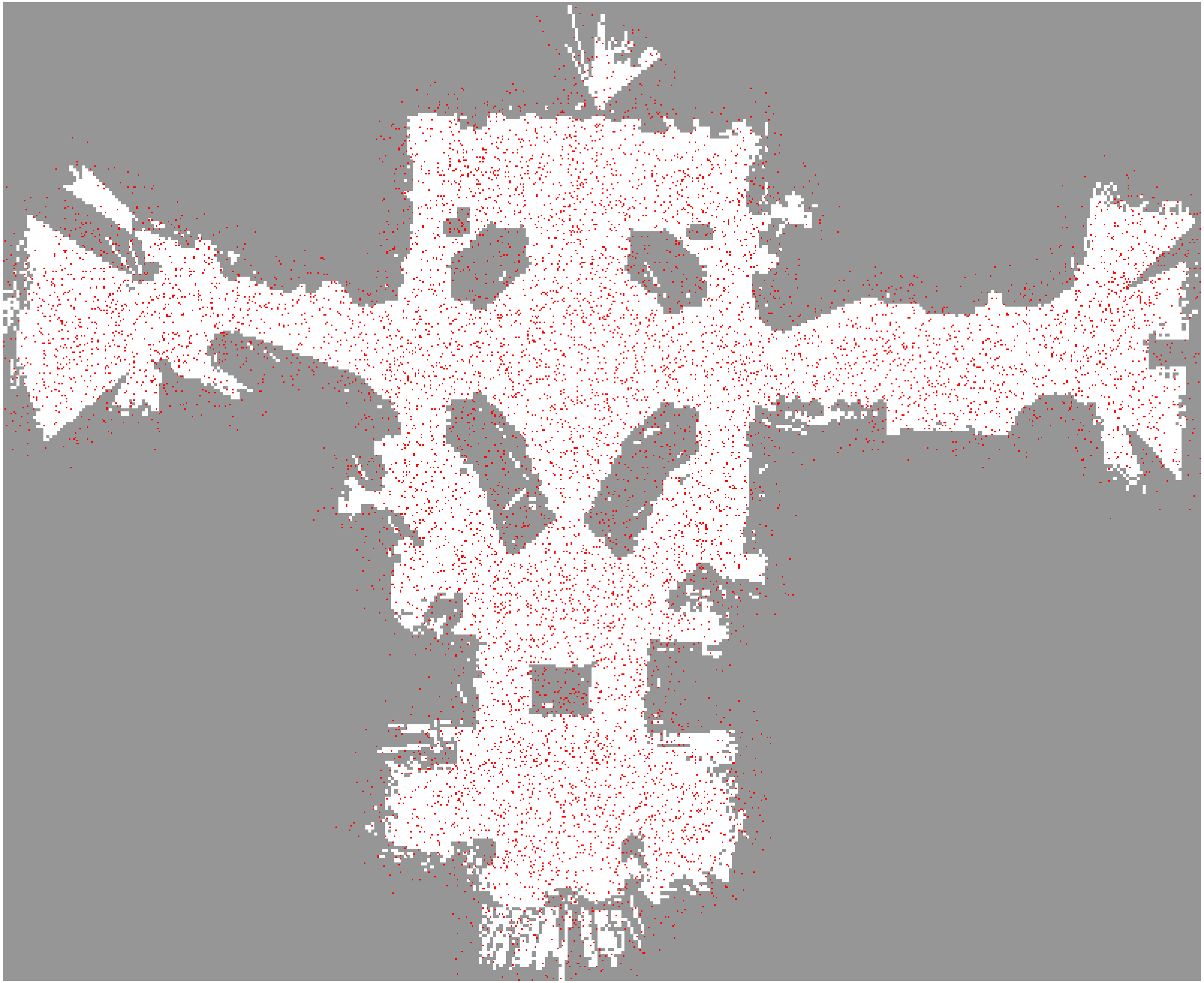
Roulette Wheel Pseudo-Code

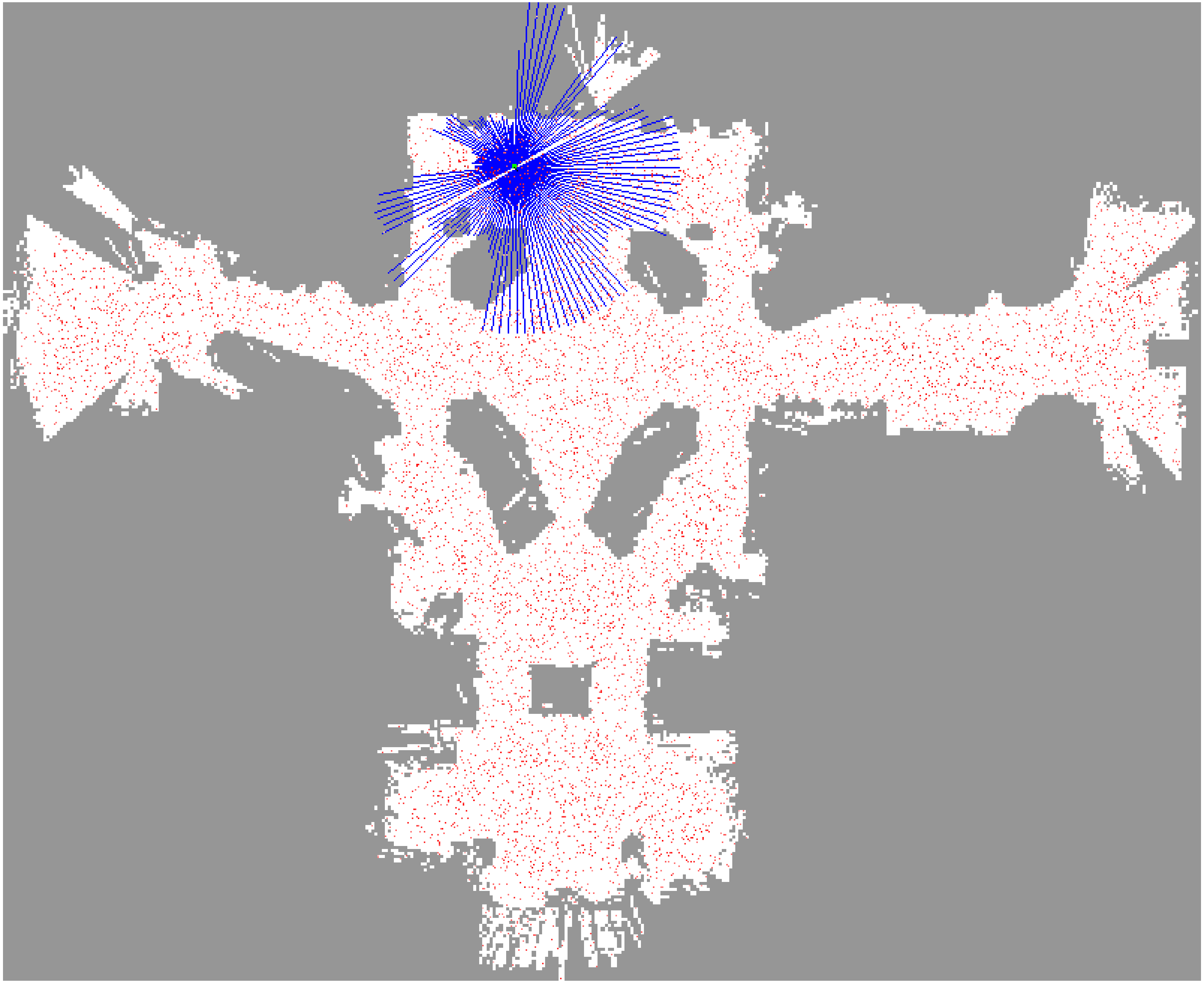

```
SELECT(particles)
total_weight ← SUM_WEIGHTS(particles)
wheel_location ← rand() * total_fitness

index ← 1
curr_sum ← WEIGHT(particles[index])
while (curr_sum < wheel_location) and (index < particles_num)
    index ← index + 1
    curr_sum ← curr_sum + WEIGHT(particles[index])
return particles[index]
```

