Spring 25 Principles of Safe Autonomy: Lecture 12: Filtering and Localization

Sayan Mitra

Reference: Probabilistic Robotics by Sebastian Thrun, Wolfram Burgard, and Dieter Fox

Slides: From the book’s website

Outline of state estimation module

Problem. Estimate the current state x_t of the system from knowledge about past observations $z_{0:t}$, control inputs $u_{0:t}$, and map m

- Introduction: Localization problem, taxonomy
- Probabilistic models: motion and measurements
- Discrete Bayes Filter
- Histogram filter and grid localization
- Particle filter

Motion and measurement models

$p_D(x_t \,|x_{0:t-1}, z_{0:t-1}, u_{1:t})$ describes motion/state evolution model

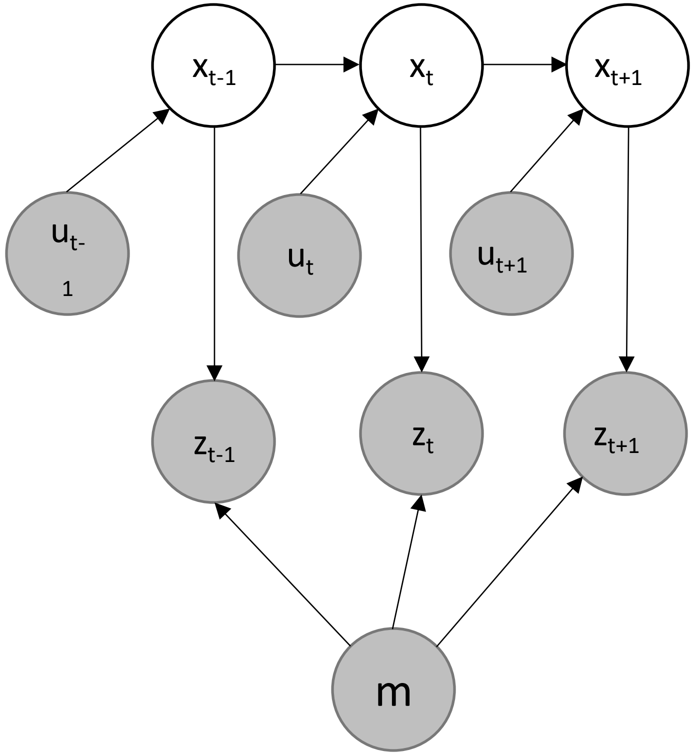
If state is complete, sufficient summary of the history then

- $p_D(x_t \,|x_{0:t-1}, z_{0:t-1}, u_{0:t-1}) = p_D(x_t \,|x_{t-1}, u_t)$ motion model
- $p_D(x'|x, u)$ if transition probabilities are time invariant

$p_M(z_t \,|x_{0:t}, z_{0:t-1}, u_{0:t-1}, m)$ describes measurement

If state is complete

- $p_M(z_t \,|x_{0:t}, z_{1:t-1}, u_{1:t}, m) = \; p(z_t \,|x_t, m)$ measurement model
- $p_M(z \,|x, m)$: time invariant measurement probability



Review of conditional probabilities

Random variable X takes values $x_1, x_2 \in \mathbb{R}^n$

$P(X = x)$ is written as $P(x)$

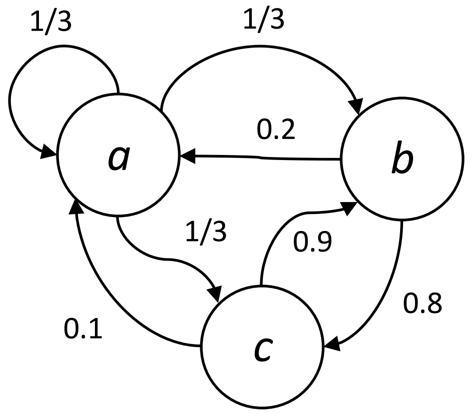
$P(X = x, Z = z)$ is written as $P(x, z)$

Conditional probability: $P(X = x \mid Z = z) = P(x|z) = \frac{P(x,z)}{P(z)}$ provided $P(z) > 0$

Bayes Rule $P(x|z) = \frac{P(Z|X)P(x)}{P(z)}$, provided $P(z) > 0$

Evolution: probabilistic Markov Chain models

A probability distribution $\pi \in P(Q)$ over a finite set of states Q can be represented by a vector $\pi \in \mathbb{R}^{|Q|}$ where $\sum \pi_i = 1$



Recall deterministic discrete transitions for automata $D: Q \to Q$

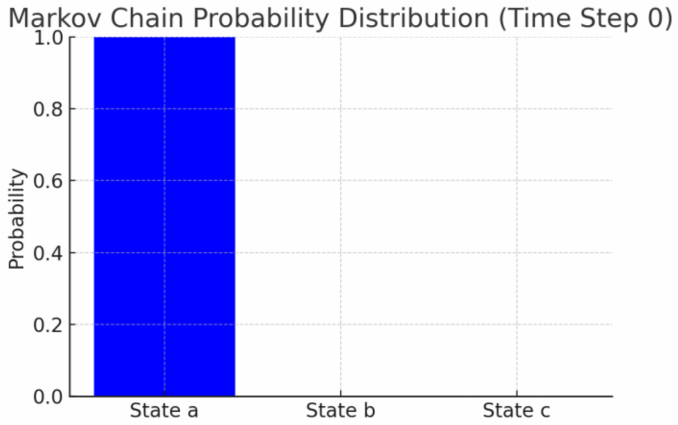
Probabilistic discrete transitions give a probability distribution $D: Q \to \boldsymbol{P}(Q)$ according to which the next state is chosen, i.e., $D(q)$ is a particular probability distribution over Q

For the example on the right $p_D(X_{t+1} = b \,| X_t = a) = \frac{1}{3}$, i.e., $D(a) = [a{:}\frac{1}{3} \;\; b{:}\frac{1}{3} \;\; c{:}\frac{1}{3}]$ $D(b) = [a{:}\frac{1}{5} \;\; b{:}0 \;\; c{:}\frac{4}{5}]$

$$\boldsymbol{D} = \begin{bmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{5} & 0 & \frac{4}{5} \\ \frac{1}{10} & \frac{9}{10} & 0 \end{bmatrix}$$

Such a state machine model is called a **Markov chain**

A probabilistic transition $\boldsymbol{D}$ can be represented by a matrix $\boldsymbol{D} \in \mathbb{R}^{|Q| \times |Q|}$ where D_{ij} gives the probability of state i to transition to j



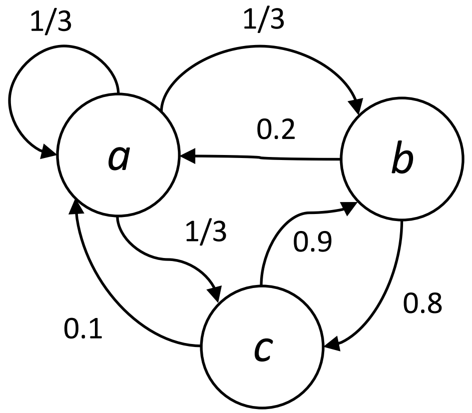
The evolution of the probability π over states can be represented as

$\pi_{t+1} = \boldsymbol{D}\pi_t$ starting with an initial distribution $\pi_0 \in \boldsymbol{P}(Q)$

Evolution and measurement: probabilistic models

Even more generally, transitions depend on outputs and history

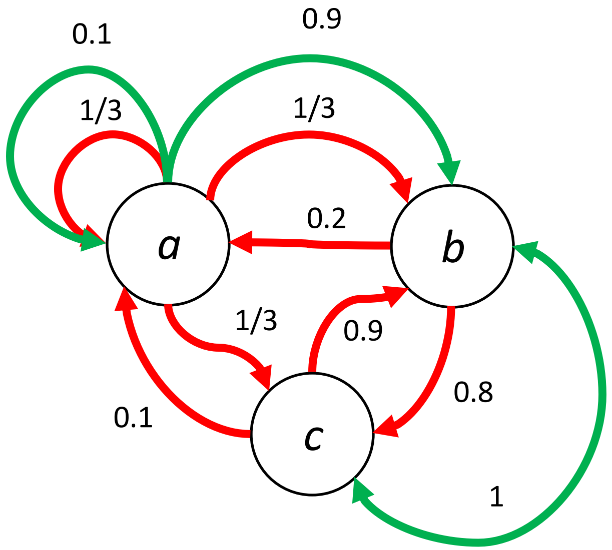
$p_D(X_t = x_t \,|X_0 = x_0, \ldots X_{t-1} = x_{t-1,} Z_1 = z_1, \ldots Z_{t-1} = z_{t-1,}, U_1 = u_1, \ldots U_t = u_{t,})$ describes state evolution model



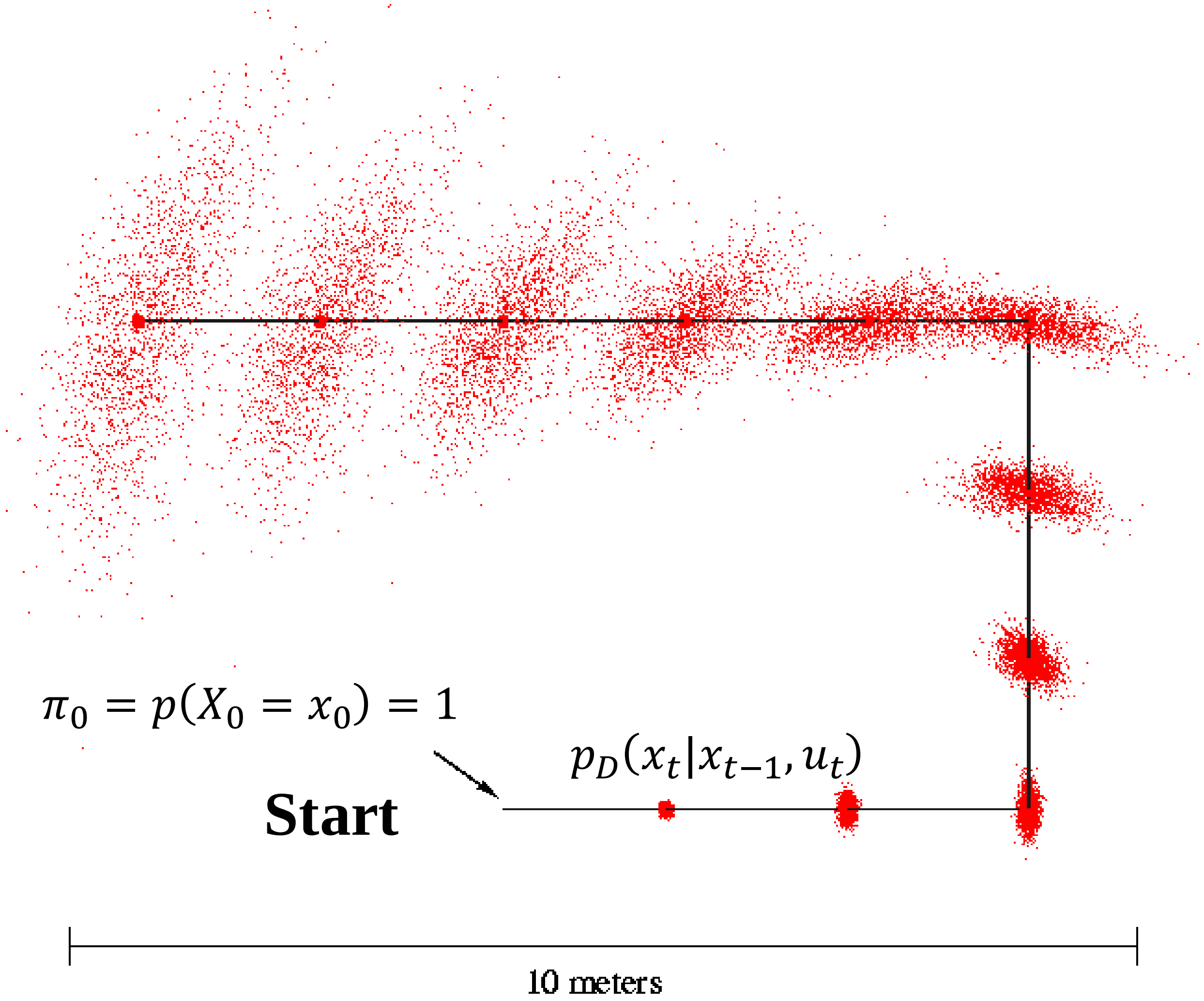
$p_D(x_t \,|x_{0:t-1}, z_{1:t-1}, u_{1:t})$ describes motion/state evolution model

If state is complete, sufficient summary of the history then

- $p_D(x_t \,|x_{0:t-1}, z_{0:t-1}, u_{0:t-1}) = p_D(x_t \,|x_{t-1}, u_t)$ transition prob.
- $p_D(x'|x,u)$ if transition probabilities are time invariant



Example Motion Model without measurements



$\pi_0 = p(X_0 = x_0) = 1$

$p_D(x_t | x_{t-1}, u_t)$

The state transition probabilities are defined by $x_{t+1} = f(x_t, u_t) + \omega_t$

where $\omega_t \sim N(0,1)$

Probabilistic measurements

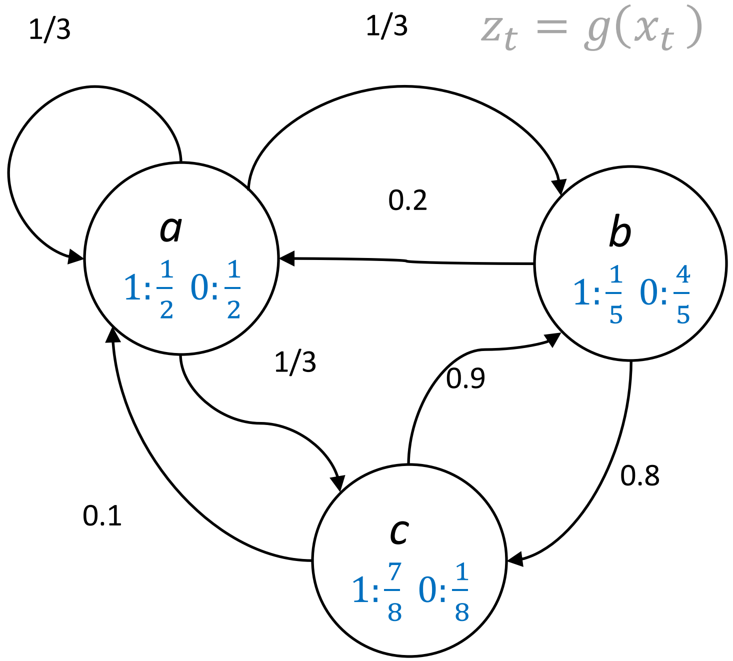
A **measurement model** gives the outputor observation probability for a given state

$$p_M(z_t = 1 | x_t = a) = \frac{1}{2}$$

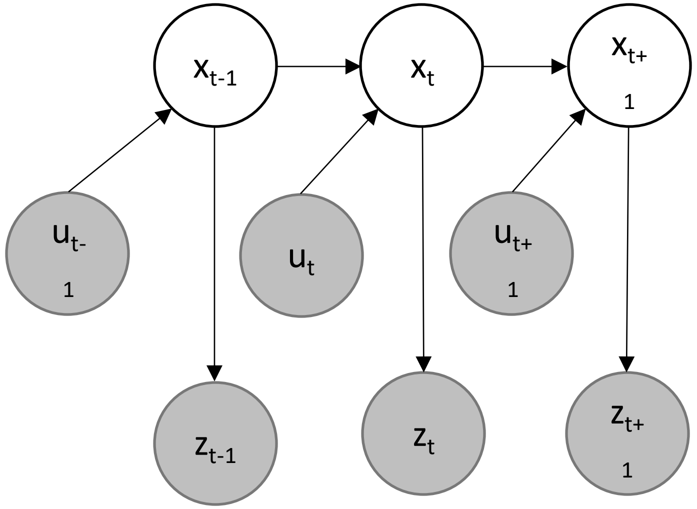
Generally, measurements can depend on history

$$p_M(z_t \,| x_{0:t}, z_{1:t-1}, u_{0:t-1})$$

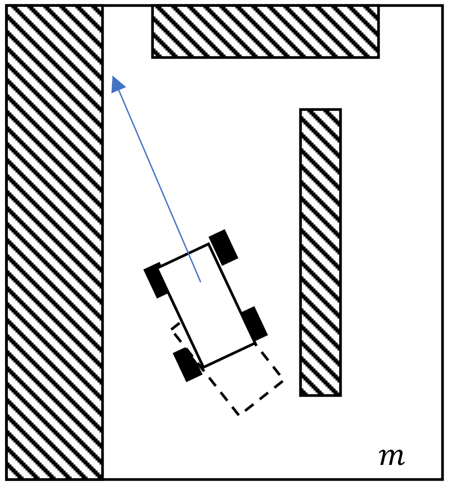
- If state is complete $p_M(z_t \,| x_{0:t}, z_{1:t-1}, u_{1:t}) = \; p(z_t \,| x_t)$
- $p_M(z_t \,| x_t)$: measurement probability
- $p_M(z \,| x)$: time invariant measurement probability



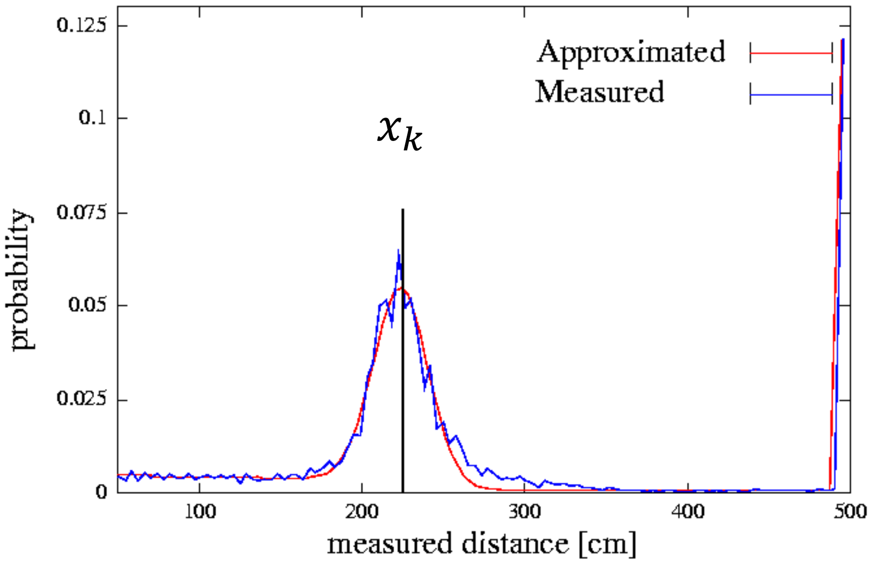
State a produces output 1 and 0 each with probability 0.5



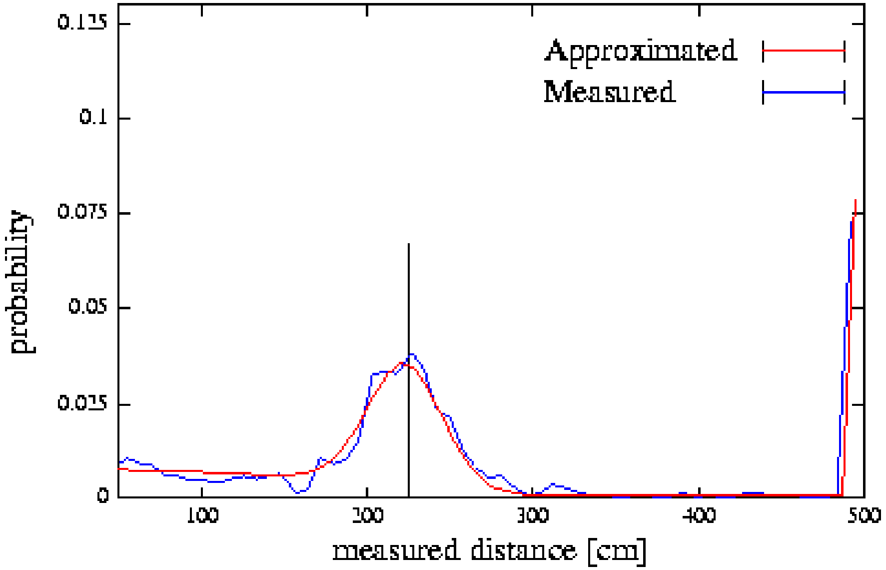
Example Proximity Sensor Measurement Models



$p_M(z_t \,|X_t = x_k)$



Laser sensor



Sonar sensor

Beliefs

Belief: Robot's knowledge about the state

True state x_t is not directly measurable or observable and the robot must infer or estimate state from measurements and this distribution of states is called the *belief*

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

Posterior distribution over state at time t given all past measurements and control. This will be calculated in two steps:

Initially: $bel(x_0) = \pi_0$

1. Prediction: $\overline{bel}(x_t) = p(x_t | z_{1:t-1}, u_{1:t})$ based on past measurements and control
2. Correction: $bel(x_t)$ from $\overline{bel}(x_t)$ based on most recent measurement z_t

Bayes Filter: Prediction and Correction

Algorithm Bayes_filter($bel(x_t), u_{t+1}, z_{t+1}$) iteratively calculates $bel(x_{t+1})$ given $bel(x_{t-1})$, the recent control u_t, and the measurement z_{t+1}

$bel(x_t): P(Q)$ is a probability distribution over Q

$\overline{bel}(x_{t+1}) = p(x_{t+1}|z_{1:t}, u_{1:t+1}) = p(x_{t+1}|u_{t+1})$ is the intermediate belief which uses only prediction but not the most recent measurement