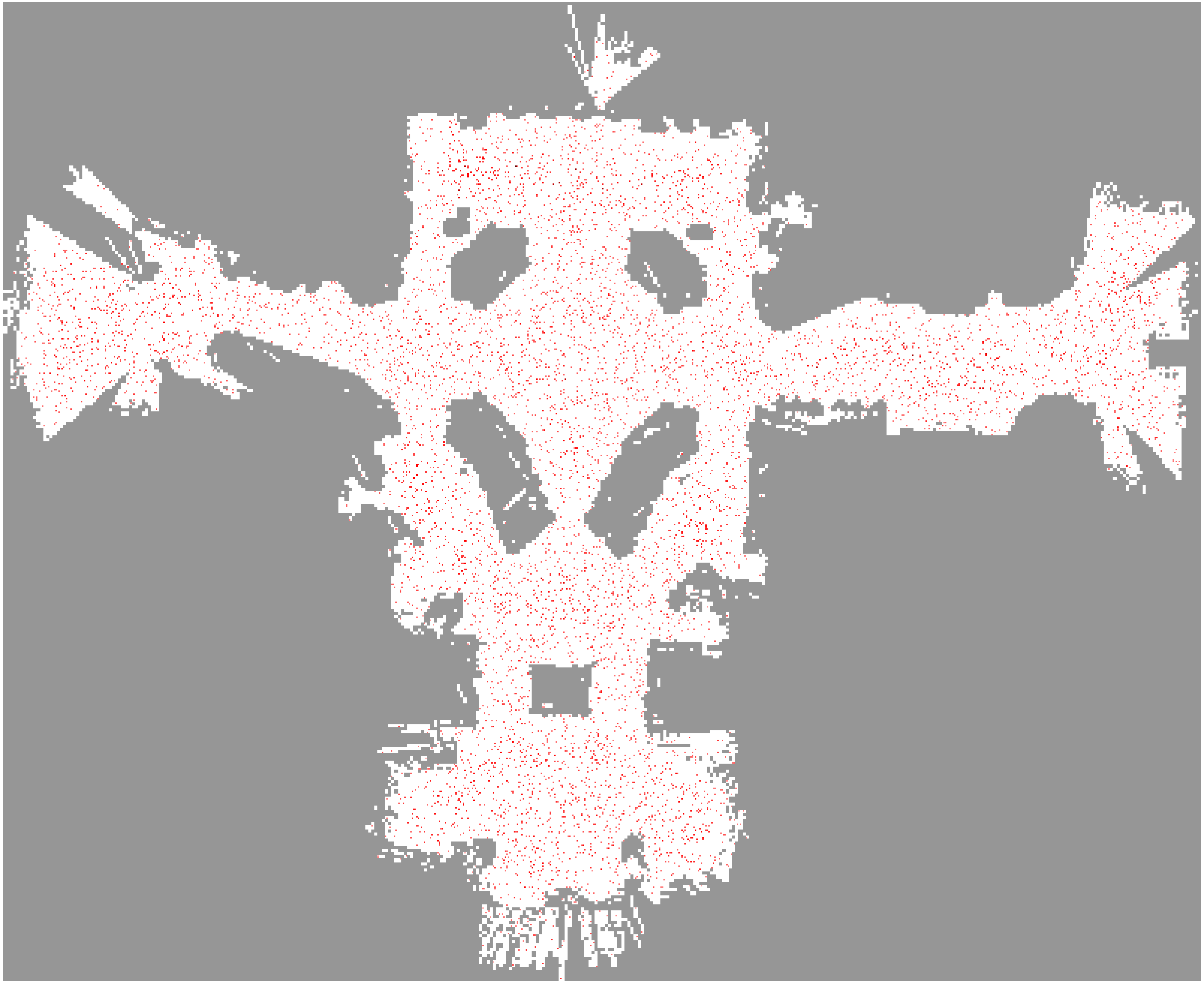
Particle Filter Example Run

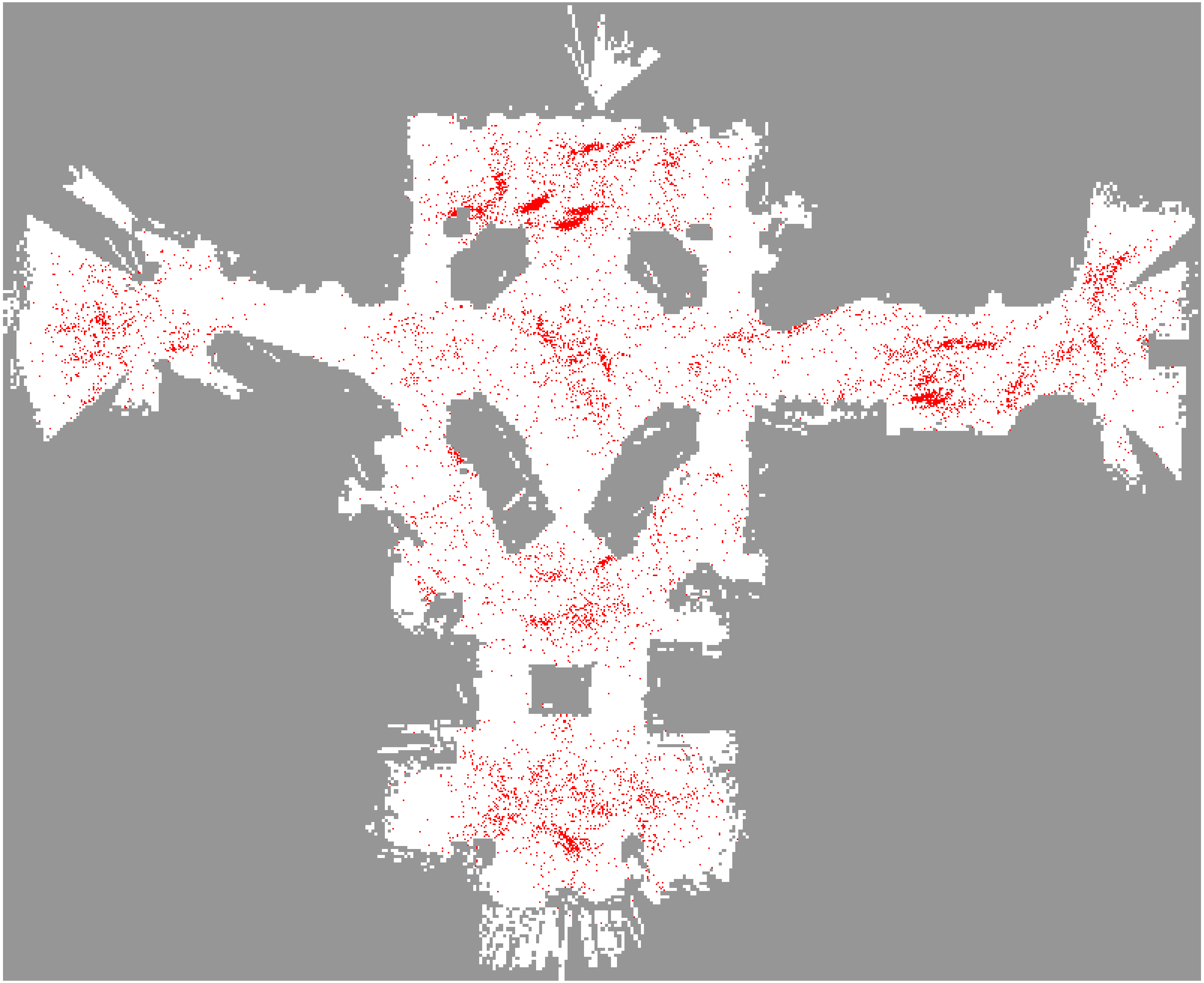
- Initial random distribution of particles:

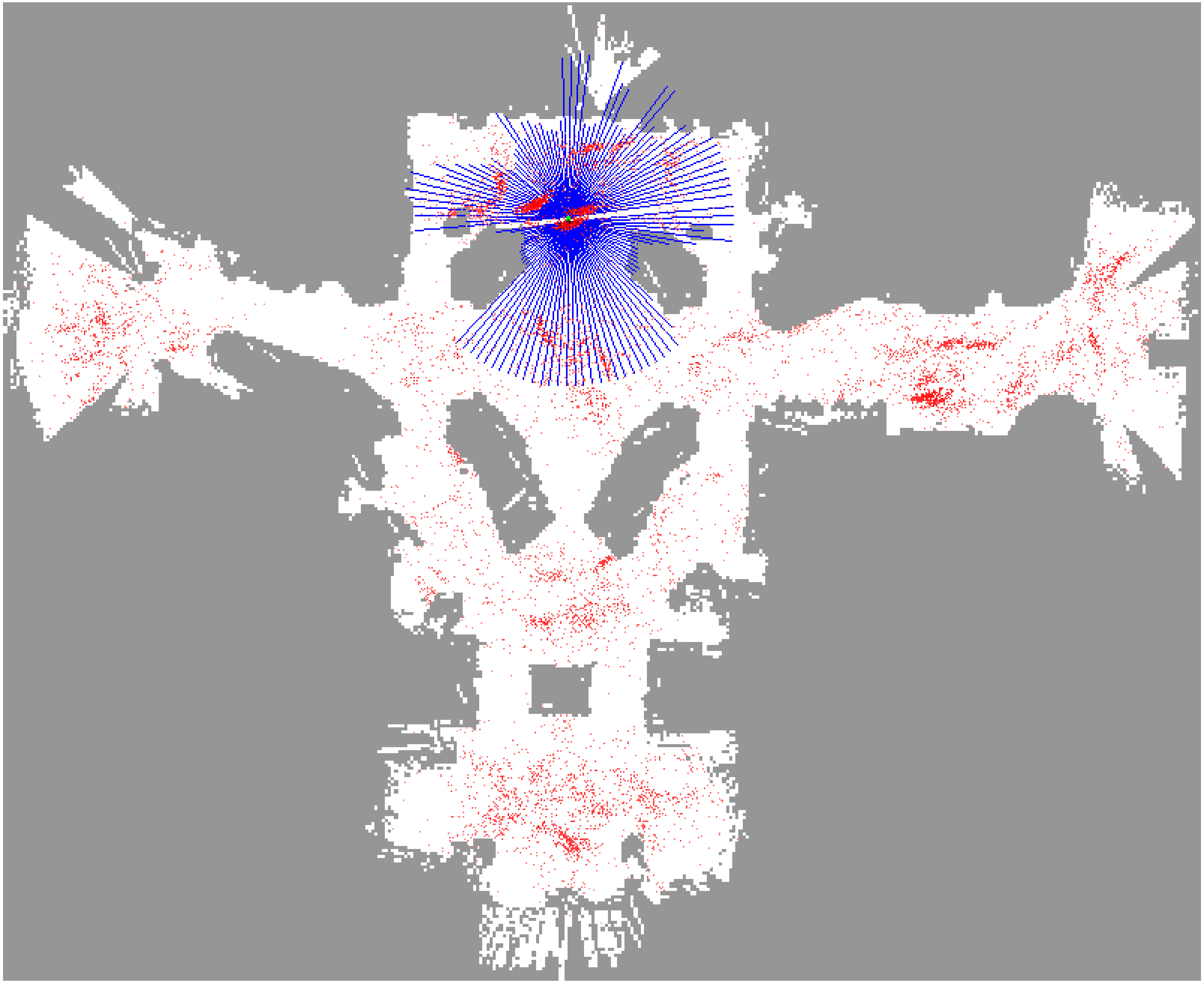


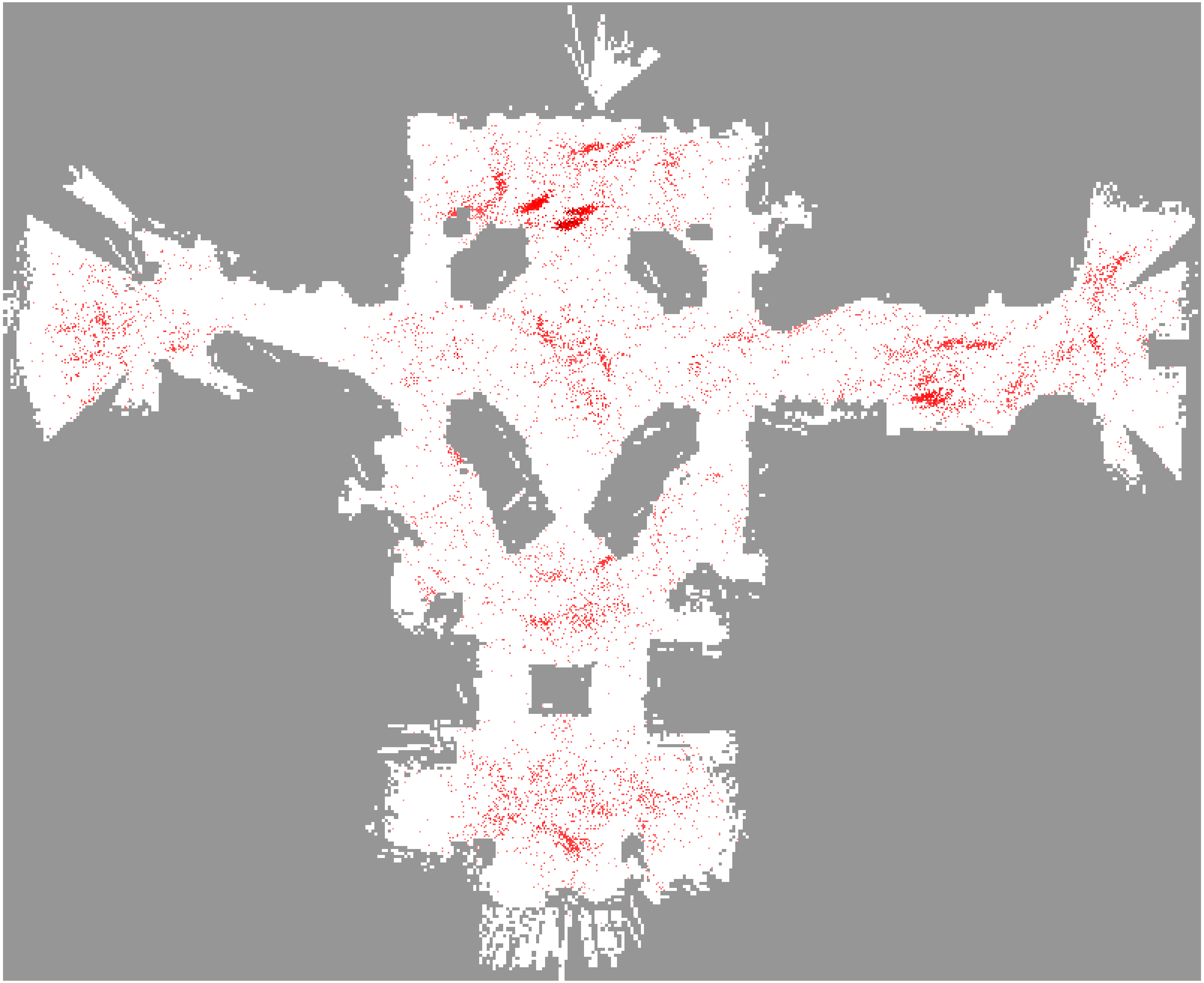