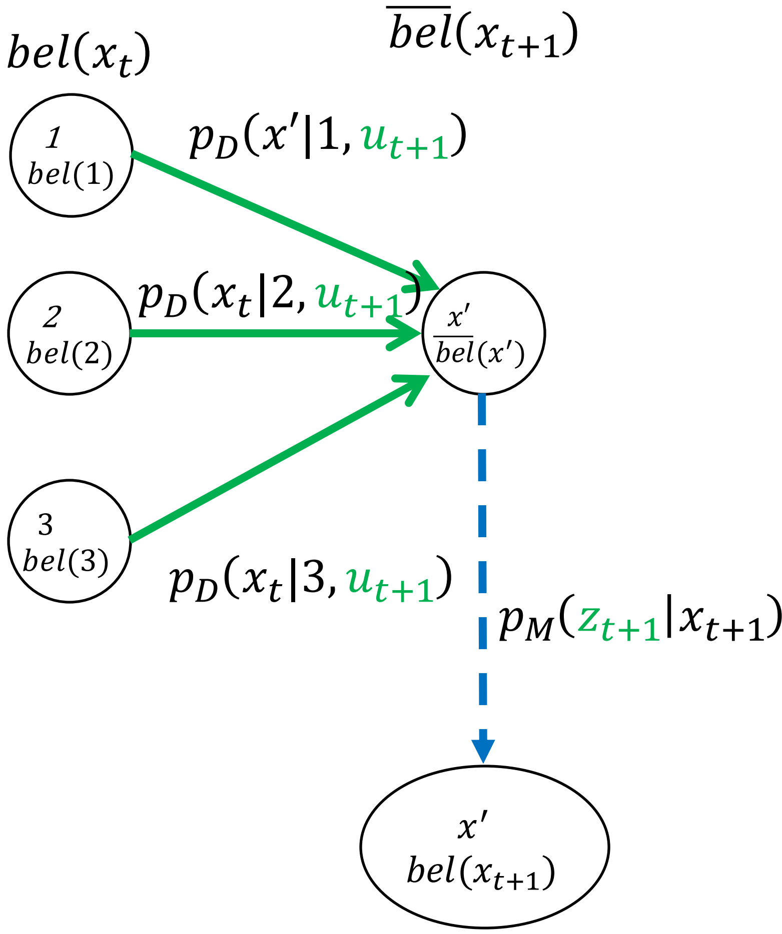
For discrete distributions for each $x' \in Q$ the beliefs can be calculated as

$$\overline{bel}(X_{t+1} = x') = \sum_{x \in Q} p_D(X_{t+1} = x'|X_t = x, U_{t+1} = u_{t+1})\, bel(X_t = x)$$

$$bel(X_{t+1} = x') = \eta\, p_M(Z_t = z_{t+1}|X_{t+1} = x)\, \overline{bel}(X_{t+1} = x)$$

where η is a normalizing constant to make $bel(x_{t+1}) \in \boldsymbol{P}(Q)$

Recall Bayes rule $P(x|z) = \frac{P(Z|X)P(x)}{P(z)}$, provided $P(z) > 0$



Histogram Filter or Discrete Bayes Filter

Finitely many states $x_i, x_k, etc.$ Random state vector X_t

$p_{k,t}$: belief at time t for state x_k; discrete probability distribution

Algorithm Discrete_Bayes_filter($\{p_{k,t-1}\}, u_t, z_t$**):**

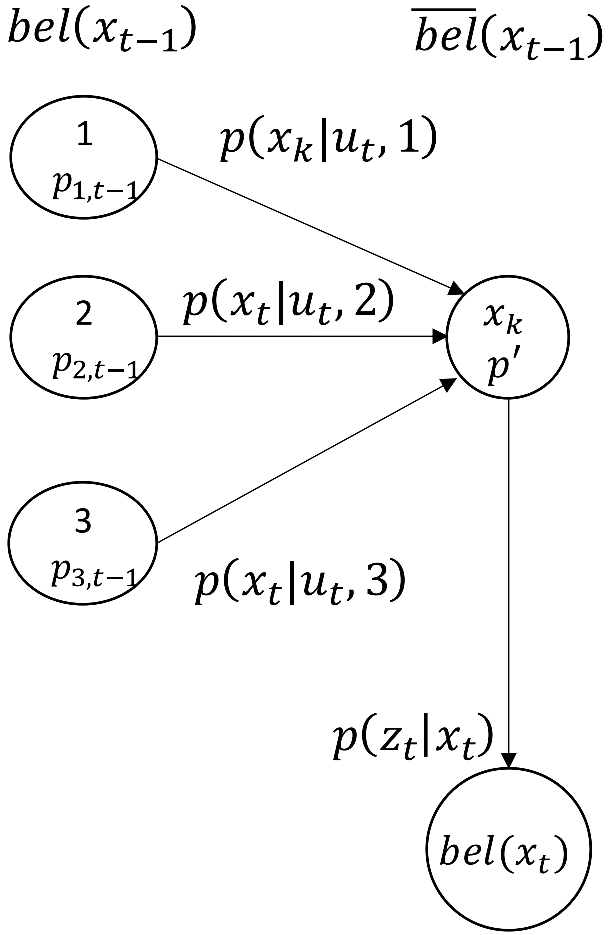
for all k do:

$$\bar{p}_{k,t} = \sum_i p(X_t = x_k | u_t, X_{t-1} = x_i) p_{i,t-1}$$

$$p_{k,t} = \eta\, p(z_t \,| X_t = x_k) \bar{p}_{k,t}$$

end for

return $\{p_{k,t}\}$



Bayes Filter: Continuous Distributions

Algorithm Bayes_filter($bel(x_{t-1}), u_t, z_t$)

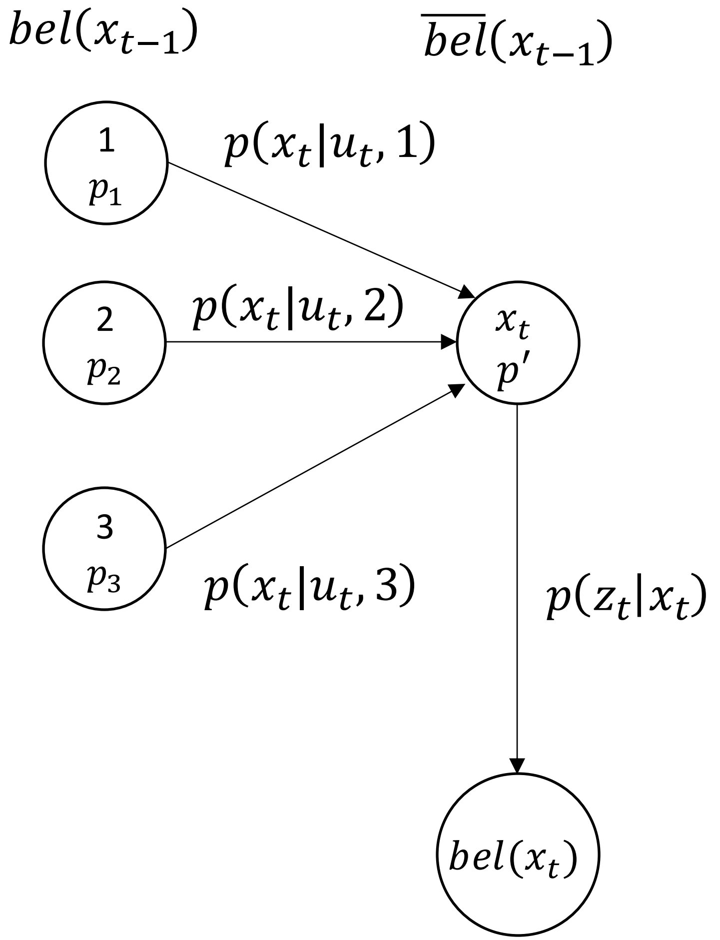
for all x_t do:

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

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

end for

return $bel(x_t)$



Grid Localization

Solves global localization in some cases kidnapped robot problem using Bayes filter

Can process raw sensor data

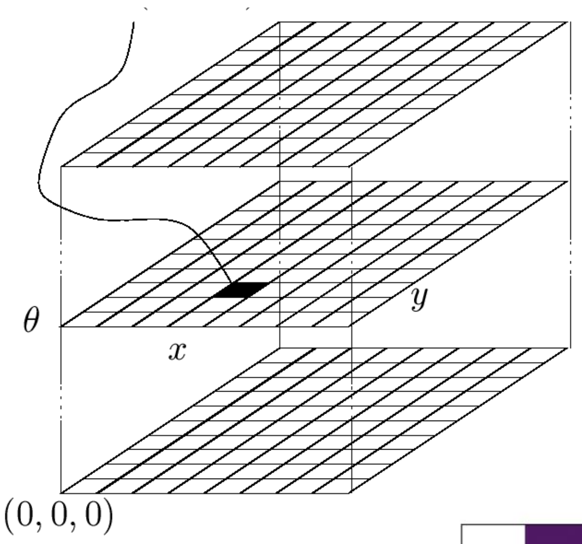
- No need for feature extraction

Non-parametric method, i.e., does not rely on specific form of probability distributions

- In particular, not bound to unimodal distributions (unlike Extended Kalman Filter)

Grid localization with bicycle model + landmarks

$bel(X_t = \langle x, y, \theta \rangle)$

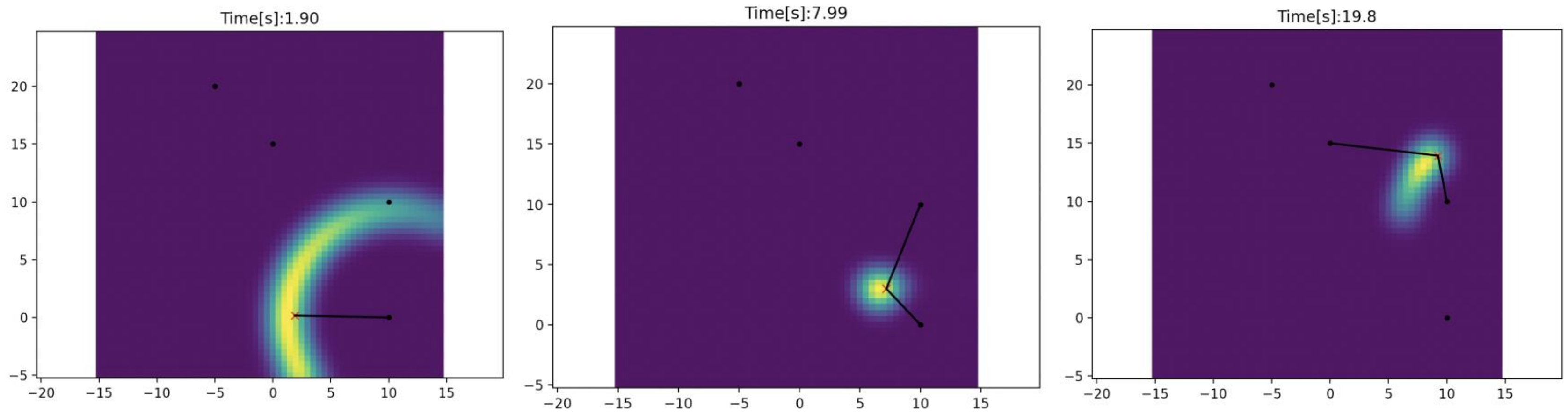


The state space Q is a quantization of position and orientation $q = \langle x, y, \theta \rangle$

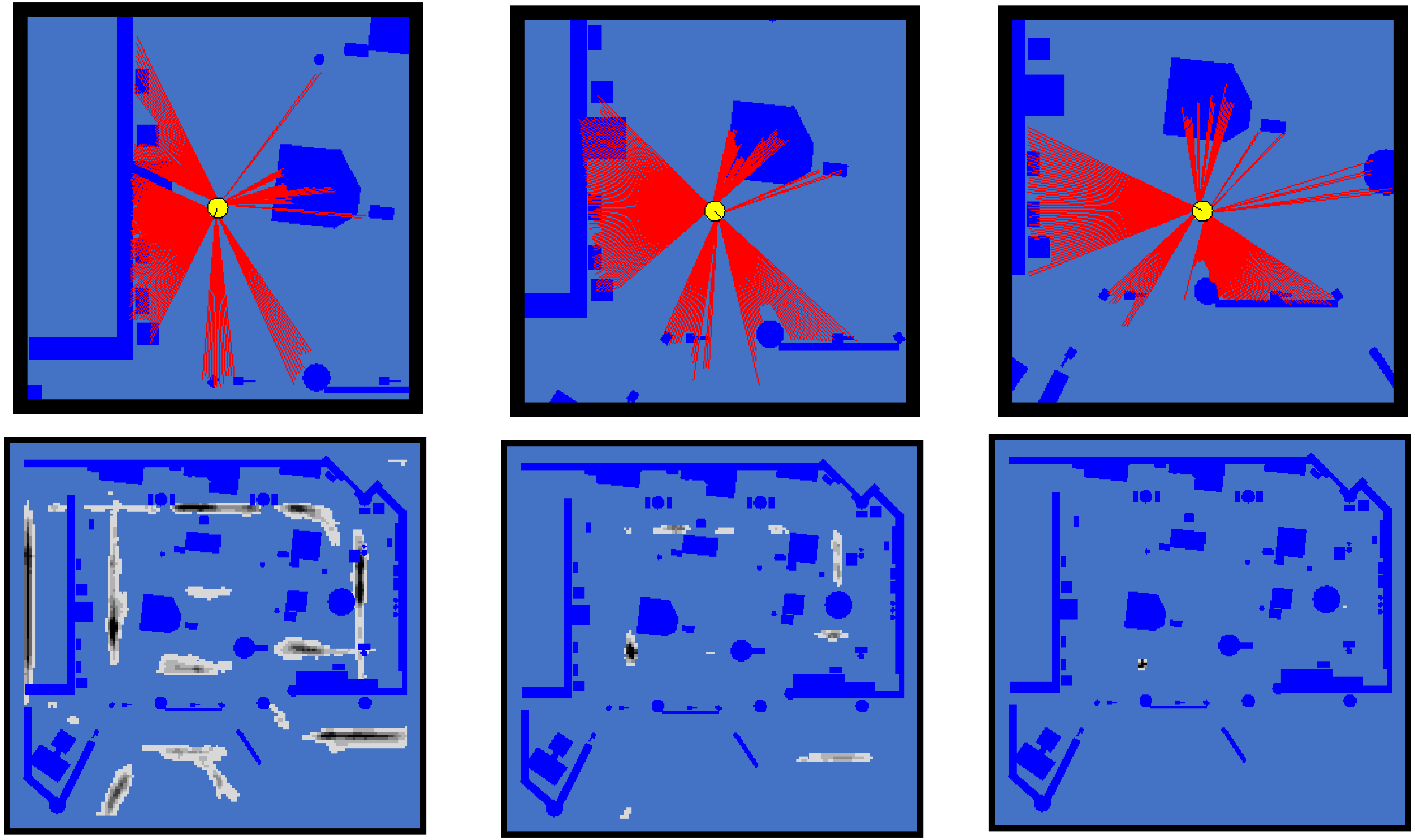
A belief is a probability distribution over states $\mathrm{bel}(q_t) \in P(Q)$

Prediction: Fixing an (steering) input u_t compute the new intermediate belief over Q using motion model $p_D(q_{t+1} | q_t, u_{t+1})$

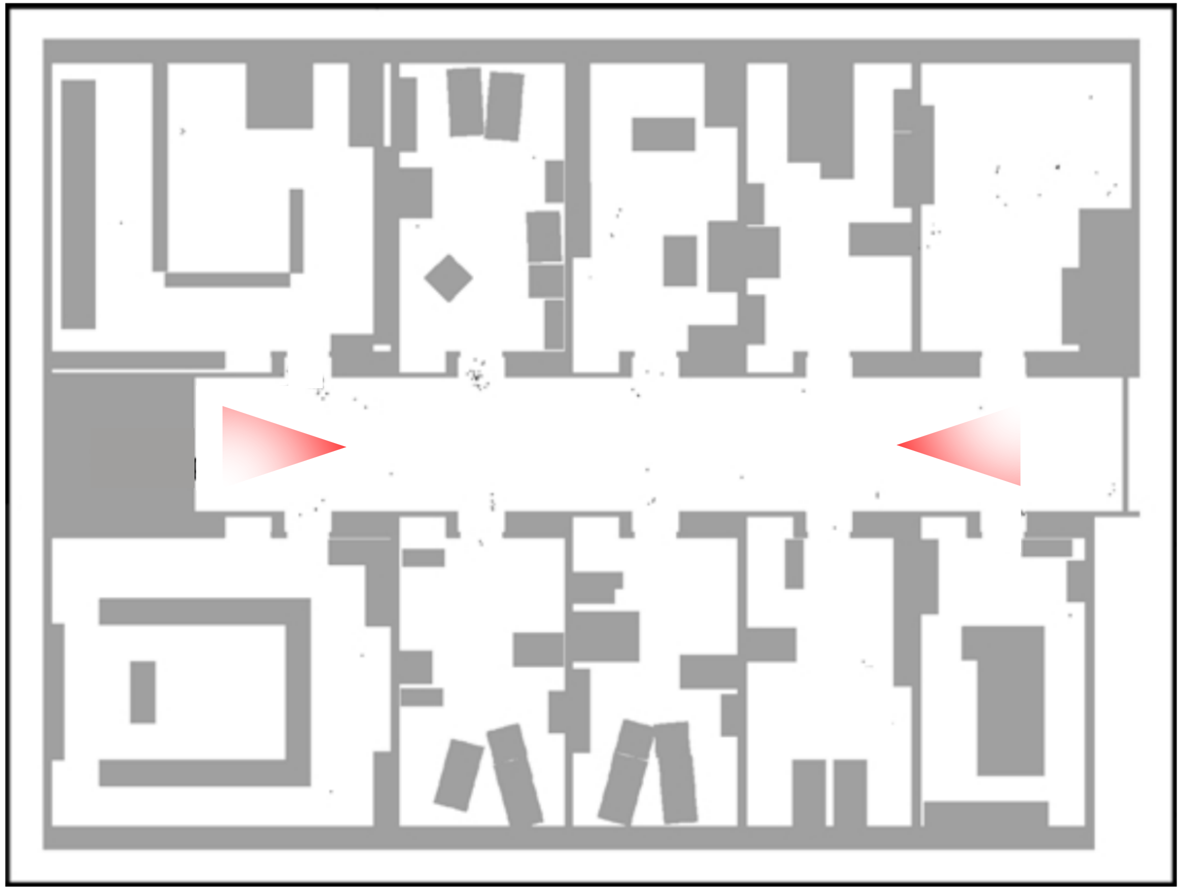
Correction: Update intermediate belief with received distance to landmark z_{t+1} based on measurement model p_M



Grid-based Localization



Ambiguity in global localization arising from locally symmetric environment



Grid localization

Algorithm Grid_localization ($\{p_{k,t-1}\}, u_t, z_t, m$)

for all k do:

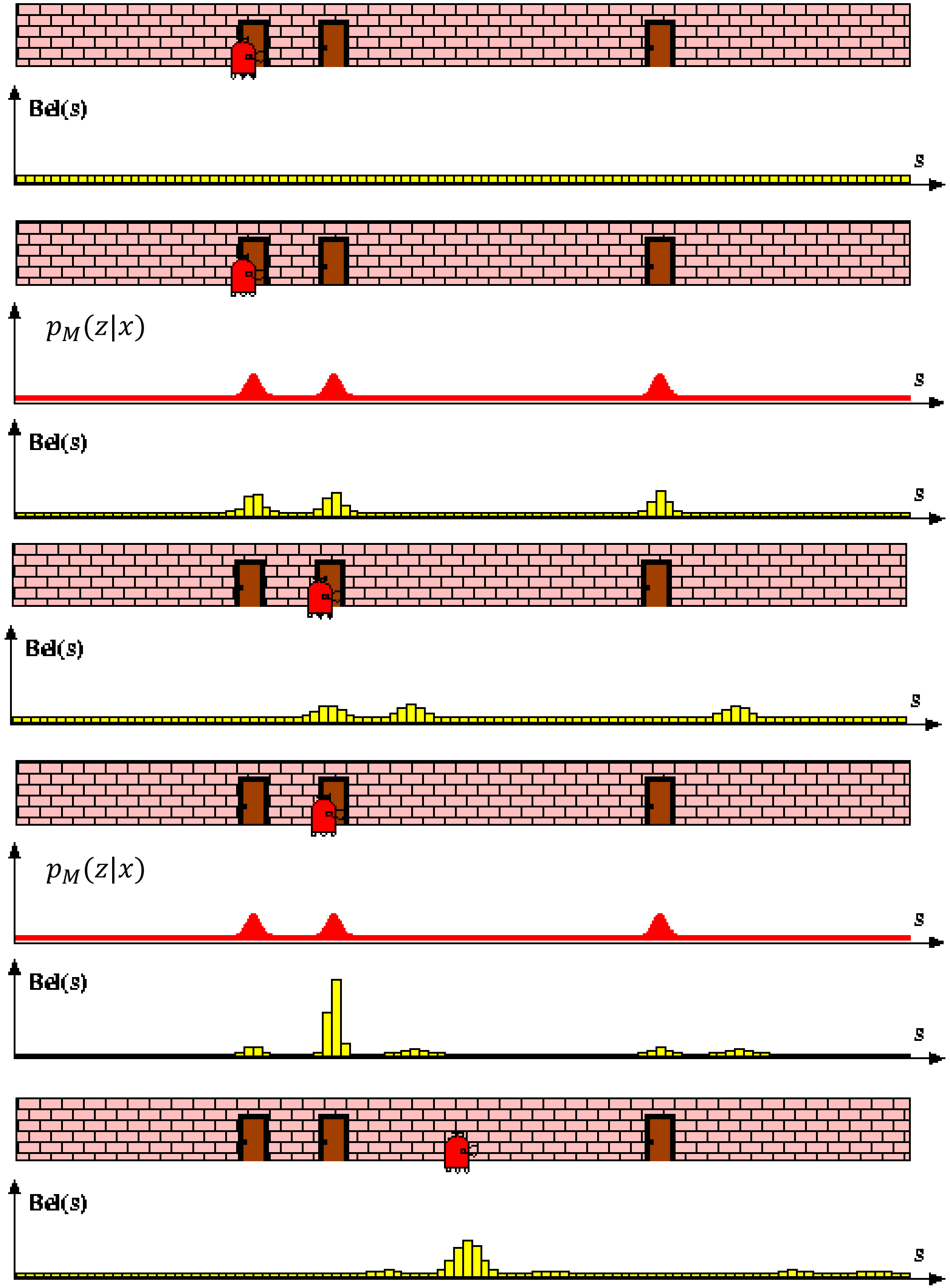
$$\bar{p}_{k,t} = \sum_i p_{i,t-1}\, \boldsymbol{motion\_model}(mean(x_k), u_t, mean(x_i))$$

$$p_{k,t} = \eta\, \bar{p}_{k,t} \boldsymbol{measurement\_model}(z_t, mean(x_k), m)$$