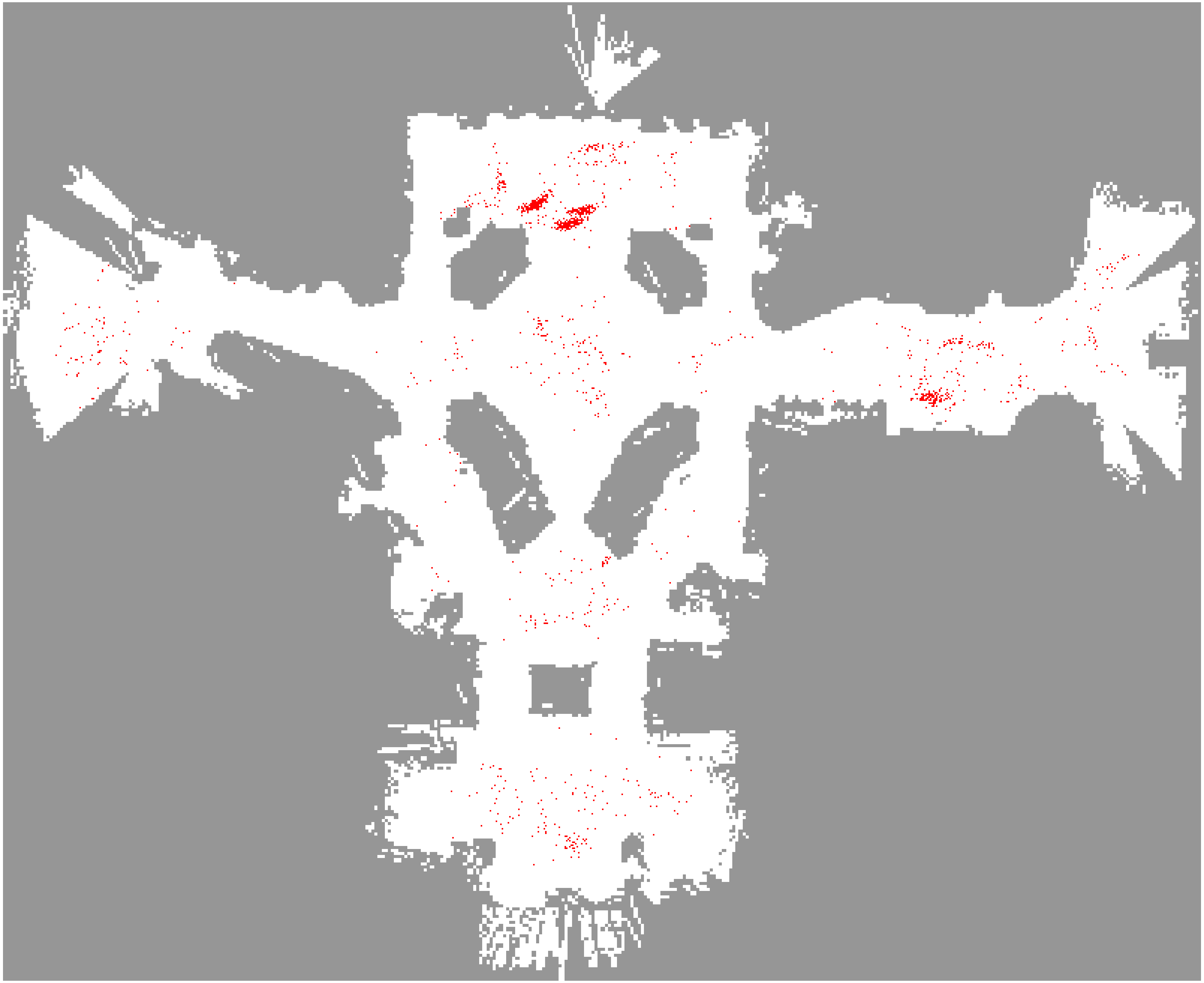


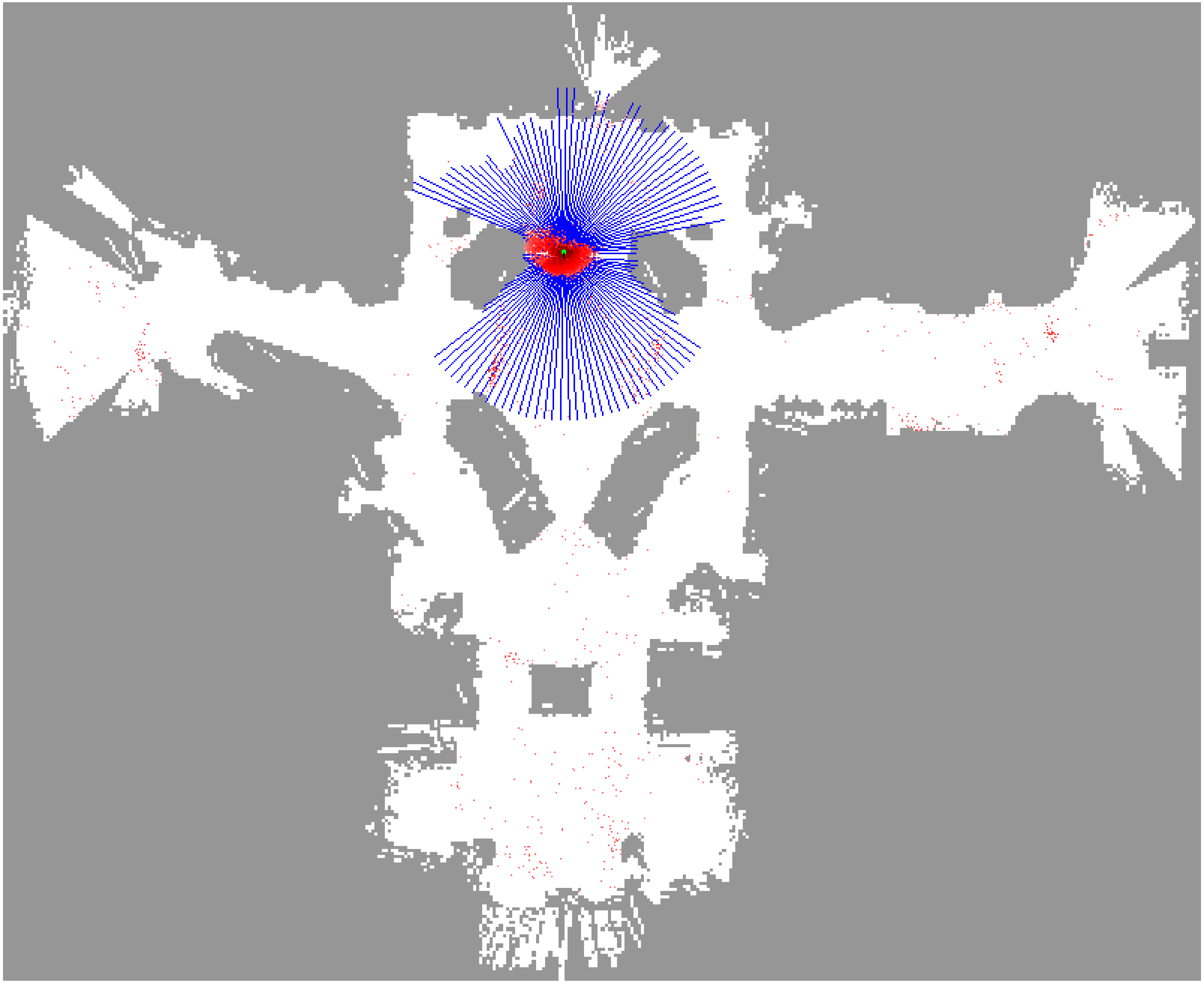