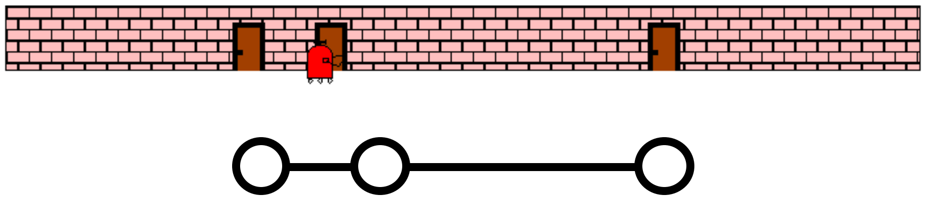
end for

return $bel(x_t)$

Grid localization, $bel(x_t)$ represented by a histogram over grid

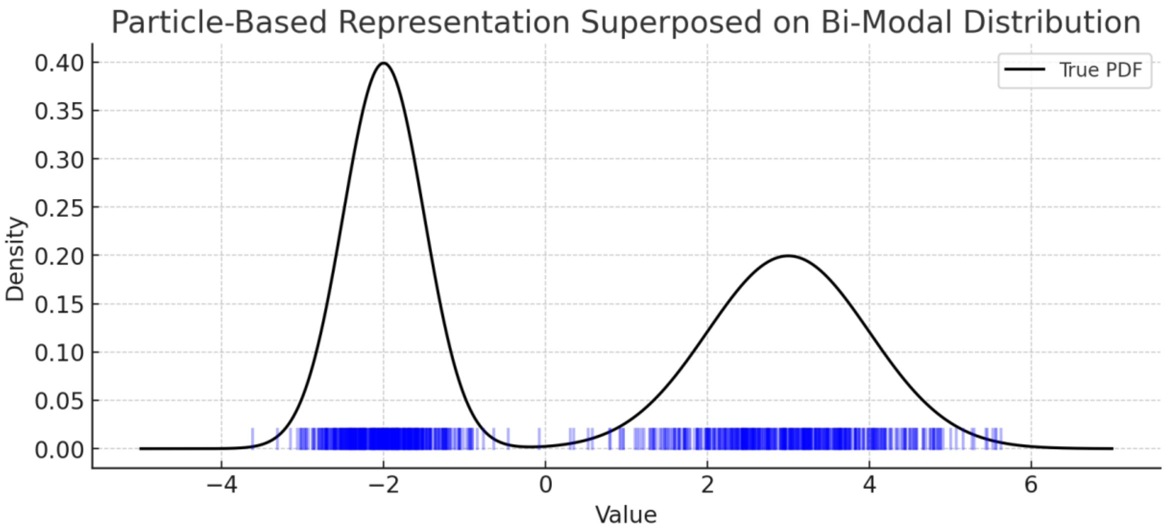


Summary



- Key variable: Grid resolution
- Two approaches
  - Topological: break-up pose space into regions of significance (landmarks)
  - Metric: fine-grained uniform partitioning; more accurate at the expense of higher computation costs
- Important to compensate for coarseness of resolution
  - Evaluating measurement/motion based on the center of the region may not be enough. *If motion is updated every 1s, robot moves at 10 cm/s, and the grid resolution is 1m, then naïve implementation will not have any state transition!*
- Computation
  - Motion model update for a 3D grid required a 6D operation, measurement update 3D
  - With fine-grained models, the algorithm cannot be run in real-time
  - Some calculations can be cached (ray-casting results)

Particle Filters



- Belief represented by finite number of parameters or particles
- Advantages
  - The representation is approximate and **nonparametric** and therefore can represent a broader set of distributions e.g., bimodal distributions
  - Can handle nonlinear transformations, e.g., under motion and measurements
- Related ideas: Monte Carlo filter, Survival of the fittest, Condensation, Bootstrap filter, Filtering: [Rubin, 88], [Gordon '93], [Kitagawa 96], Dynamic Bayesian Networks: [Kanazawa '95]

Particle filtering algorithm

$X_t = x_t^{[1]}, x_t^{[2]}, \dots x_t^{[M]}$ particles

Algorithm Particle_filter(X_{t-1}, u_t, z_t):

$\bar{X}_{t-1} = X_t = \emptyset$

for all m in [M] do:

sample $x_t^{[m]} \sim p_D(x_t | u_t, x_{t-1}^{[m]})$

$w_t^{[m]} = p_M\left(z_t \middle| x_t^{[m]}\right)$

$\bar{X}_t = \bar{X}_t + \langle x_t^{[m]}, w_t^{[m]} \rangle$

for all m in [M] do:

draw $i\ with\ probability \propto w_t^{[i]}$

add $x_t^{[i]}\ to\ X_t$

return X_t

ideally, $x_t^{[m]}$ is selected with probability prop. to $p(x_t \mid z_{1:t}, u_{1:t})$

$\bar{X}_{t-1}$ is the temporary particle set

// sampling from state transition dist.

// calculates *importance factor* w_t or weight

// resampling or importance sampling; these are distributed according to $\eta\, p\left(z_t \middle| x_t^{[m]}\right) \overline{bel}(x_t)$

// survival of fittest: moves/adds particles to parts of the state space with higher probability

Importance Sampling

suppose we want to compute $E_f[I(x \in A)]$ but we can only sample from density g

$E_f[I(x \in A)]$

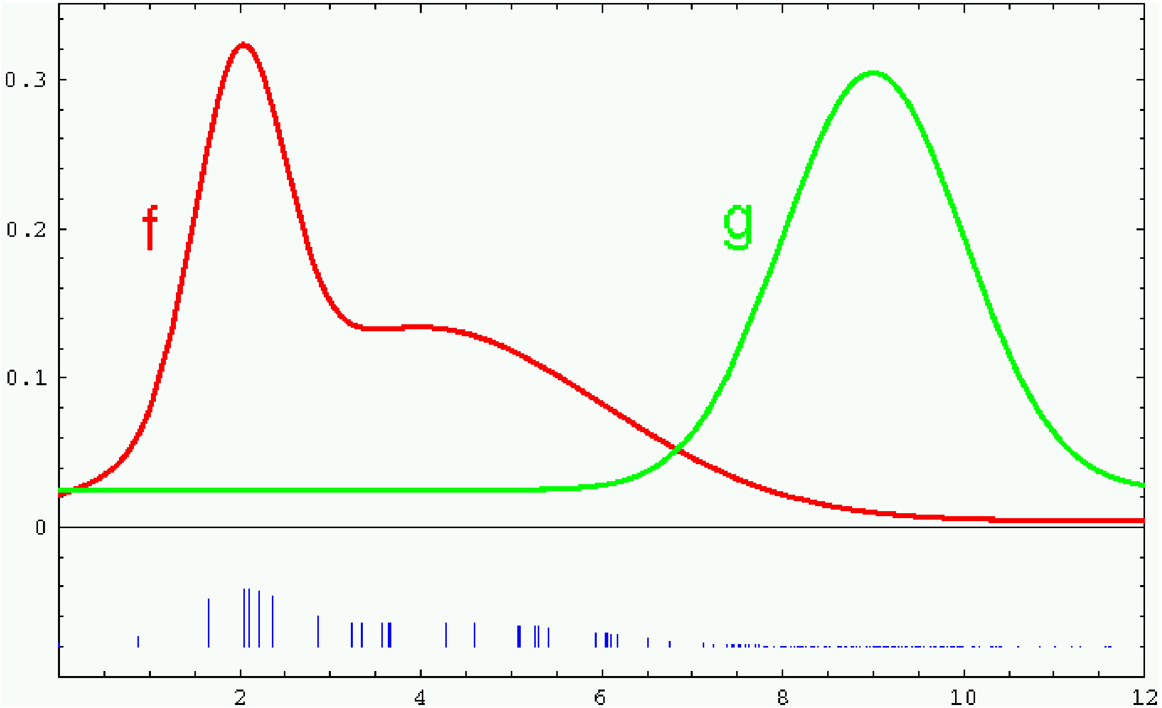
$= \int f(x) I(x \in A) dx$

$= \int \frac{f(x)}{g(x)} g(x) I(x \in A) dx$, provided $g(x) > 0$

$= \int w(x) g(x) I(x \in A) dx$

$= E_g[w(x) I(x \in A)]$

We need $f(x) > 0 \Rightarrow g(x) > 0$



Weight samples: $w = f / g$

Monte Carlo Localization (MCL)

$X_t = x_t^{[1]}, x_t^{[2]}, \dots x_t^{[M]}$ particles

Algorithm MCL(X_{t-1}, u_t, z_t,m):

$\bar{X}_{t-1} = X_t = \emptyset$

for all m in [M] do:

$x_t^{[m]} = \boldsymbol{sample\_motion\_model}(u_t\ x_{t-1}^{[m]})$

$w_t^{[m]} = \boldsymbol{measurement\_model}(z_t, x_t^{[m],m})$

$\bar{X}_t = \bar{X}_t + \langle\, x_t^{[m]}, w_t^{[m]} \rangle$

for all m in [M] do:

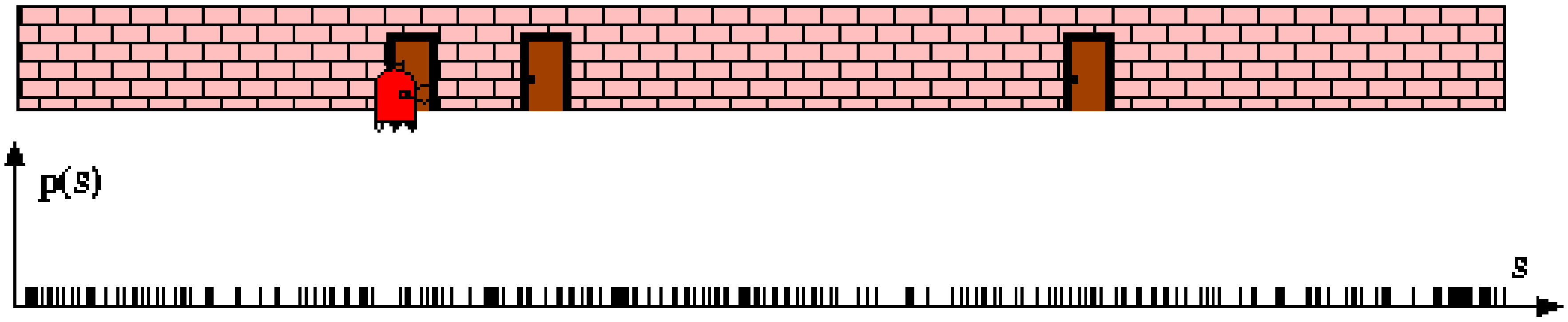
draw $i\ with\ probability\ \propto w_t^{[i]}$

add $x_t^{[i]}\ to\ X_t$

return X_t

Plug in motion and measurement models in the particle filter

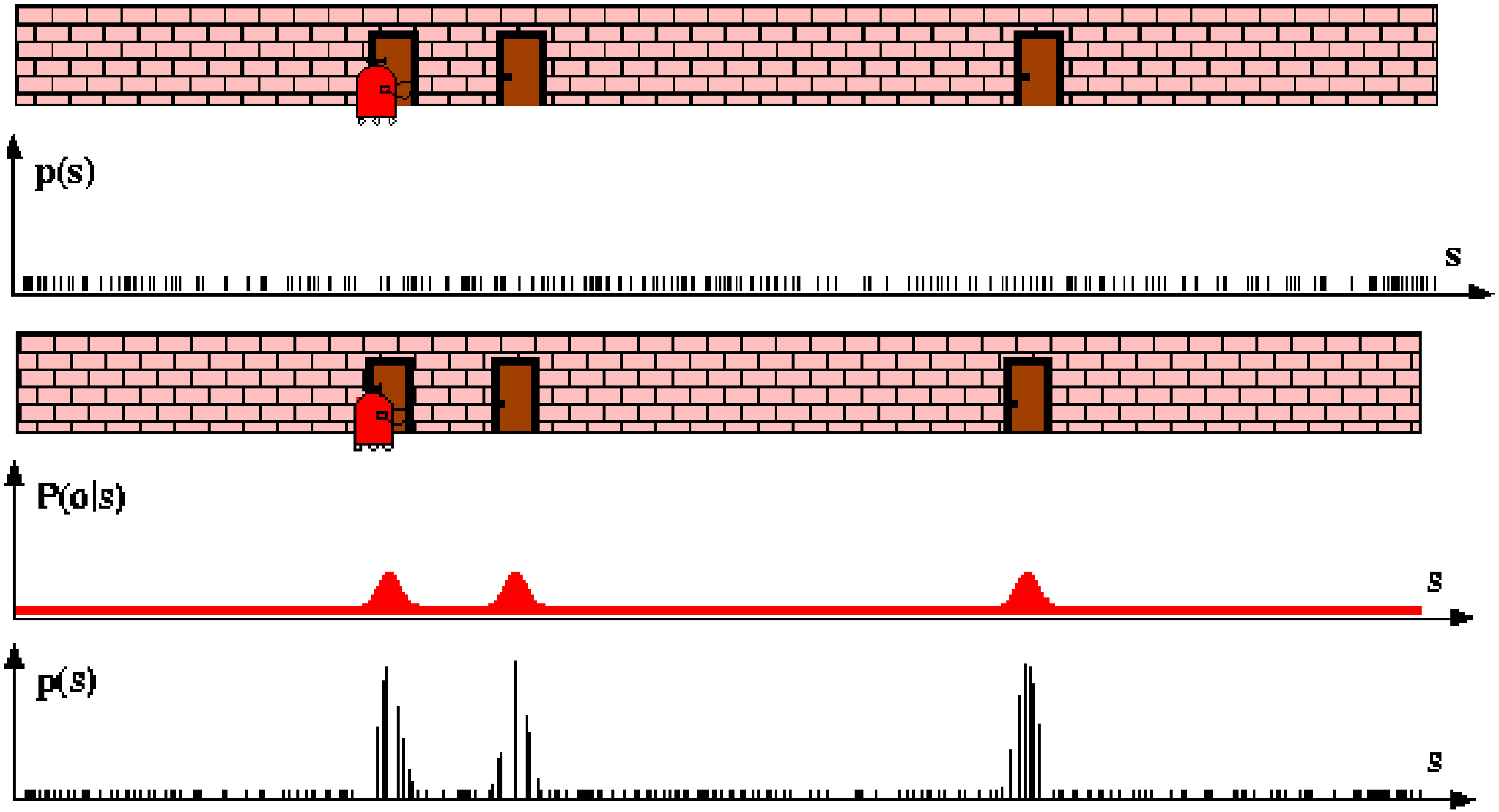
Particle Filters



Sensor Information: Importance Sampling

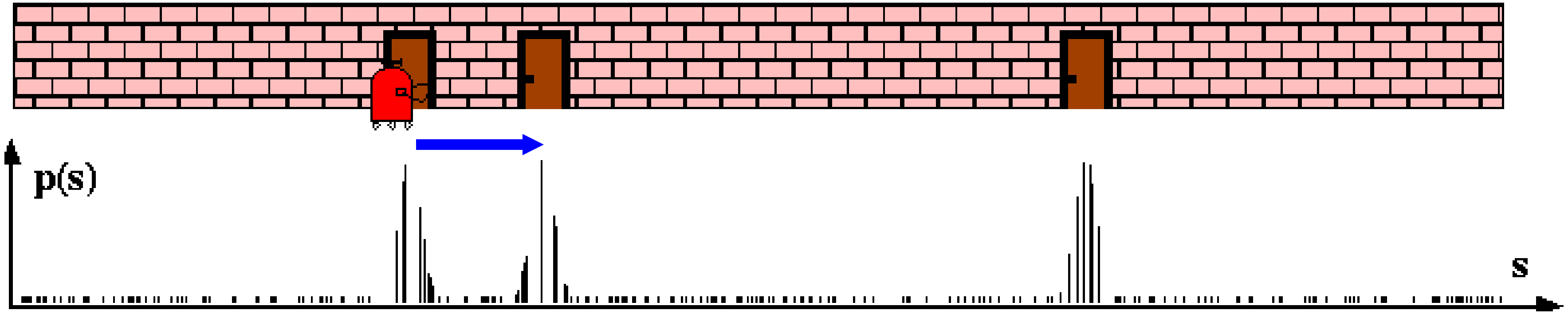
$$Bel(x) \leftarrow \alpha\ p(z \mid x)\ Bel^{-}(x)$$

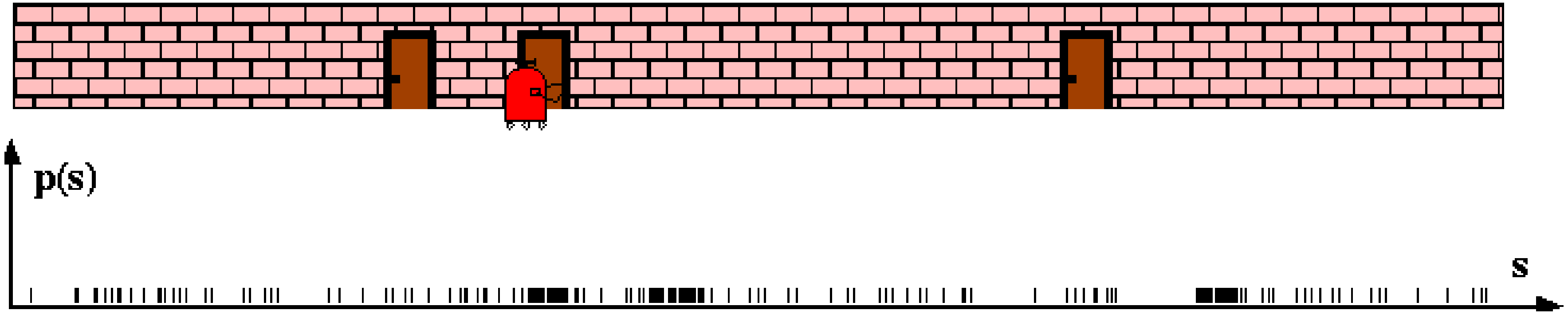
$$w \leftarrow \frac{\alpha\ p(z \mid x)\ Bel^{-}(x)}{Bel^{-}(x)} = \alpha\ p(z \mid x)$$



Robot Motion

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

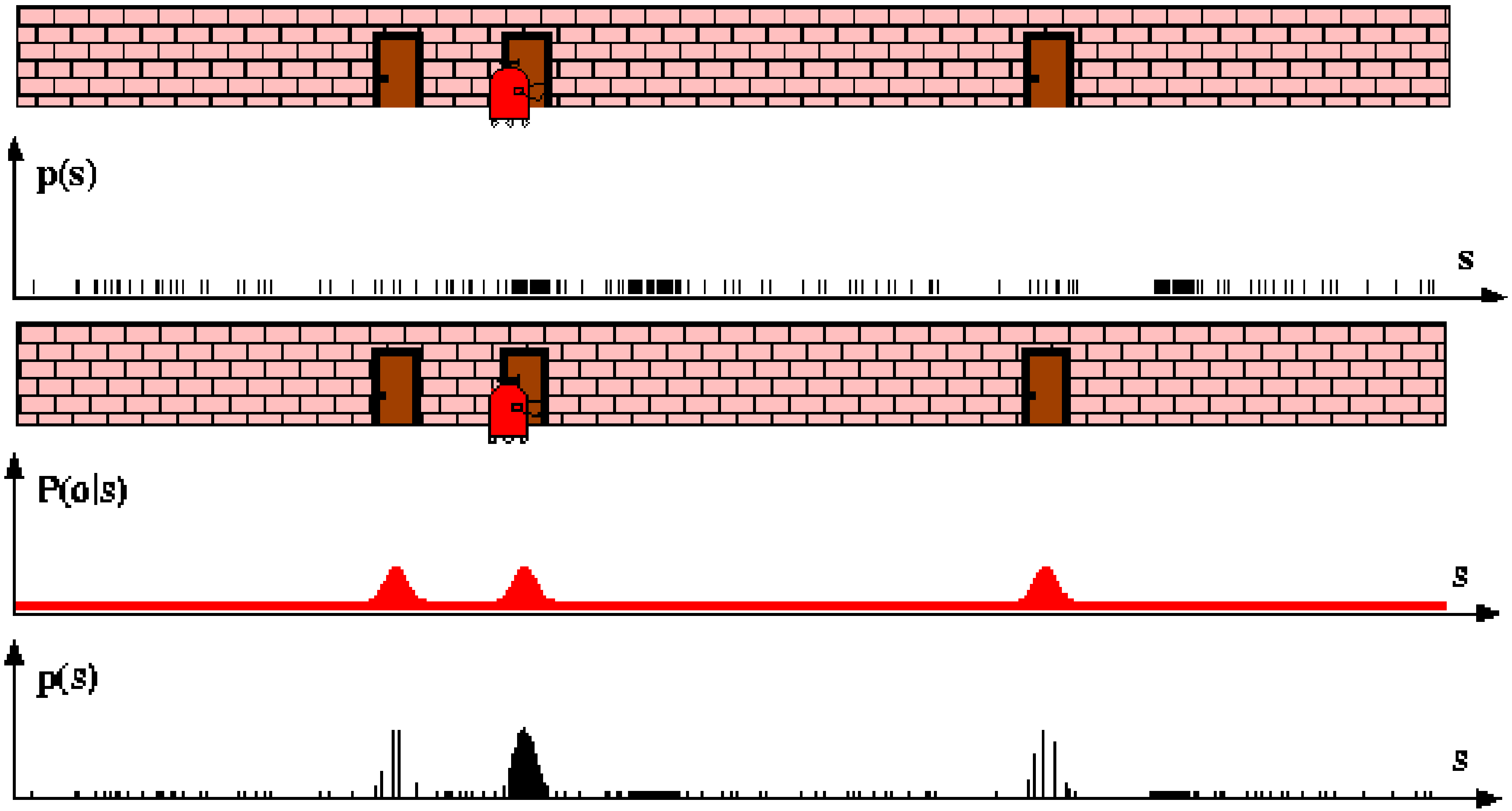




Sensor Information: Importance Sampling

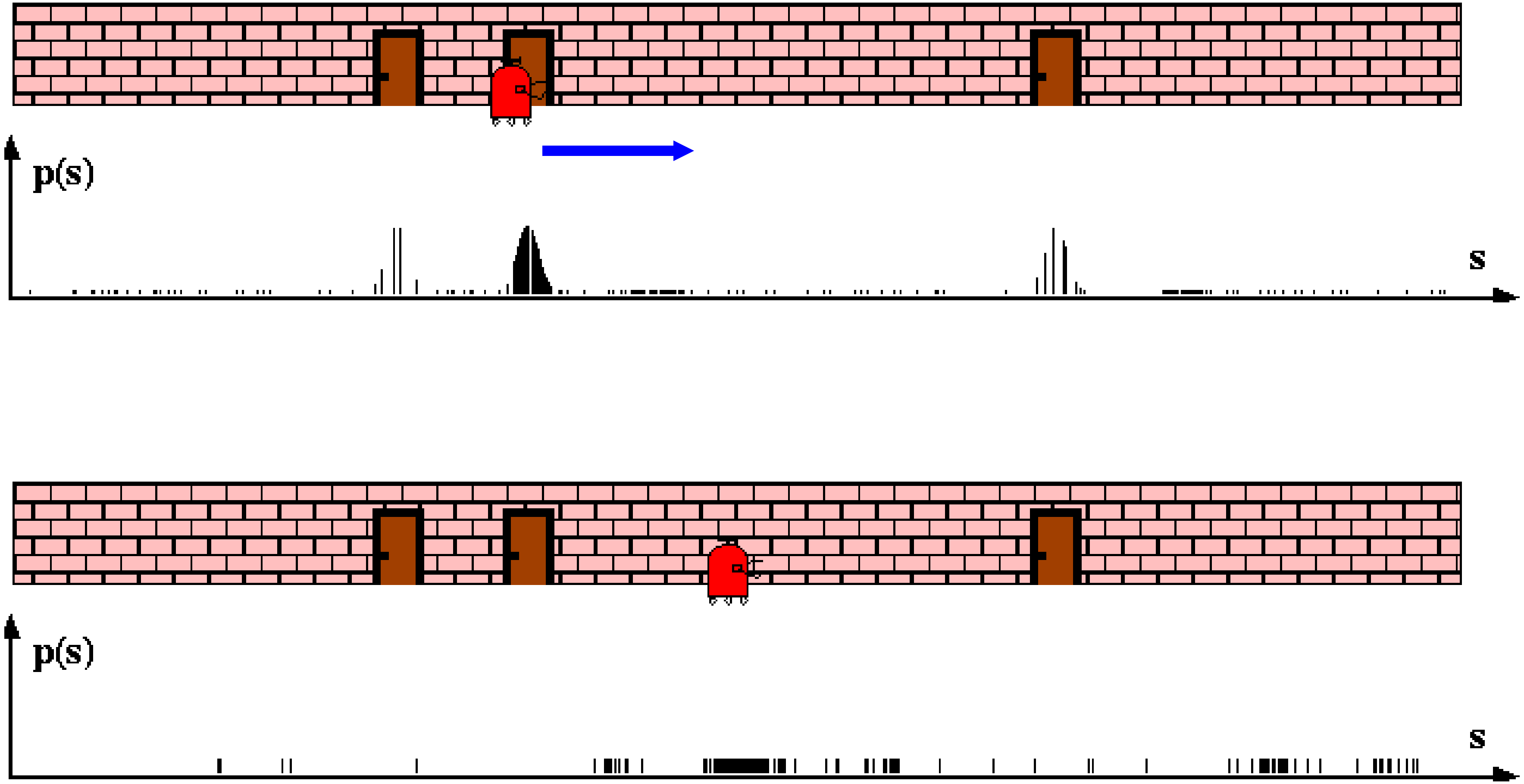
$$Bel(x) \leftarrow \alpha\ p(z \mid x)\ Bel^{-}(x)$$

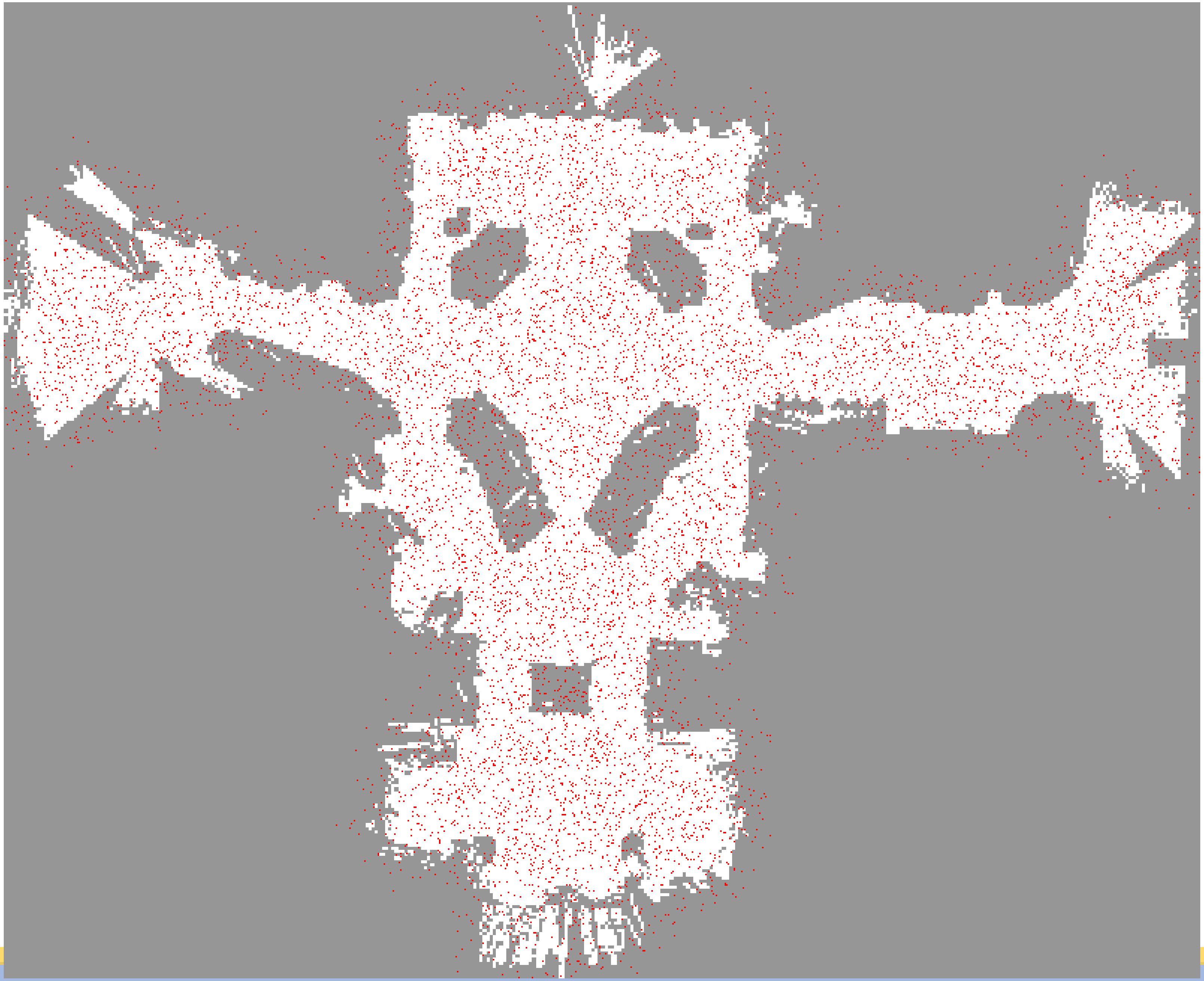
$$w \leftarrow \frac{\alpha\ p(z \mid x)\ Bel^{-}(x)}{Bel^{-}(x)} = \alpha\ p(z \mid x)$$

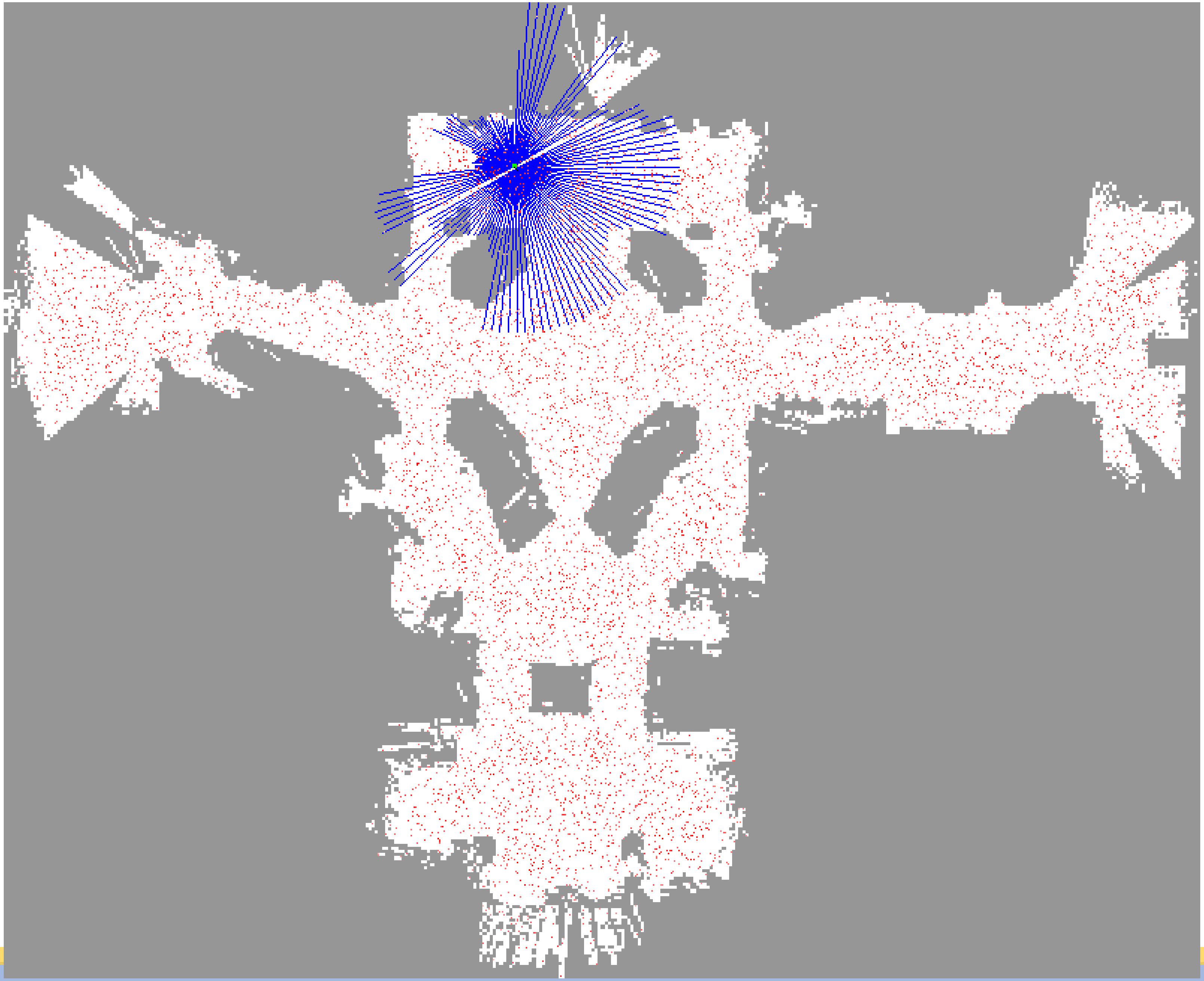


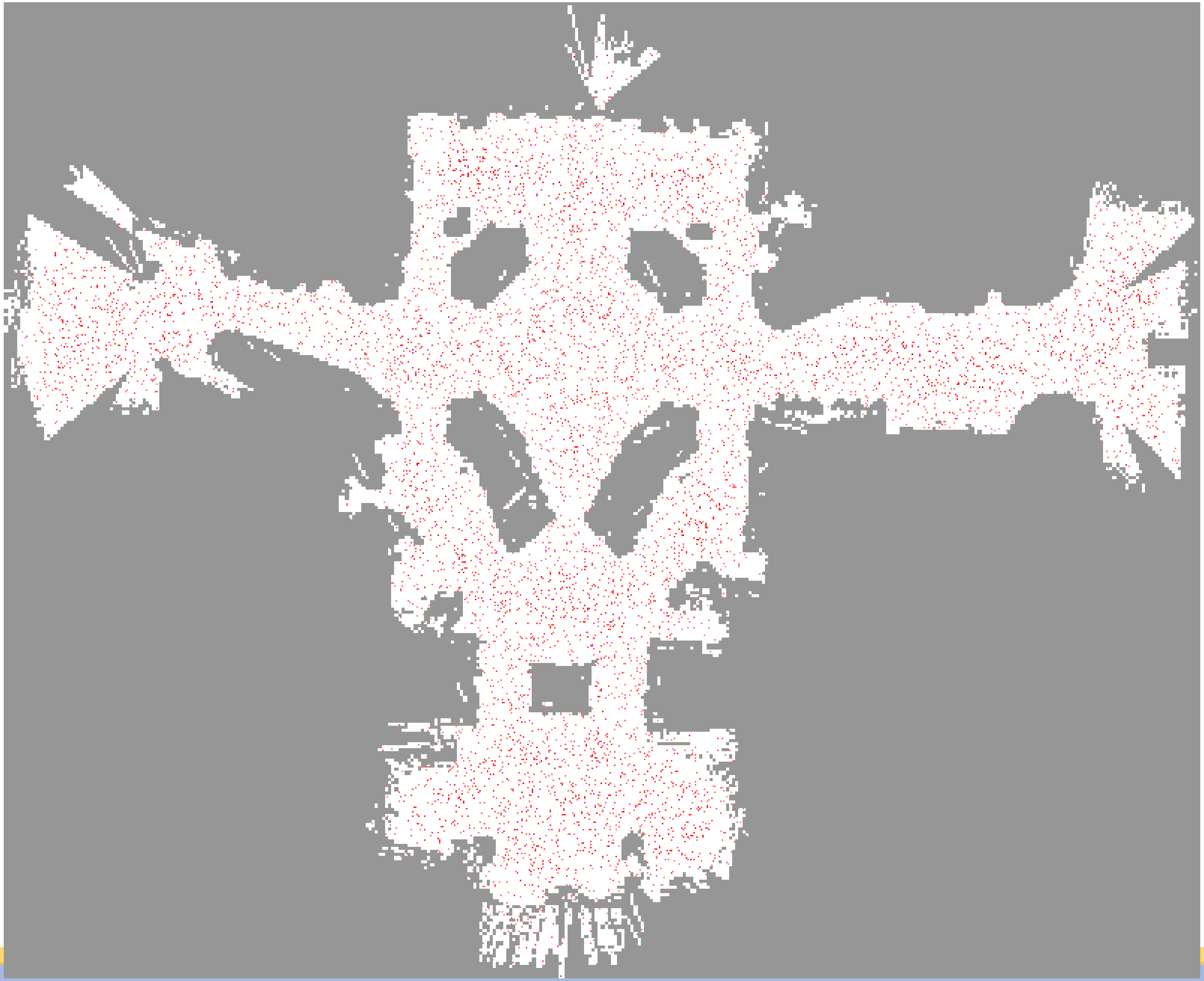
Robot Motion

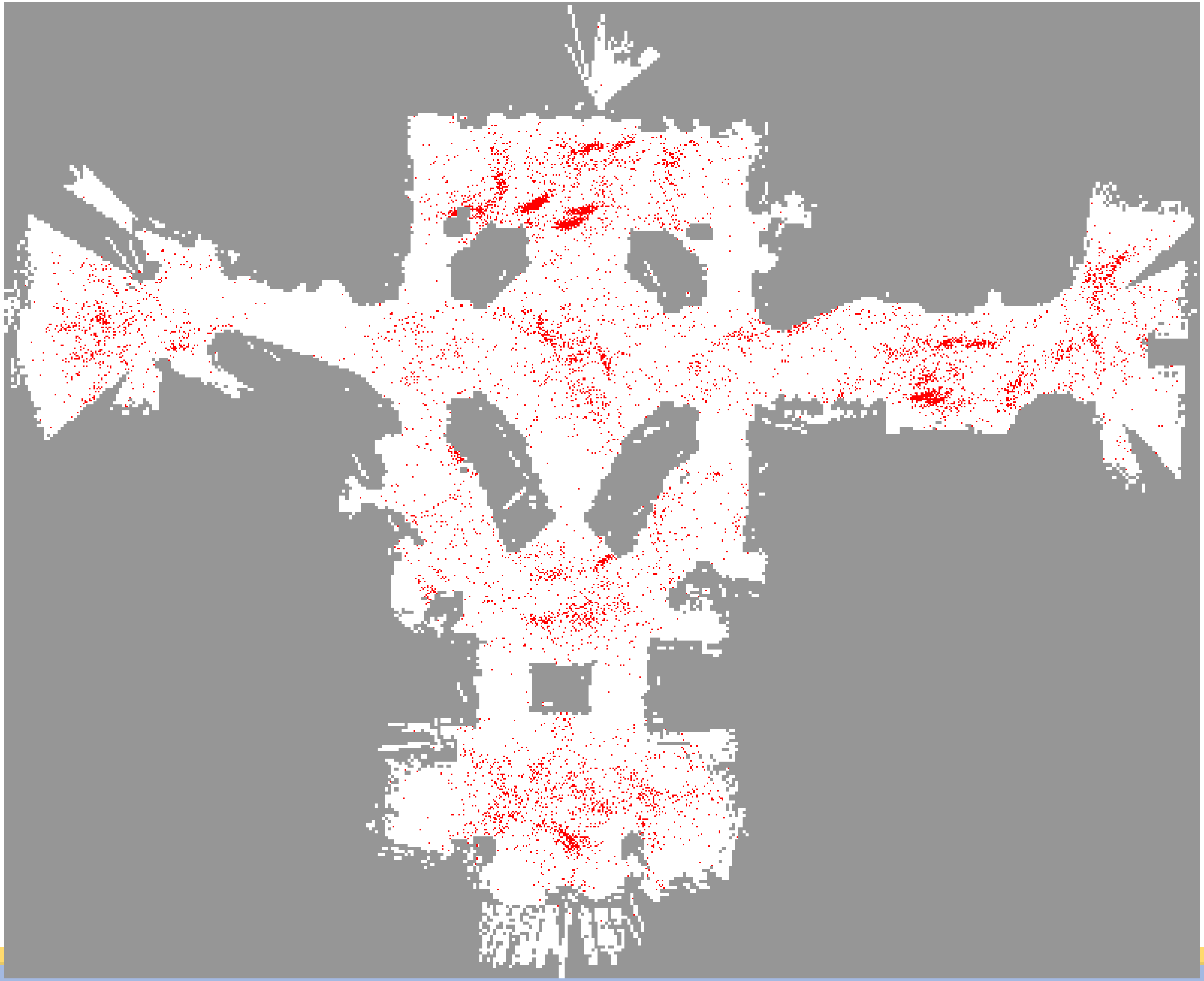
$$Bel^-(x) \leftarrow \int p(x \mid u, x')\, Bel(x')\, \mathrm{d}x'$$

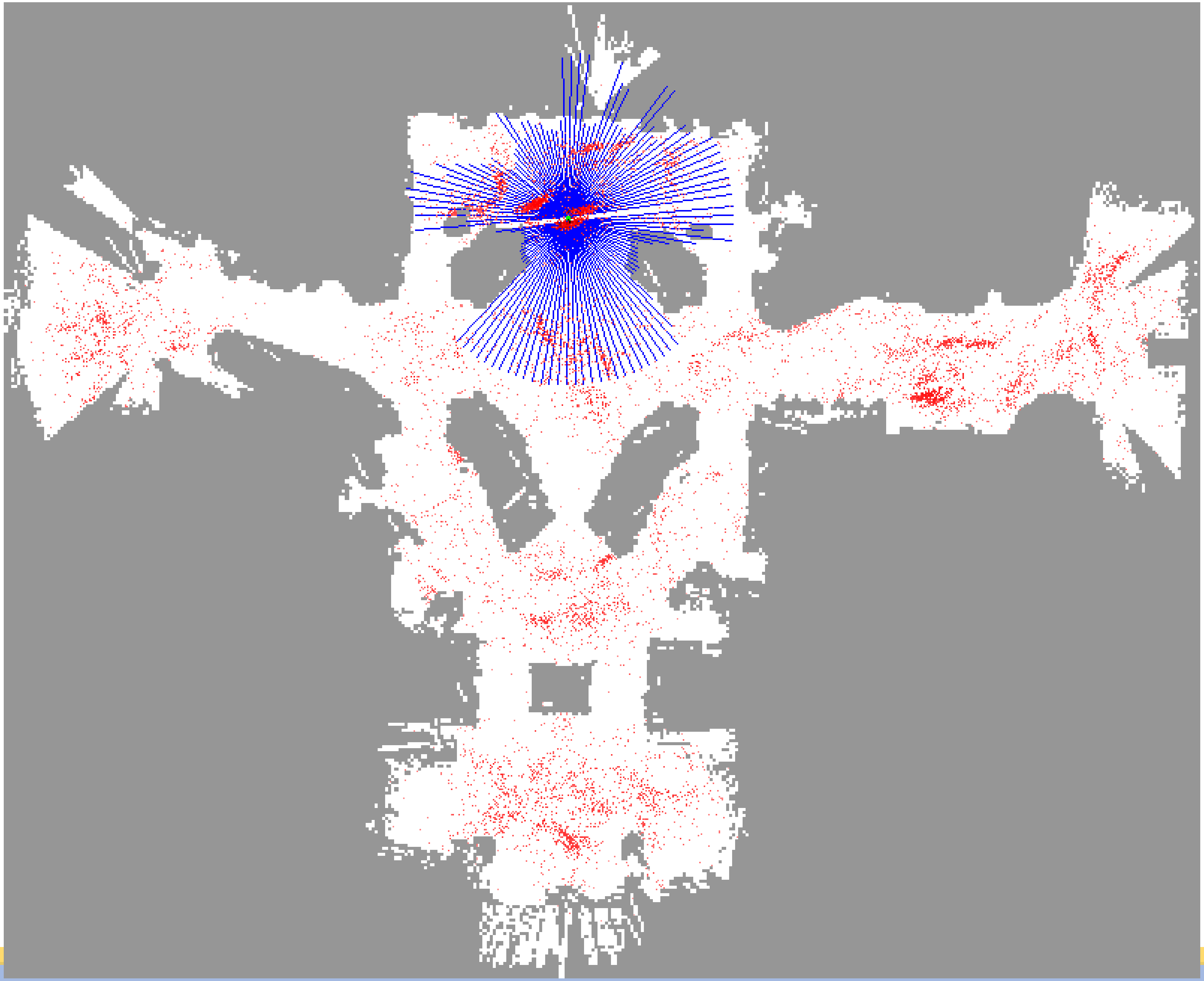


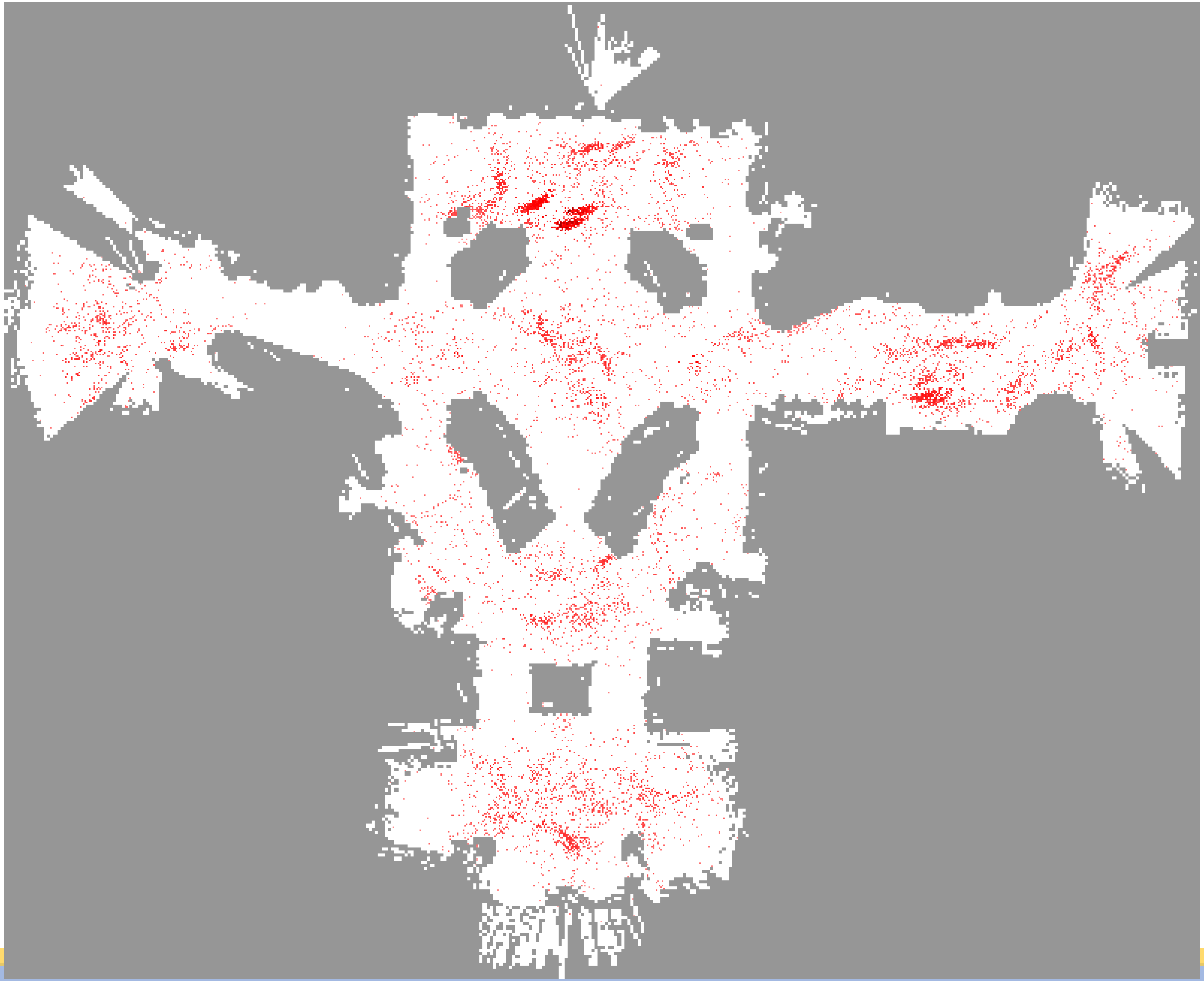


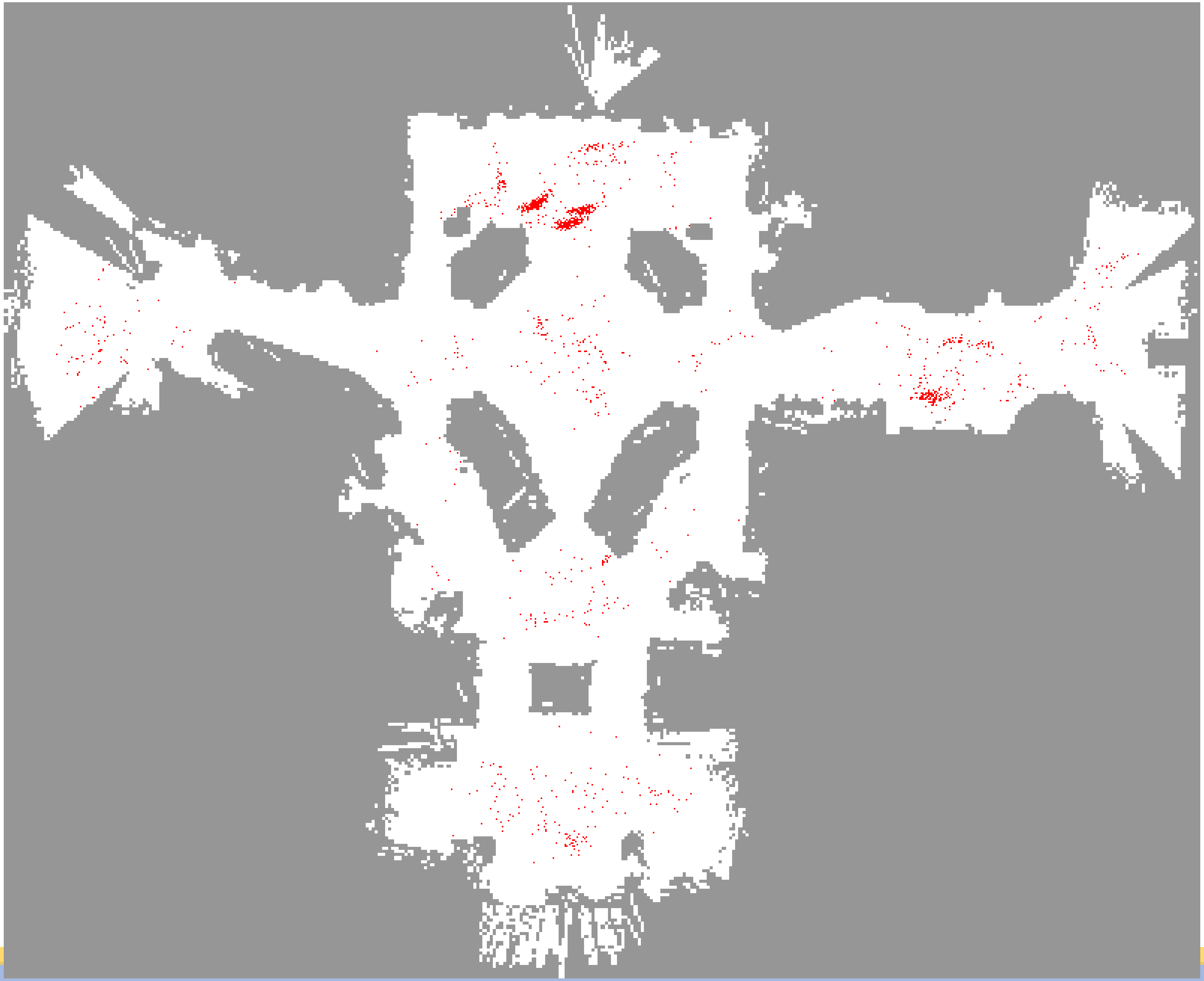


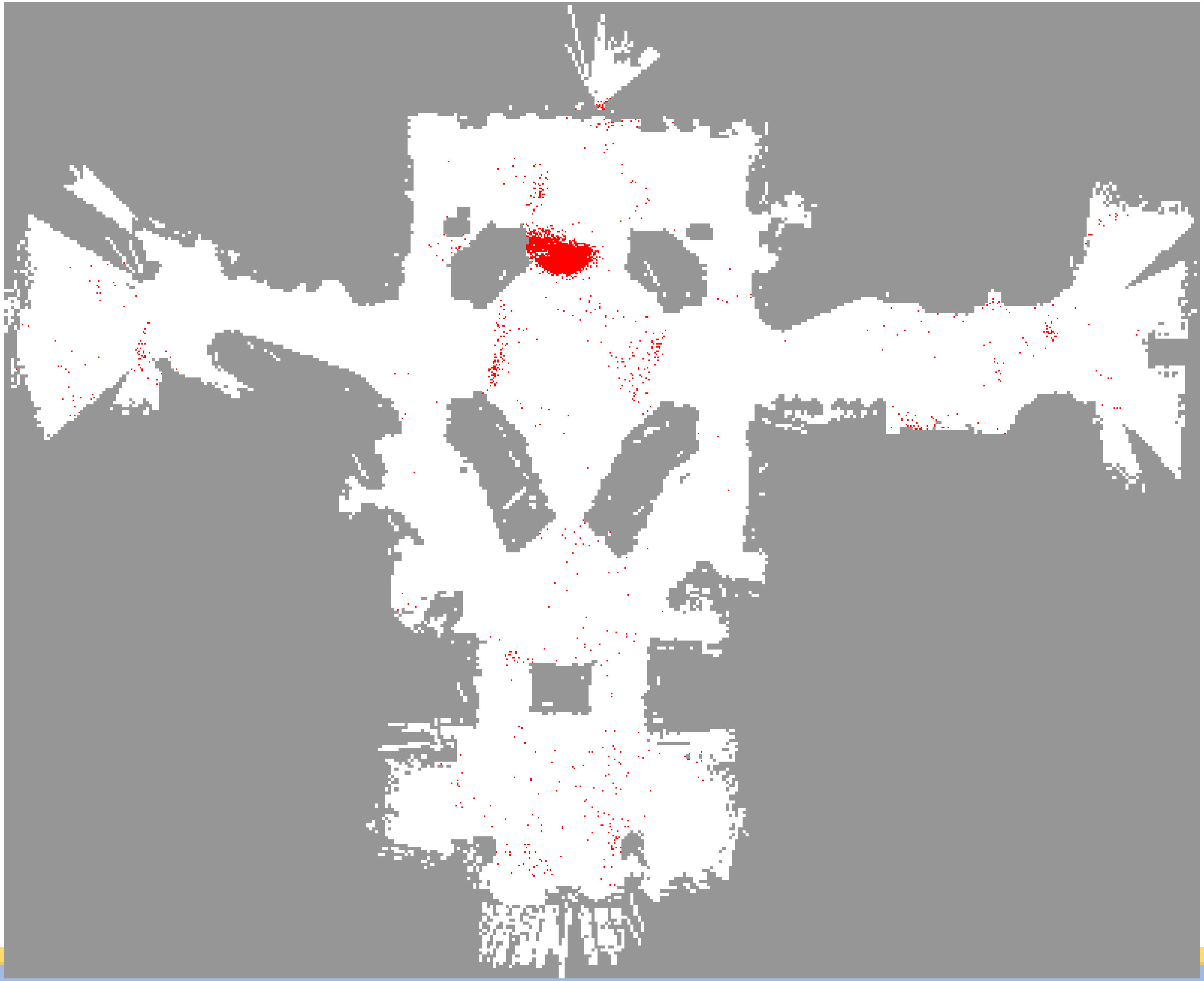


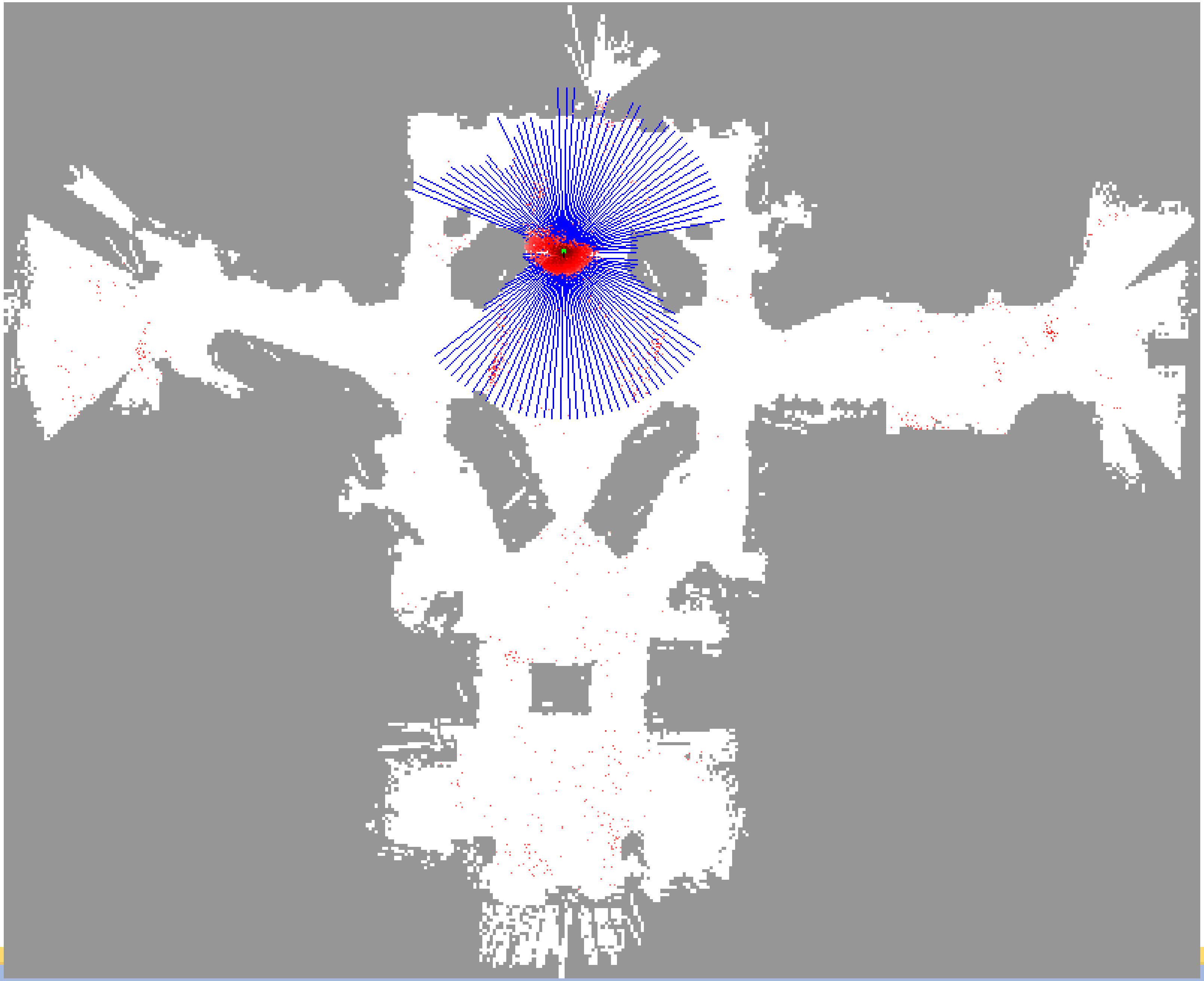


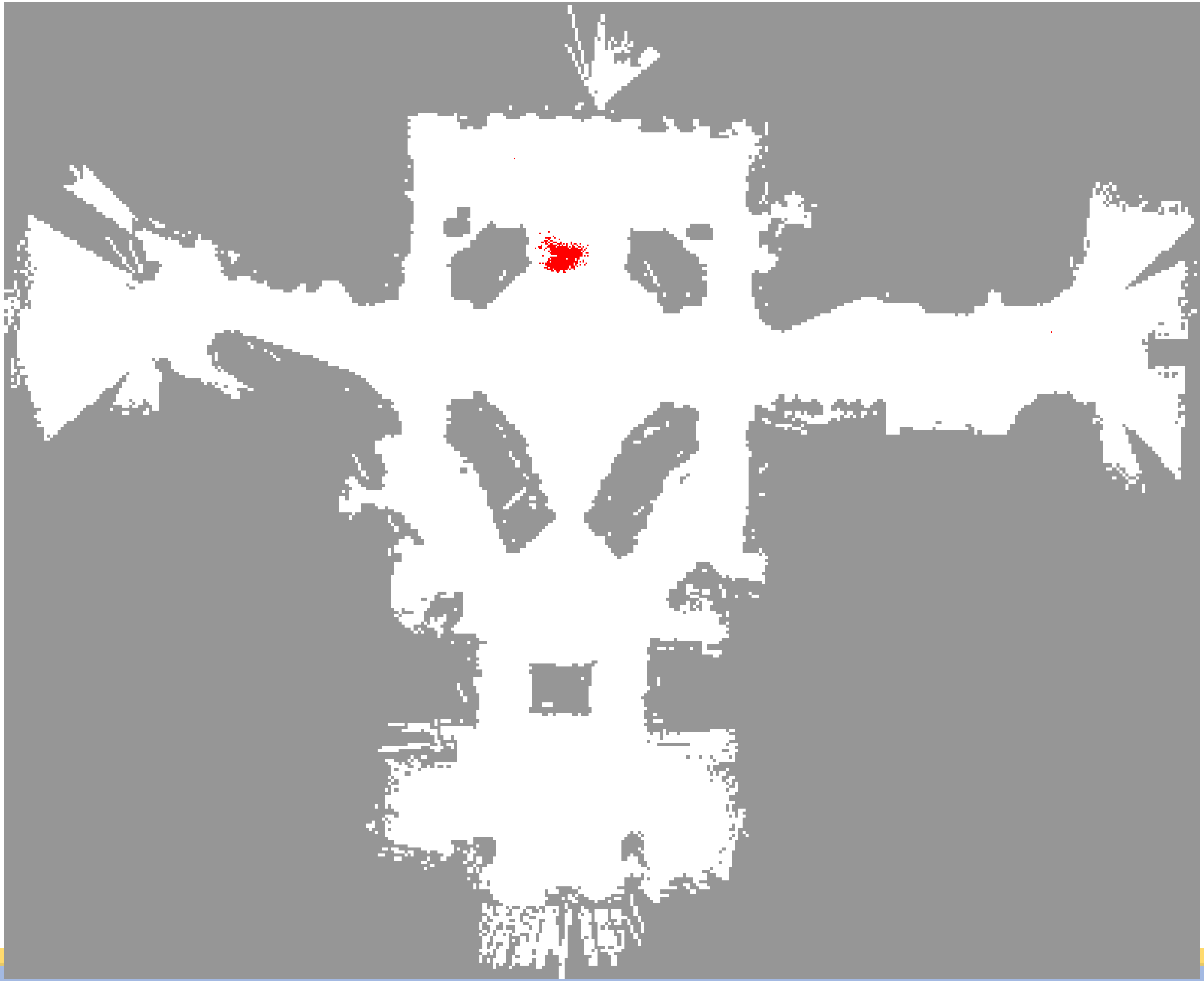


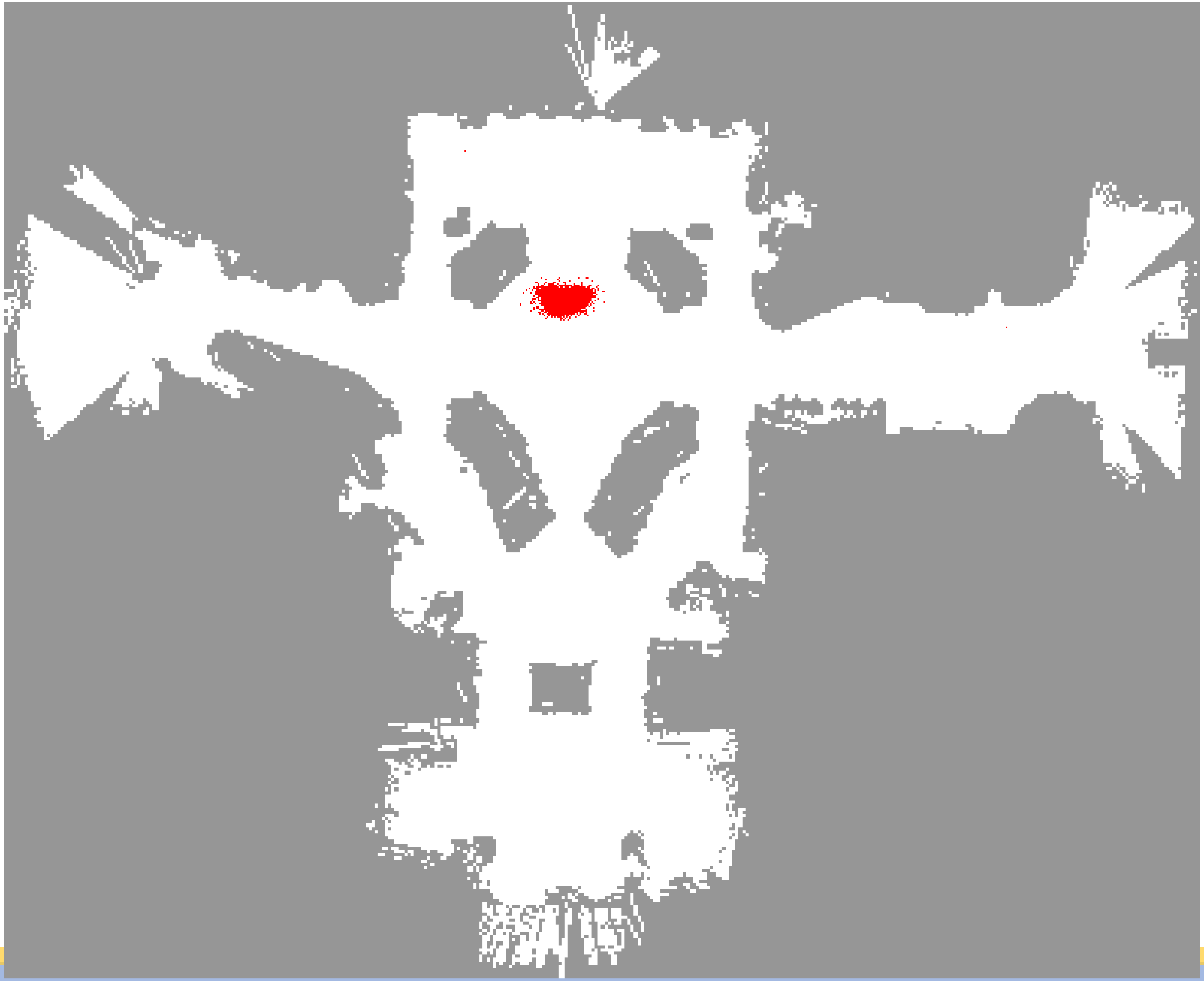


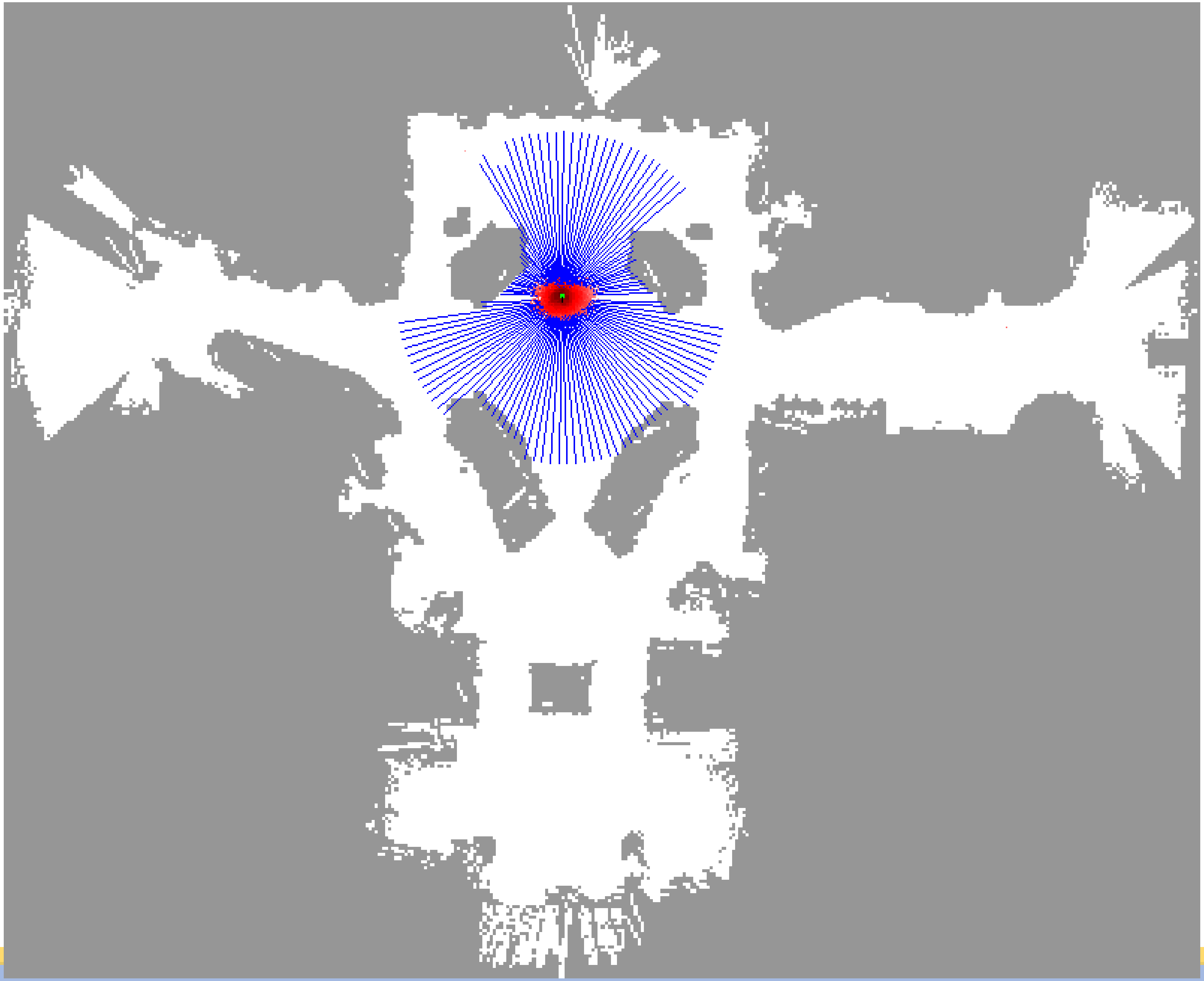


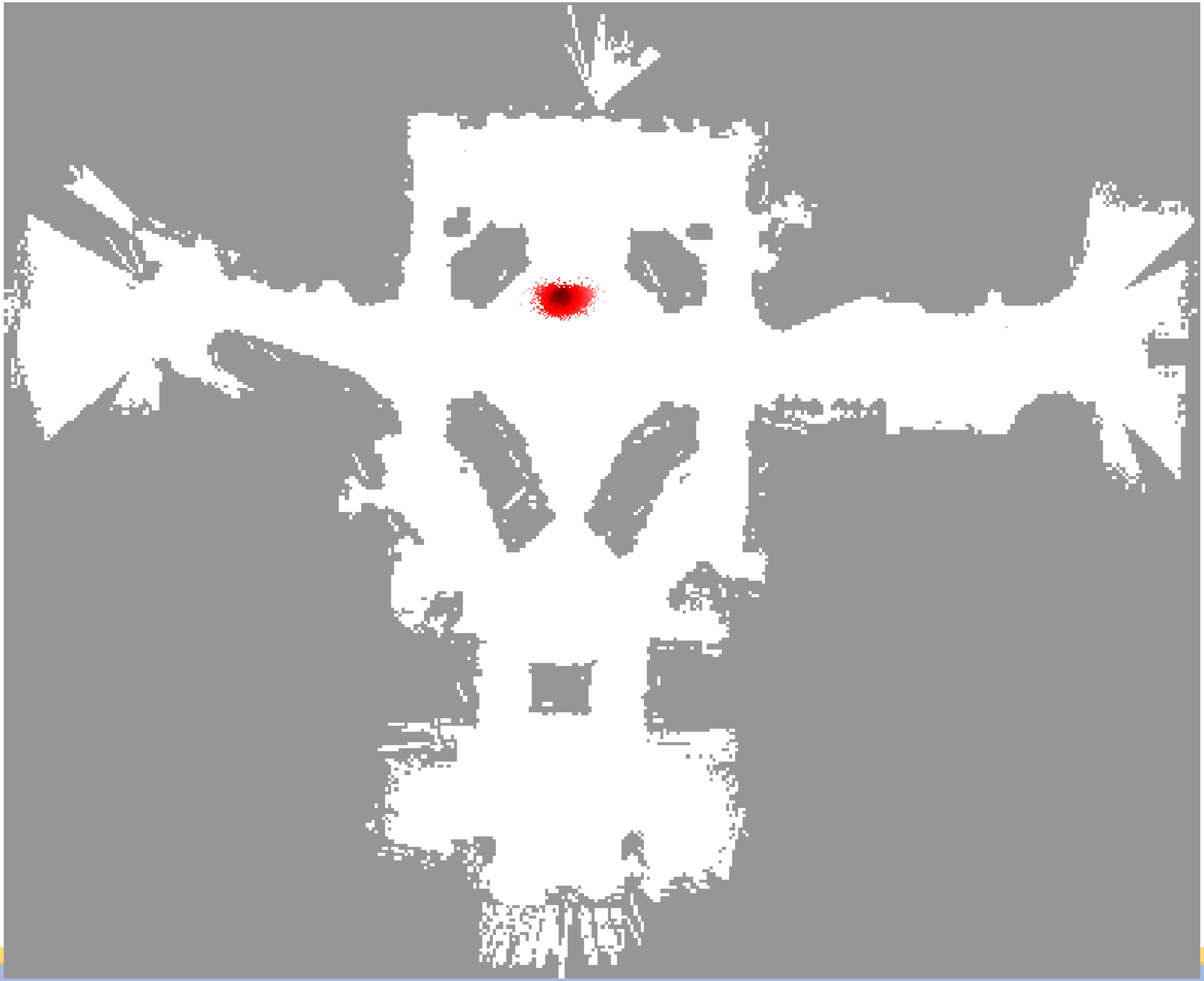


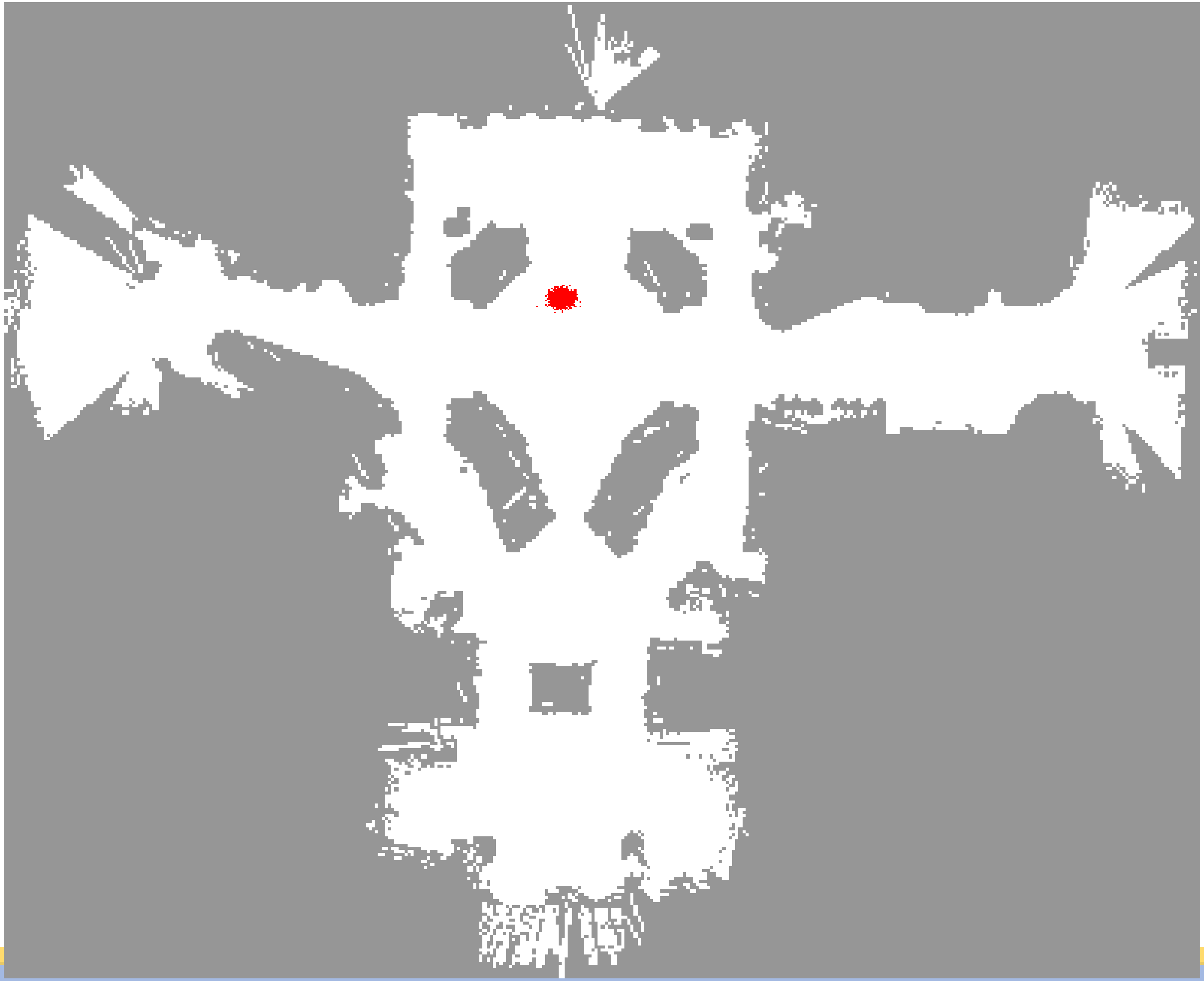


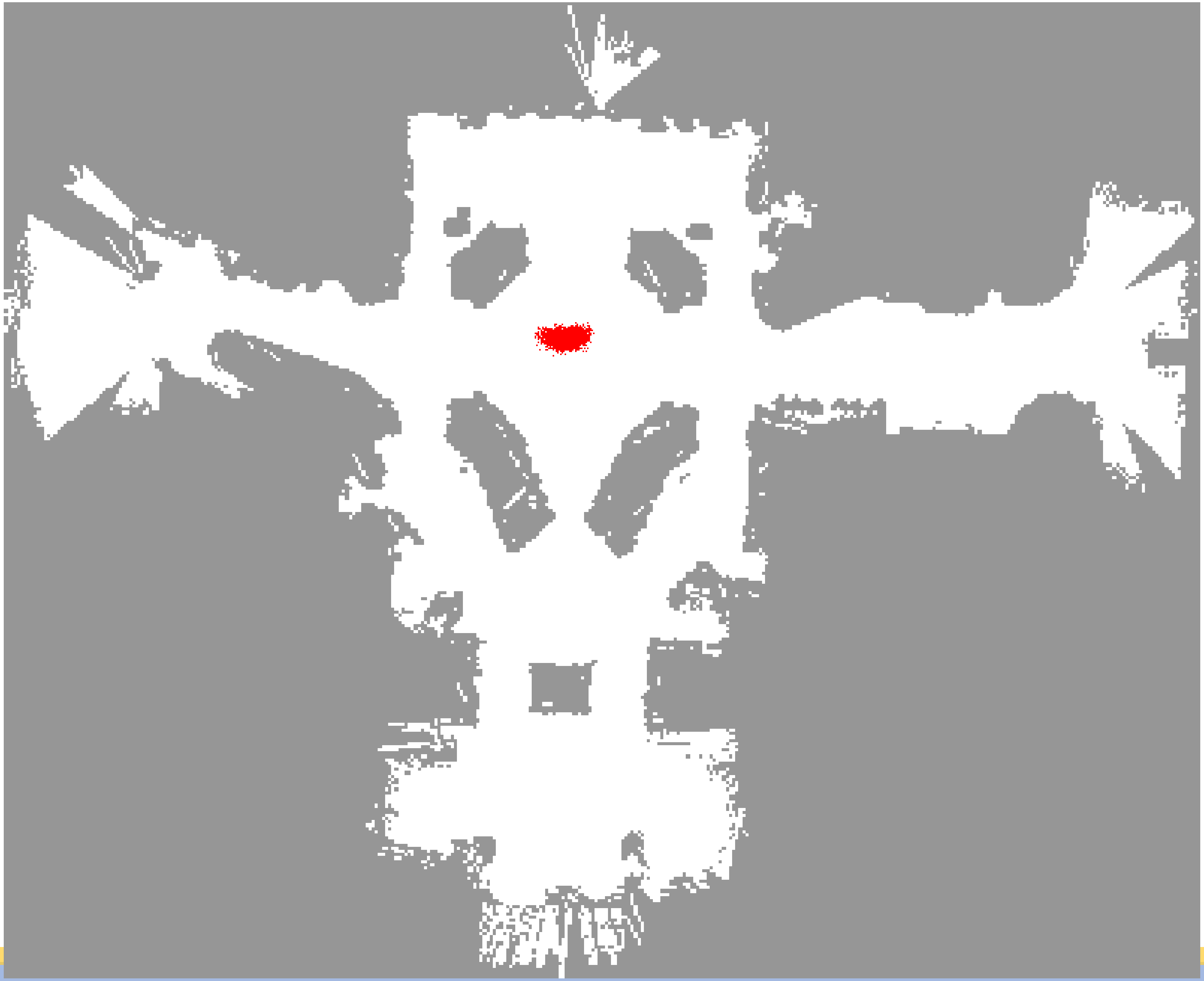


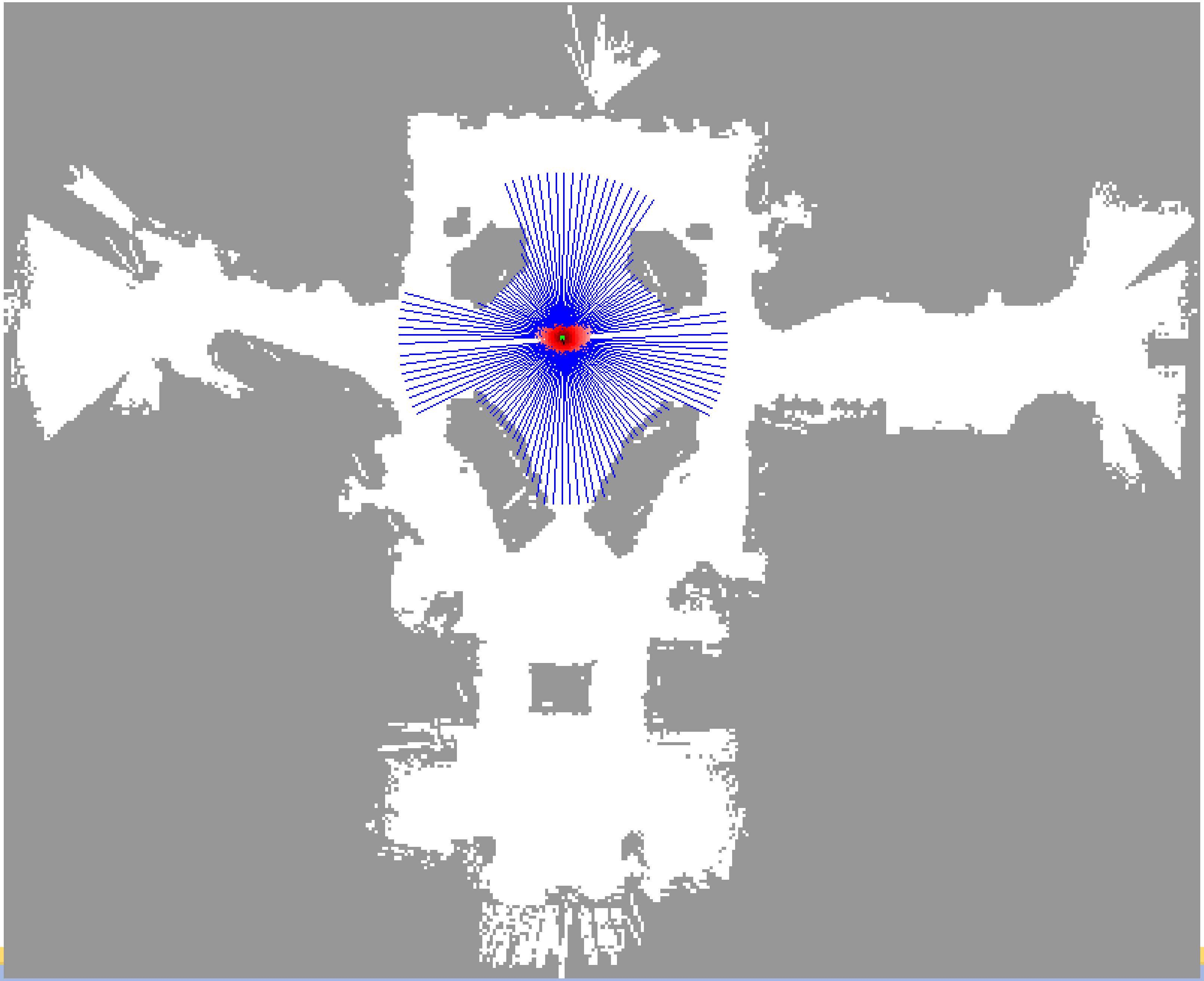


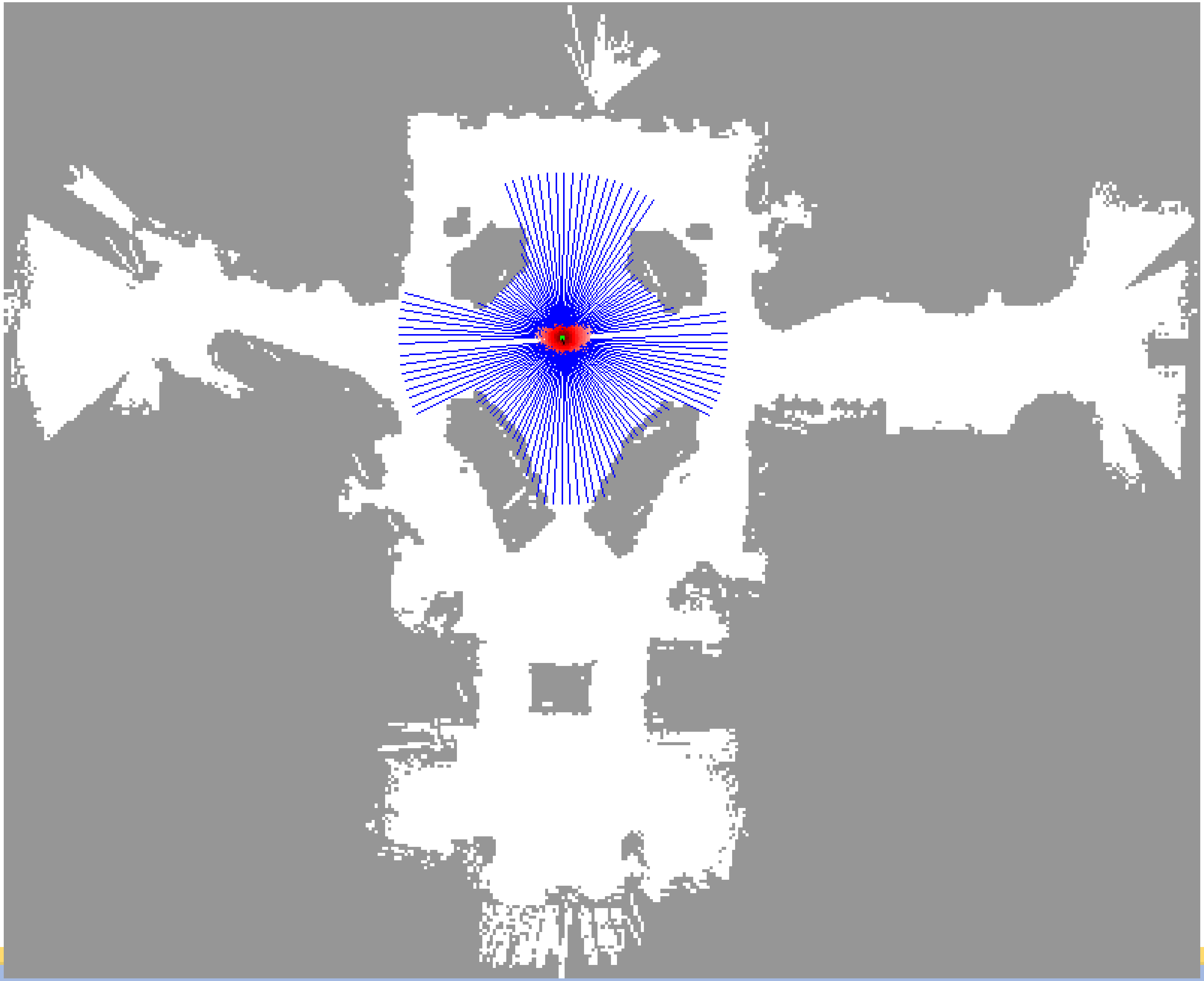




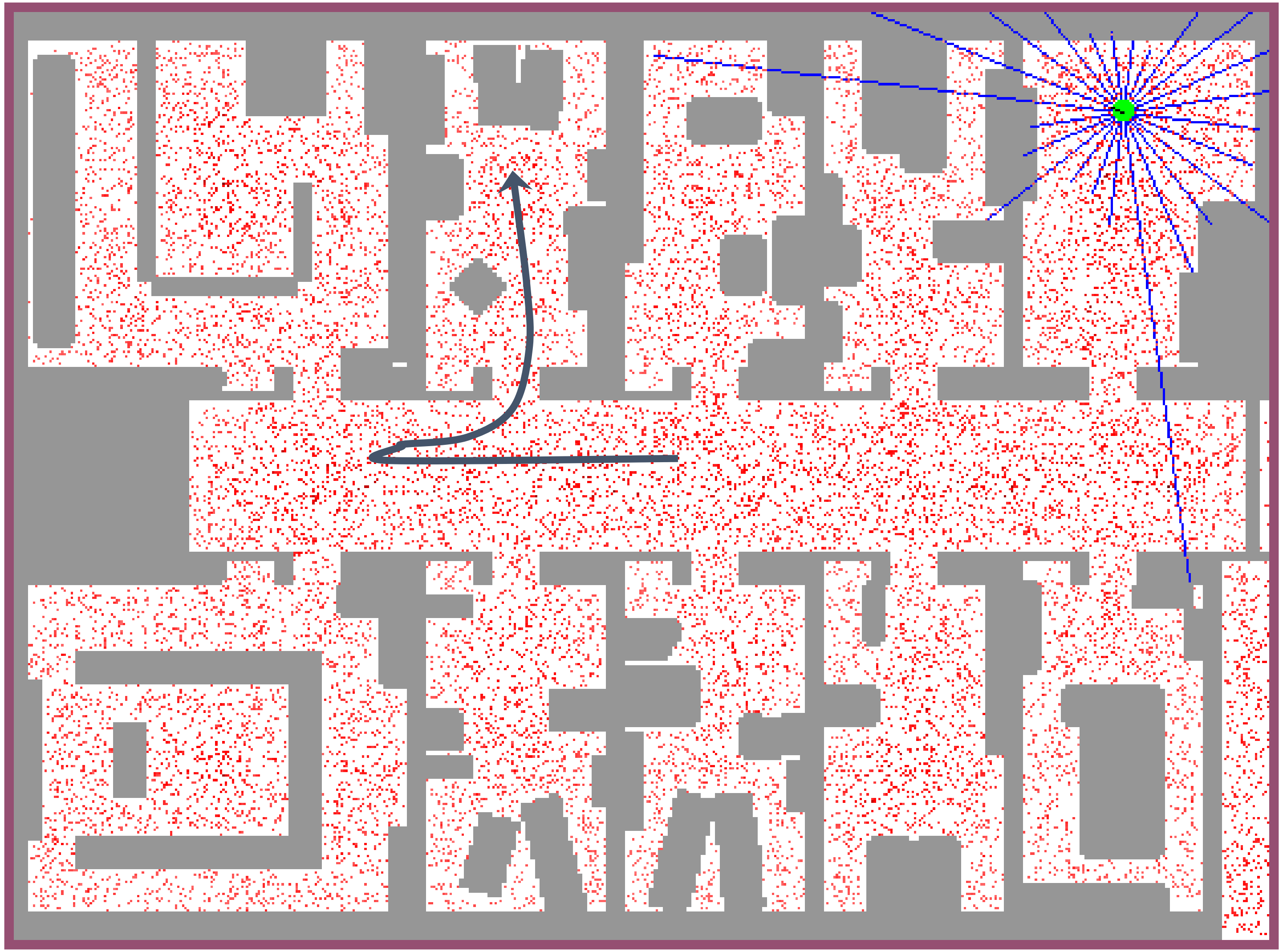




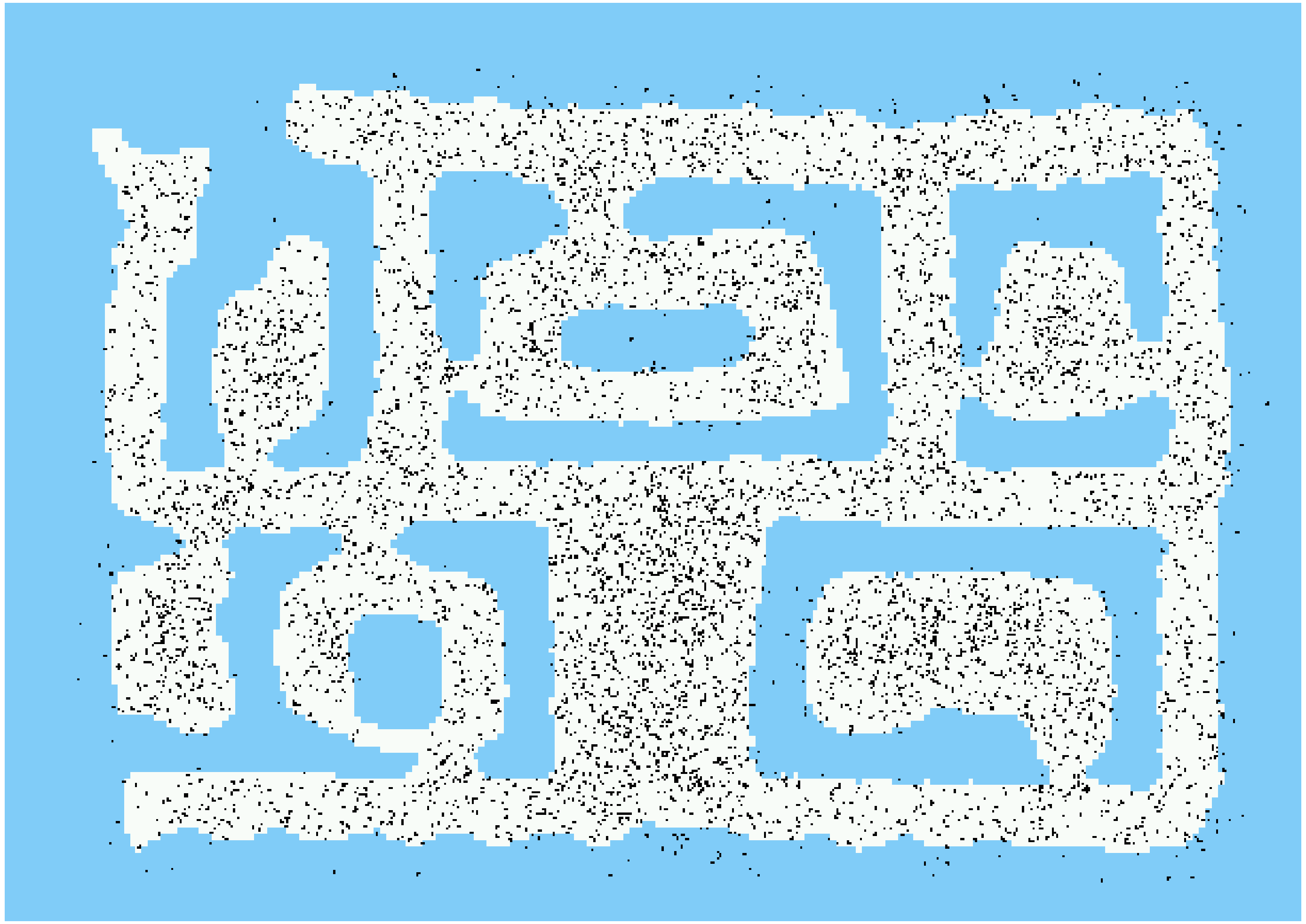




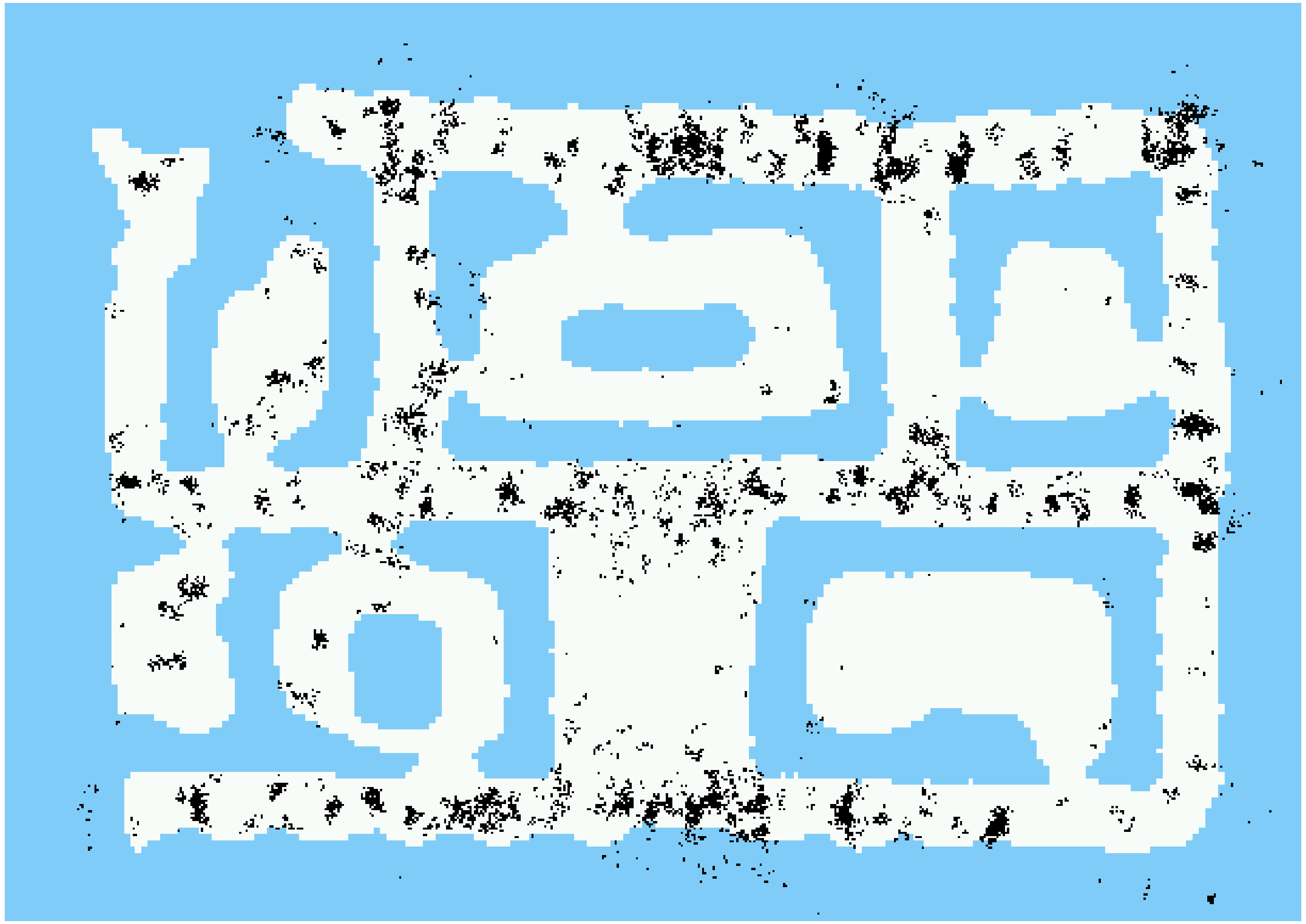
Sample-based Localization (sonar)



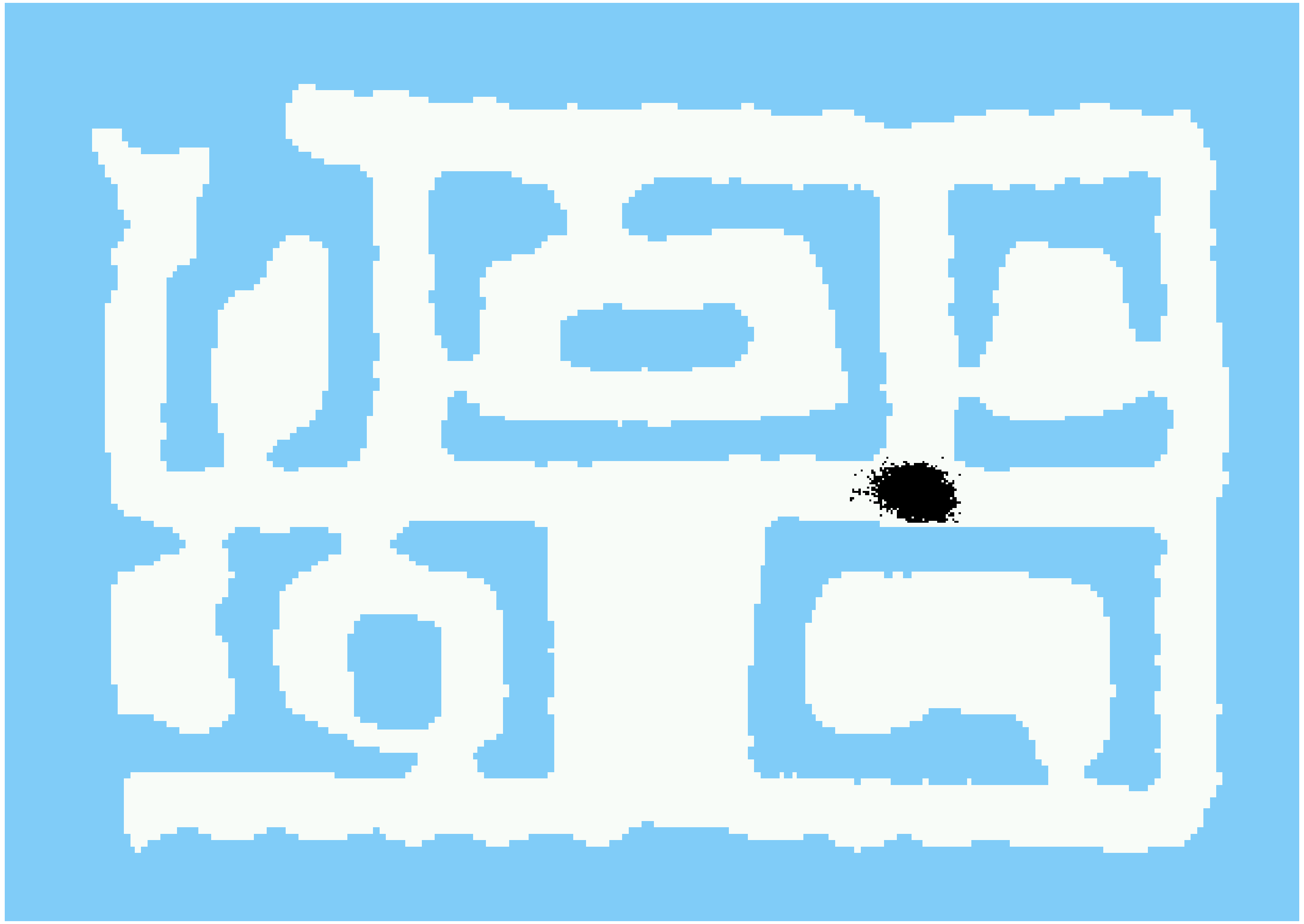
Initial Distribution



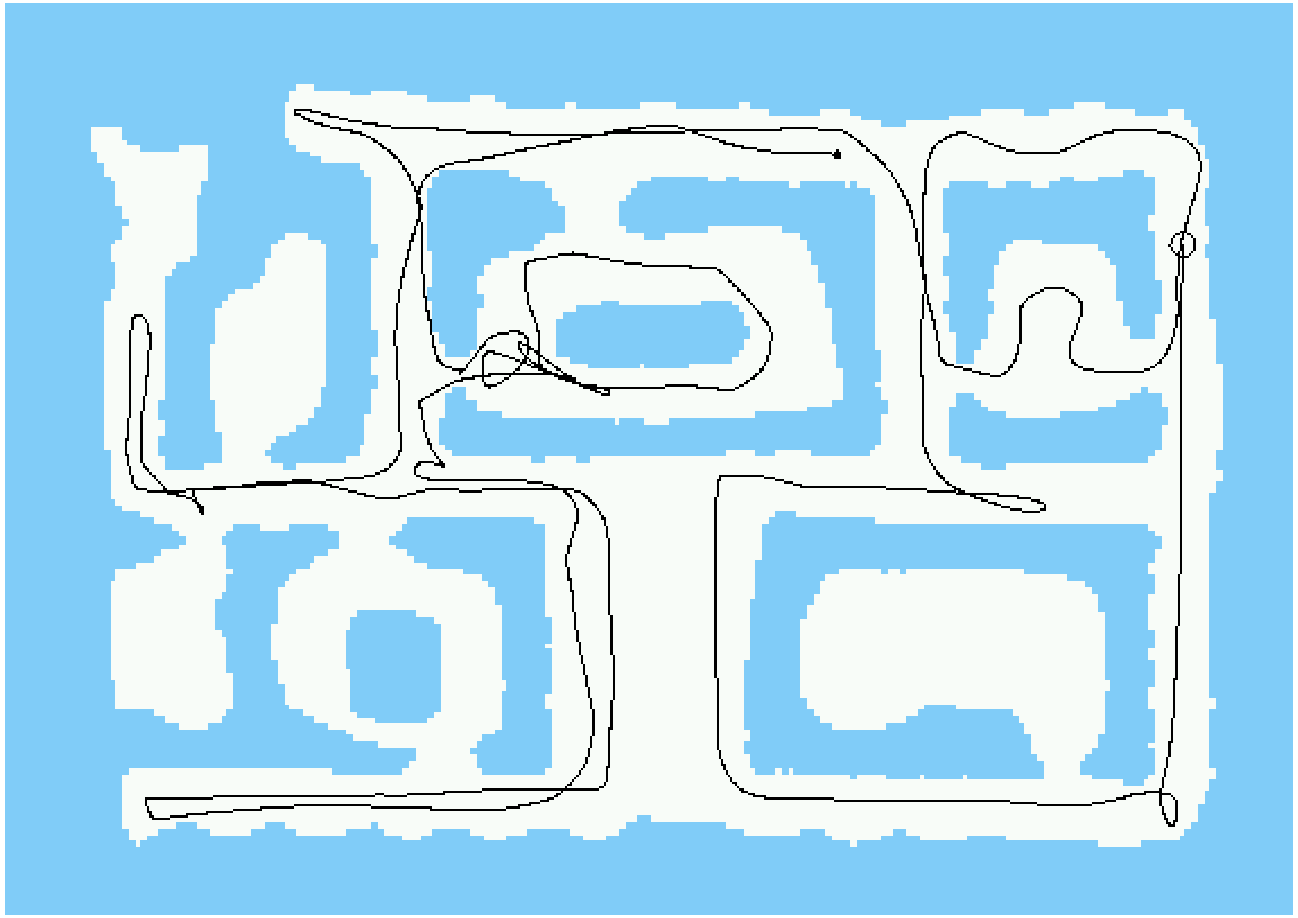
After Incorporating Ten Ultrasound Scans



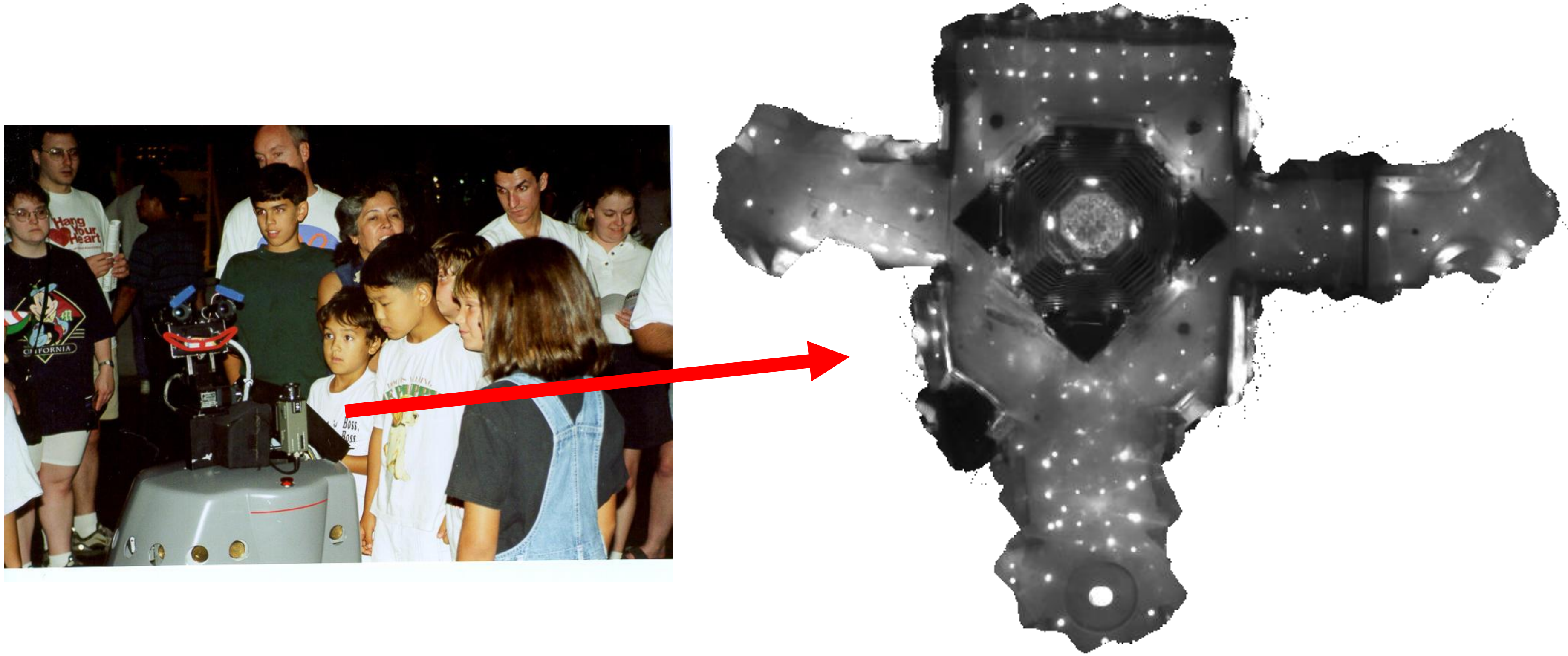
After Incorporating 65 Ultrasound Scans



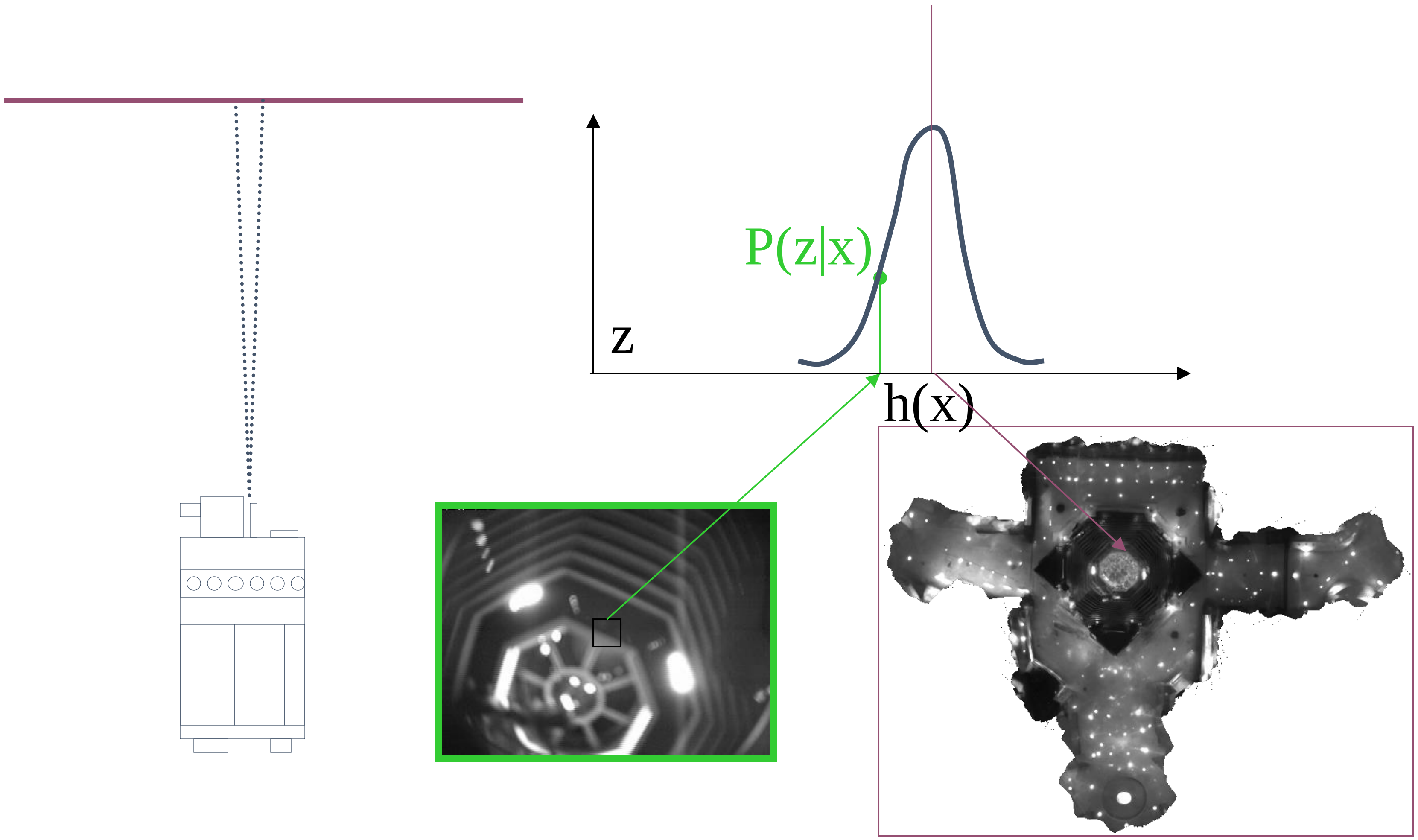
Estimated Path



Using Ceiling Maps for Localization

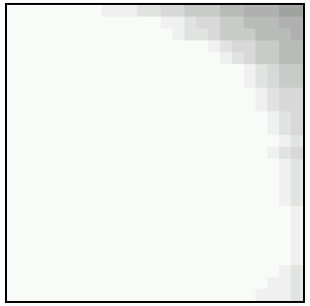


Vision-based Localization

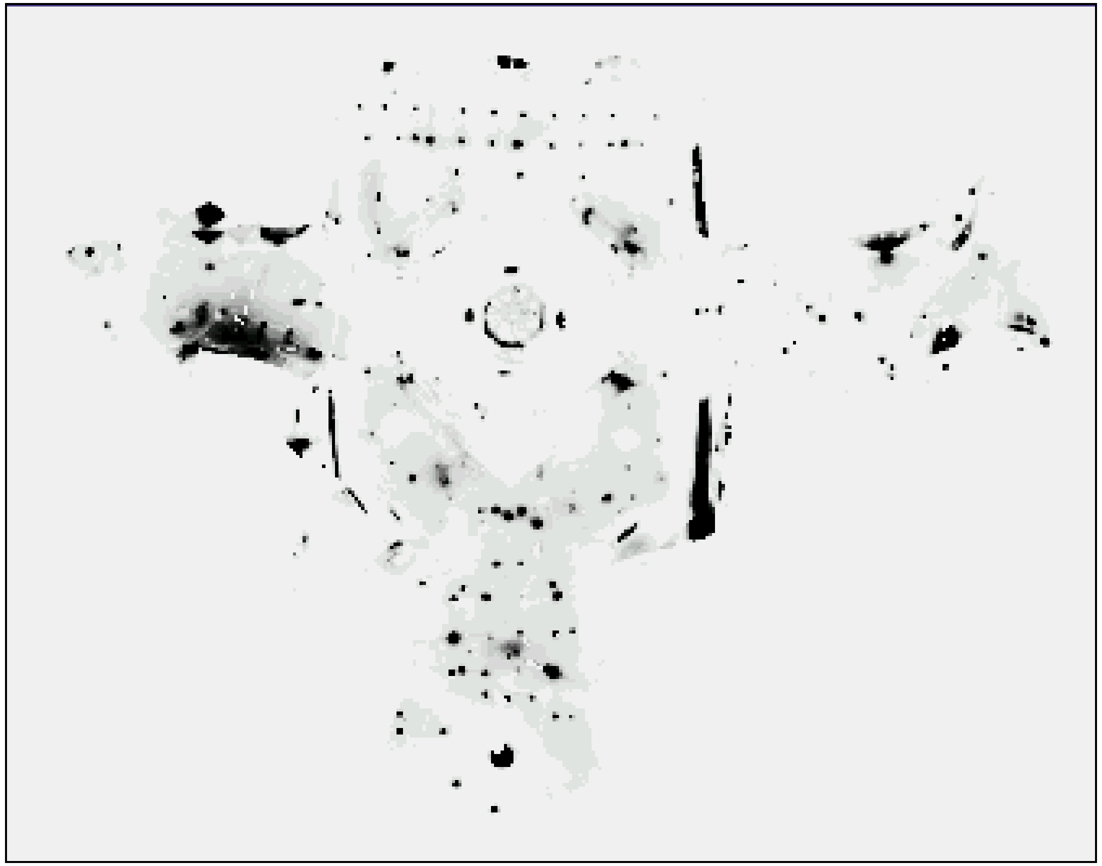


Under a Light:

Measurement z:

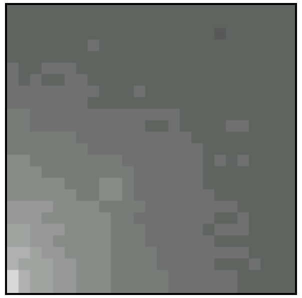


***P(z|x)*:**

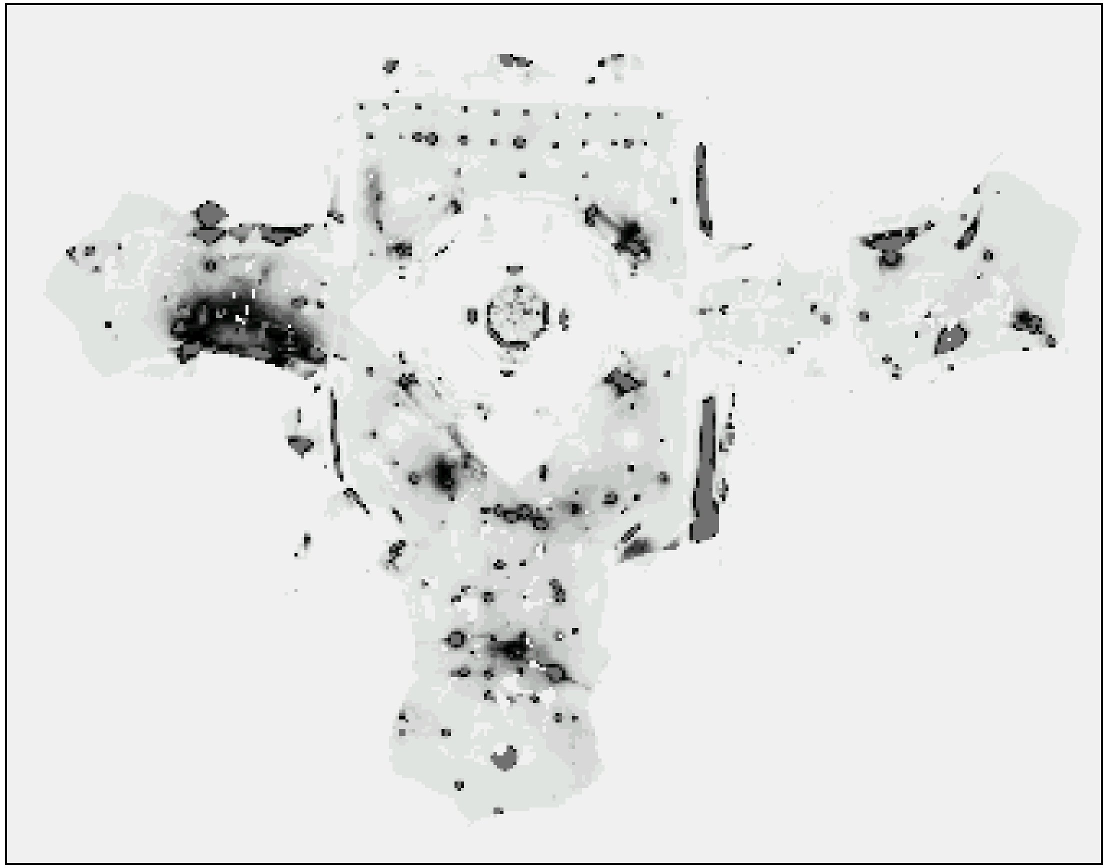


Next to a Light

Measurement z:



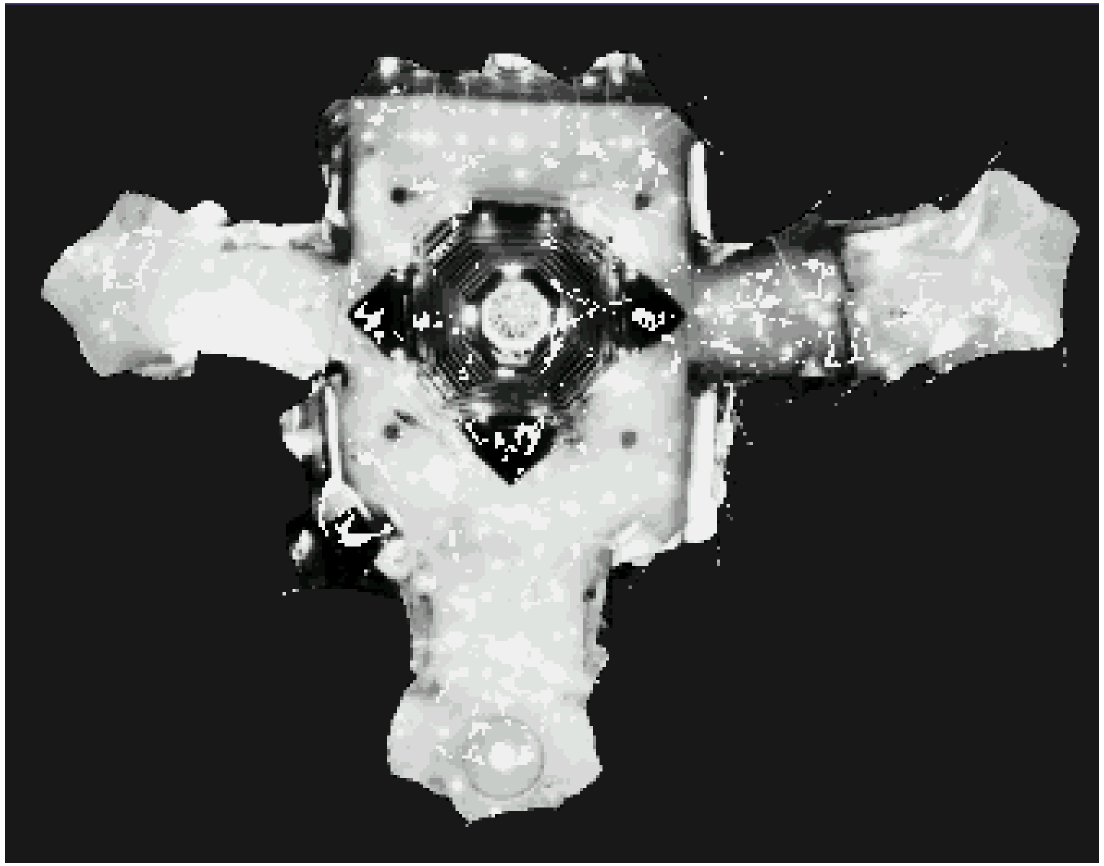
***P(z|x)*:**



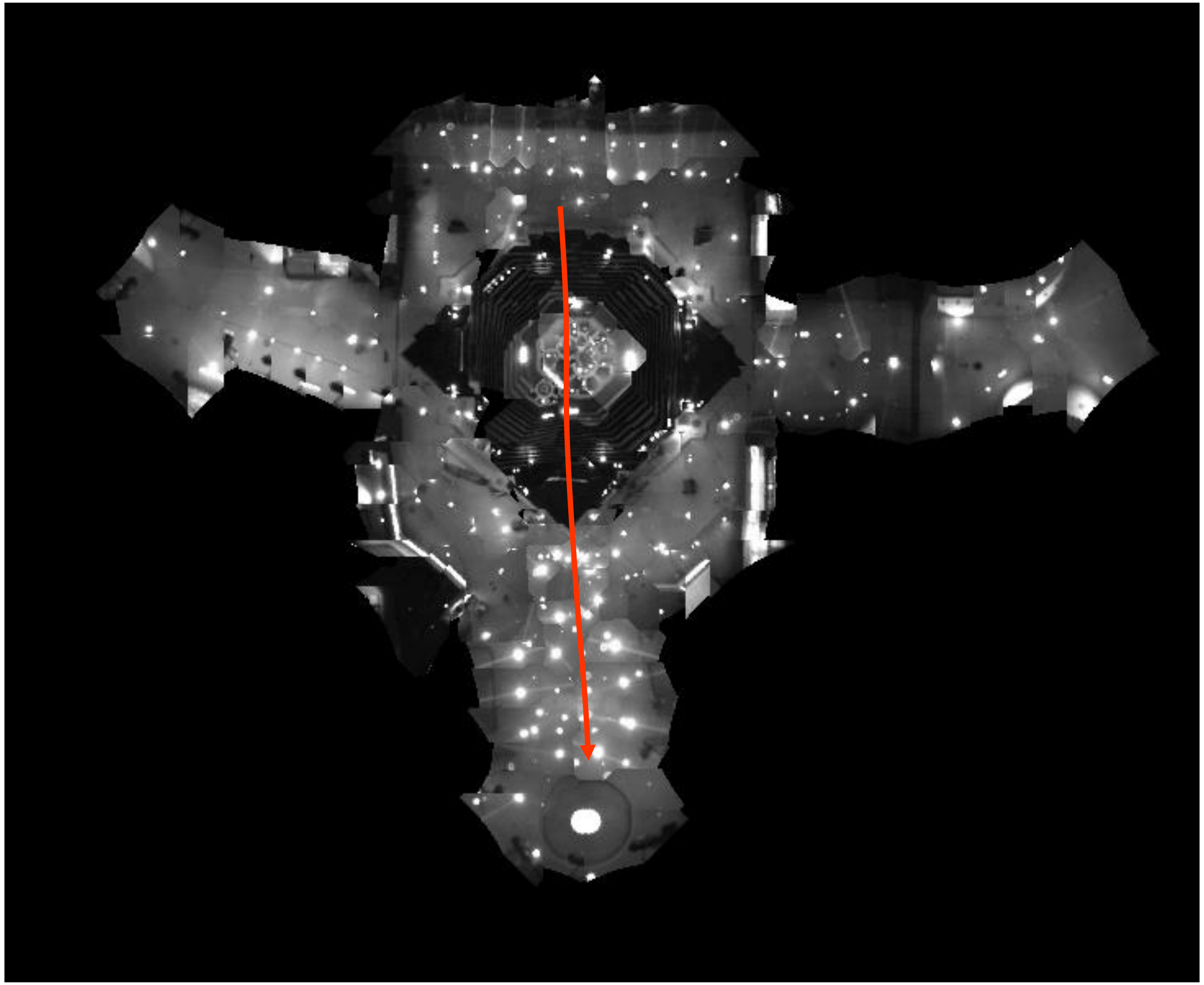
Elsewhere

Measurement z:

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



Global Localization Using Vision



Limitations

- The approach described so far is able to
  - track the pose of a mobile robot and to
  - globally localize the robot.

- Can we deal with localization errors (i.e., the kidnapped robot problem)?
- How to handle localization errors/failures?
  - Particularly serious when the number of particles is small

Approaches

- Randomly insert samples
  - Why?
  - The robot can be teleported at any point in time
- How many particles to add? With what distribution?
  - Add particles according to localization performance
  - Monitor the probability of sensor measurements $p(z_t|z_{1:t-1}, u_{1:t}, m)$
  - For particle filters: $p(z_t|z_{1:t-1}, u_{1:t}, m) \approx \frac{1}{M}\sum w_t^{[m]}$
- Insert random samples proportional to the average likelihood of the particles (the robot has been teleported with higher probability when the likelihood of its observations drops).

Summary

- Particle filters are an implementation of recursive Bayesian filtering
- They represent the posterior by a set of weighted samples.
- In the context of localization, the particles are propagated according to the motion model.
- They are then weighted according to the likelihood of the observations.
- In a re-sampling step, new particles are drawn with a probability proportional to the likelihood of the observation.