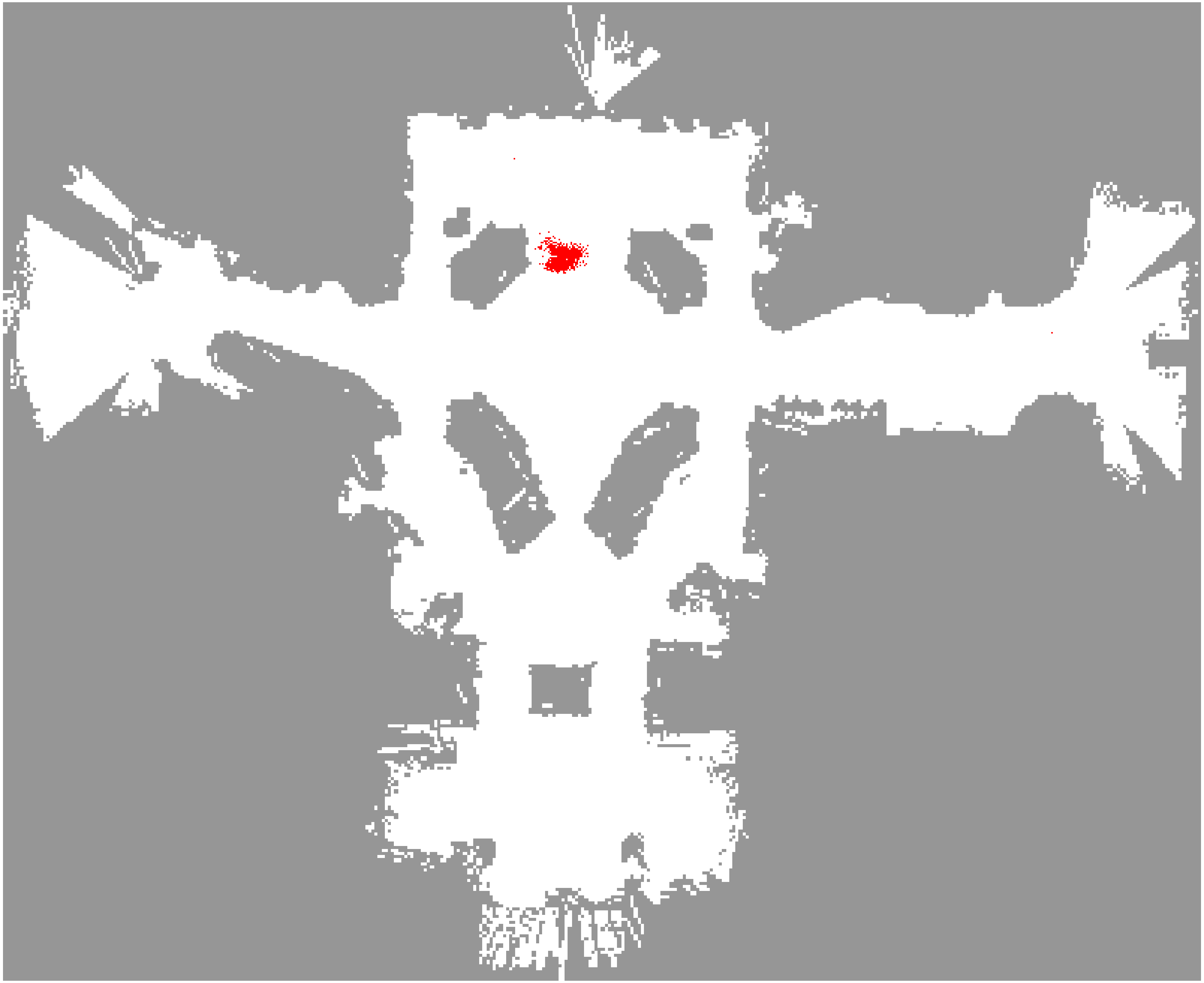


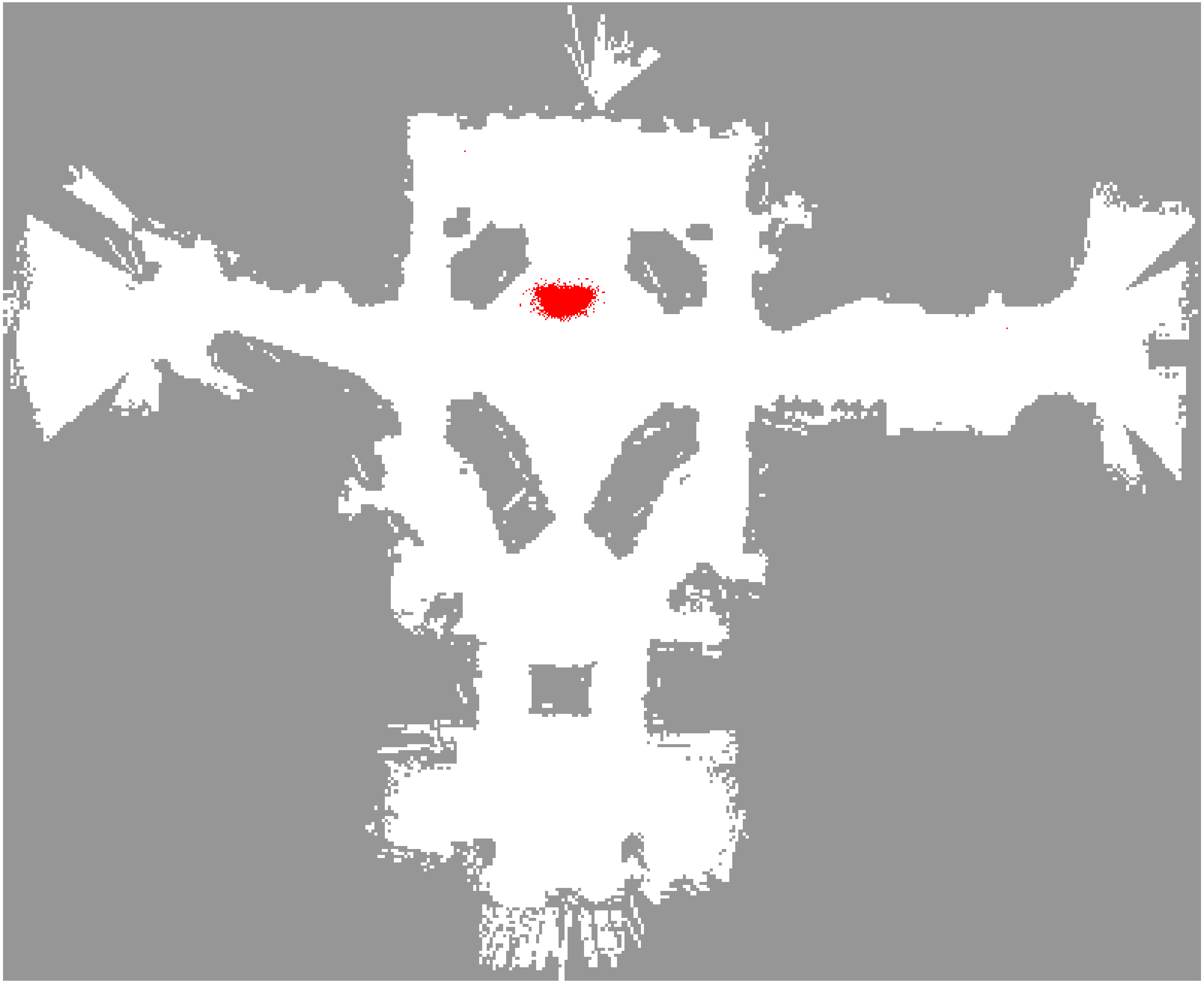


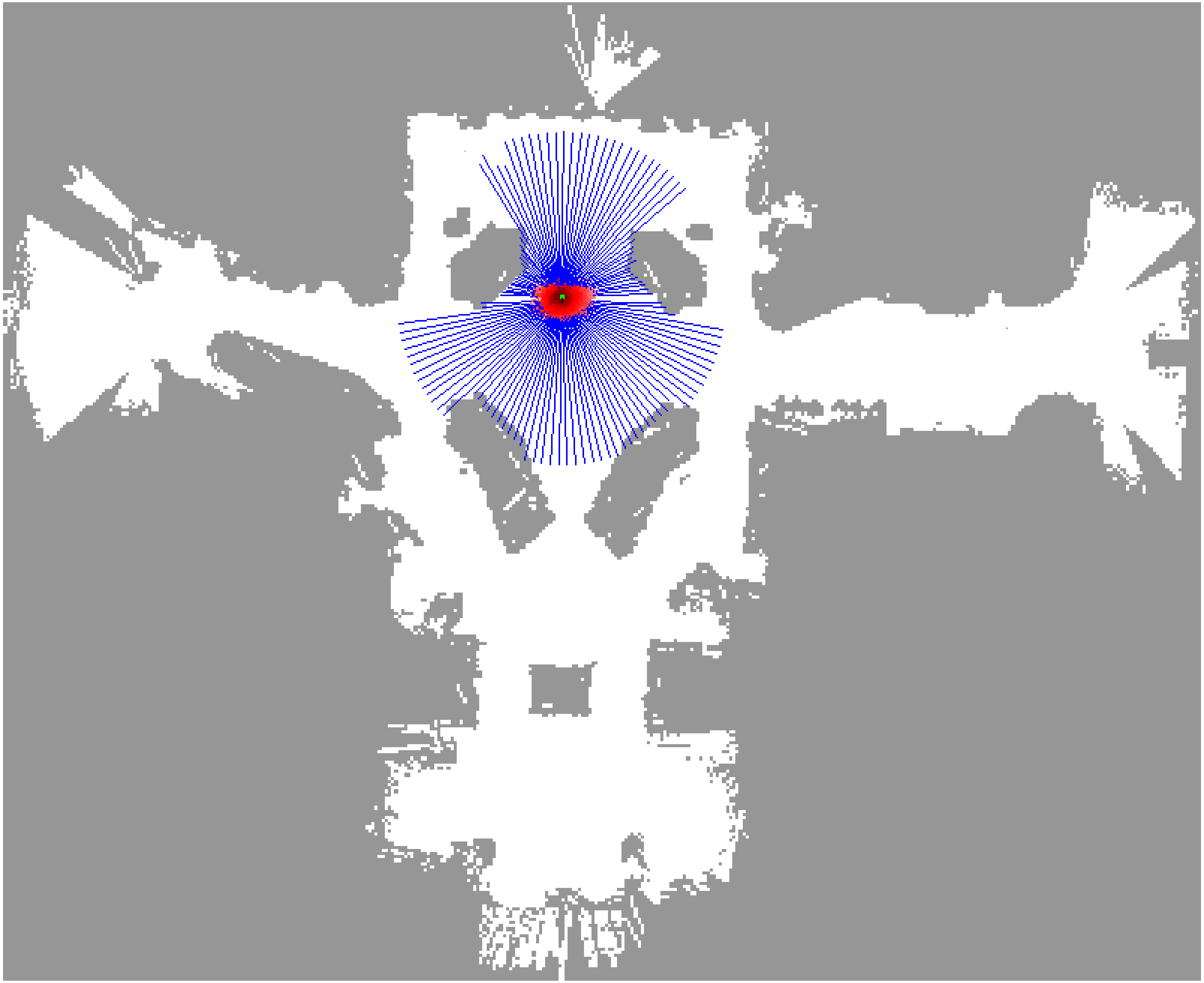


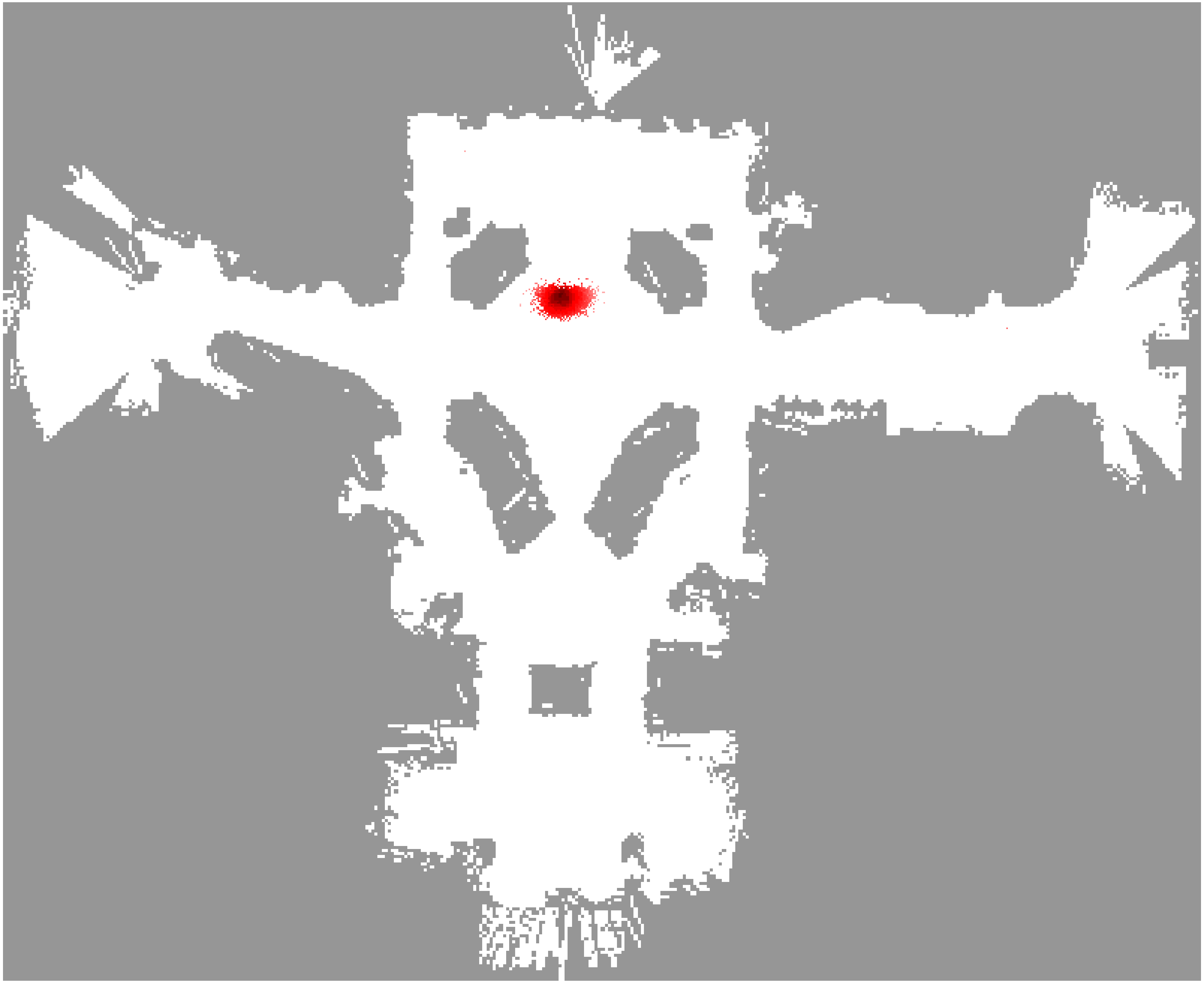


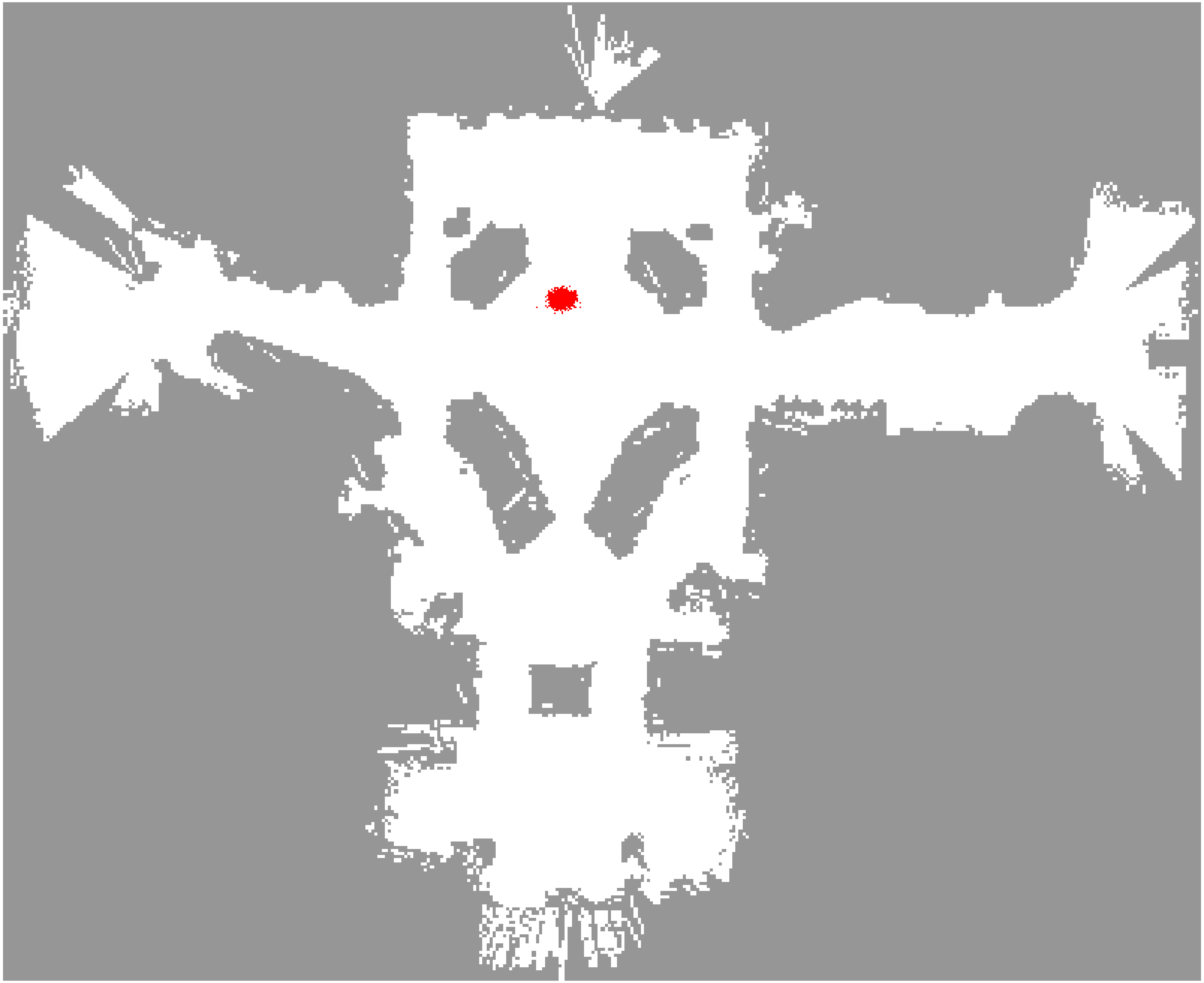


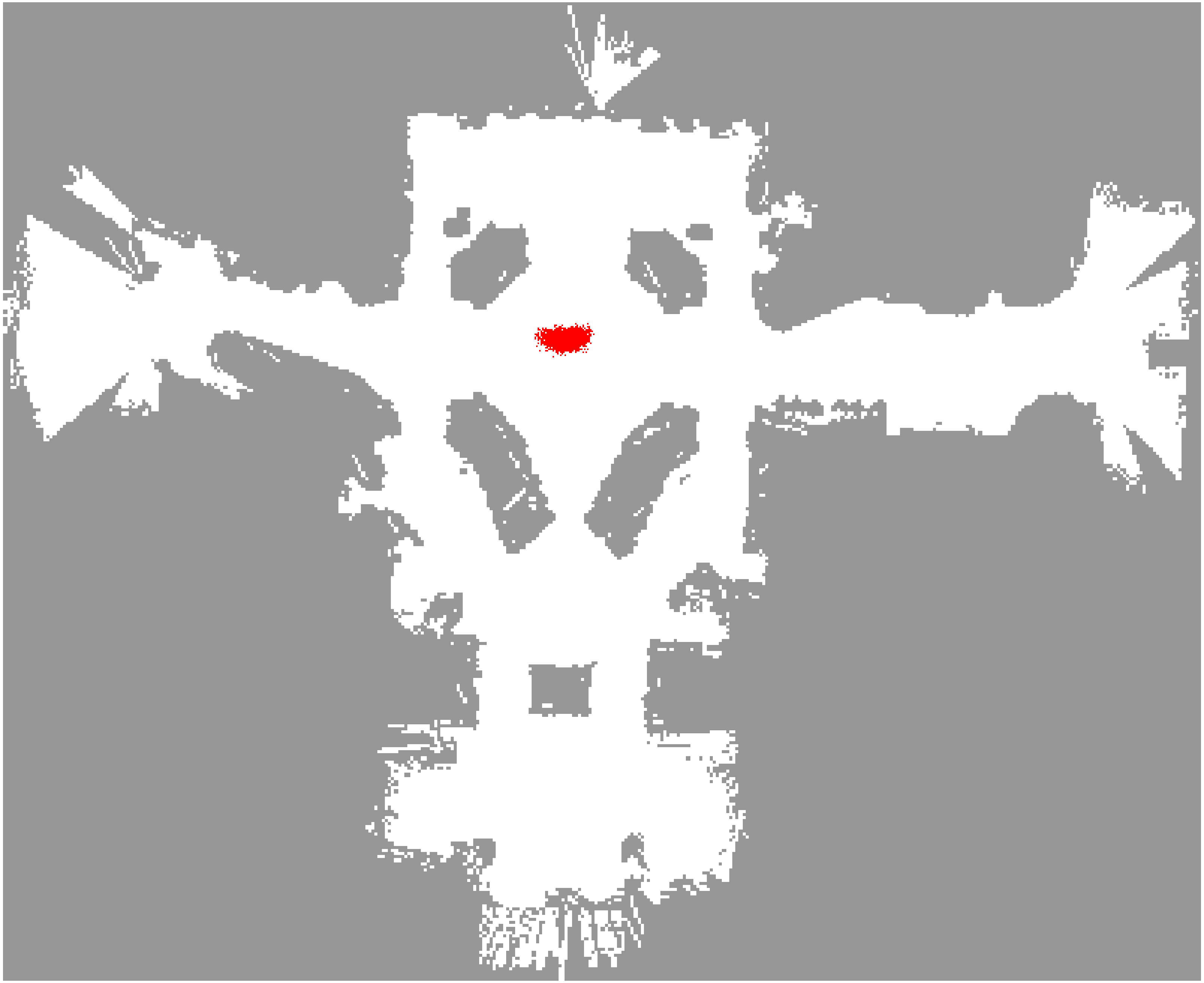












Particle Filter Example Run

- Final distribution of particles: