Particle Filter

CS 3630

The Particle Filter

Particle filters represent a probability density function as a set of weighted samples.

The weighted samples are

1. Pushed through the motion model (including uncertainty)
2. Reweighted based on sensor measurements (using the sensor model)
3. Resampled using the new weights to define a probability distribution on the sample set.

- The approach is easy to implement, and has low computational overhead.
- Complexity does not grow exponentially with dimension of the state space.

Two localization problems

- "Global" localization
  - Figure out where the robot is, but we don't know where the robot started
  - Sometimes called the "kidnapped robot problem"

- "Position tracking"
  - Figure out where the robot is, given that we know where the robot started

➢ **To solve these problems at time t, we estimate**

$$\boldsymbol{Bel(x_t) = P(x_t | u_1, z_1, u_2 \ \ldots, z_t)}$$

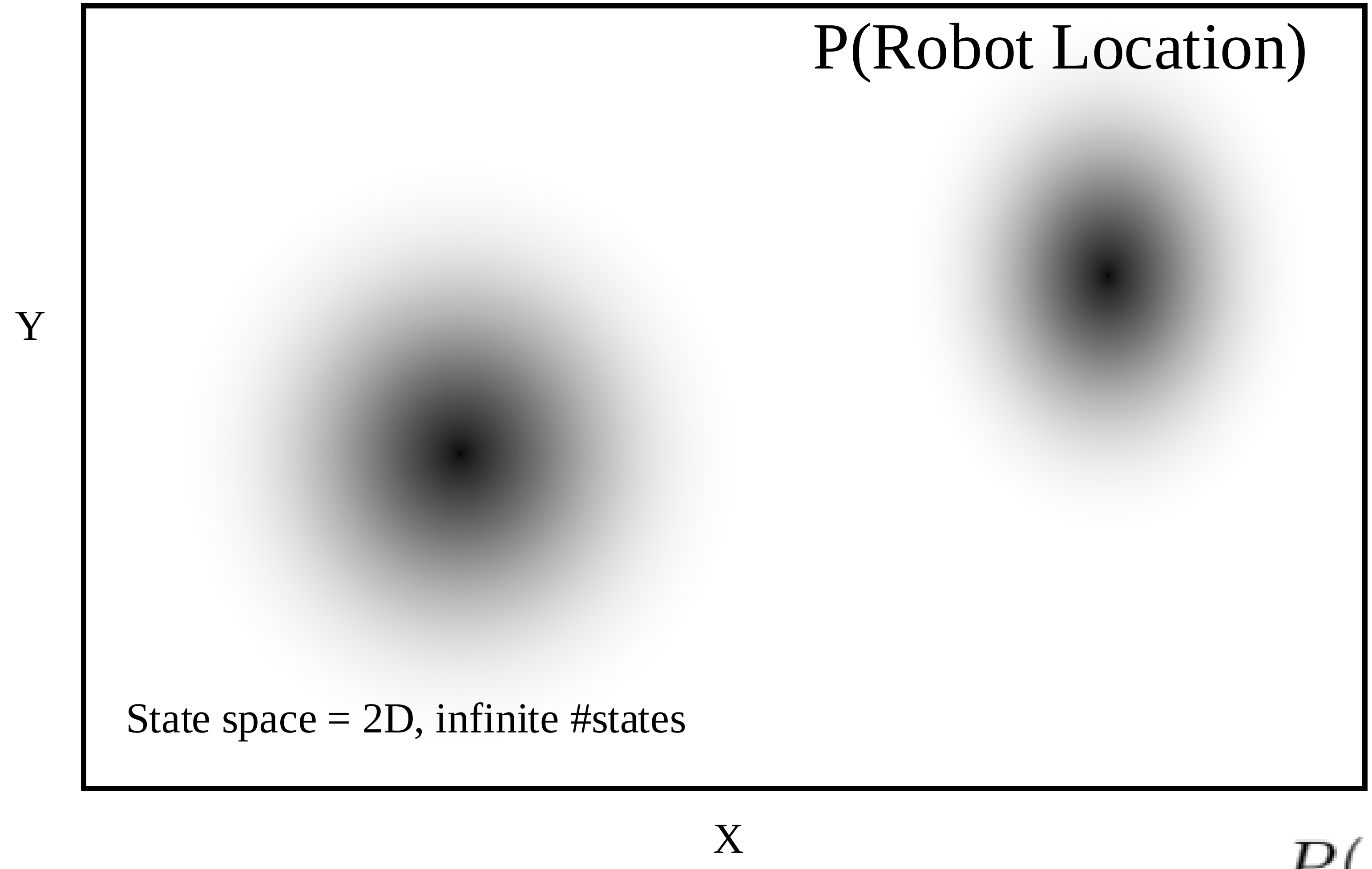
➢ ***The hard part: it's not feasible to exactly calculate or represent $Bel(x_t)$.***

Sampling to Approximate Densities

- Densities can become arbitrarily complex, even when noise models are Gaussian.
- One issue is nonlinear measurement and noise models.
- A second issue is the curse of dimensionality (for grid-based methods).

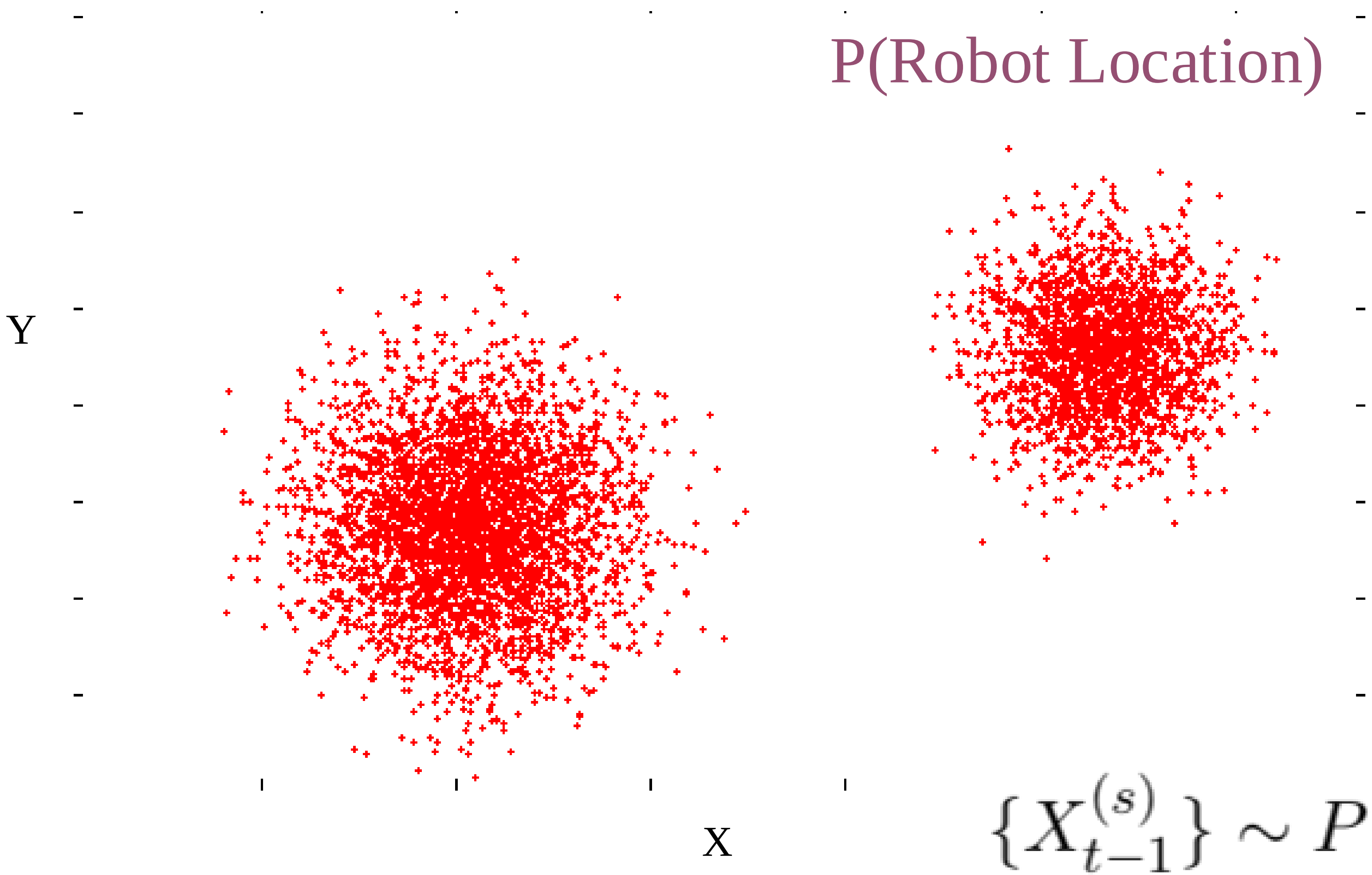
- One way out: ***sampling!***

Probability of Robot Location



$P(X_{t-1}|Z^{t-1})$

Sampling as Representation



$$\{X_{t-1}^{(s)}\} \sim P(X_{t-1} | Z^{t-1})$$

Particle Filter

- Represent *p*(*x*) by set of N weighted, random samples, called *particles,* of the form: $< (x_i, y_i), w_i >$

  (x_i, y_i) represents robot's pose

  w_i represents a weight, where $\sum w_i = 1$

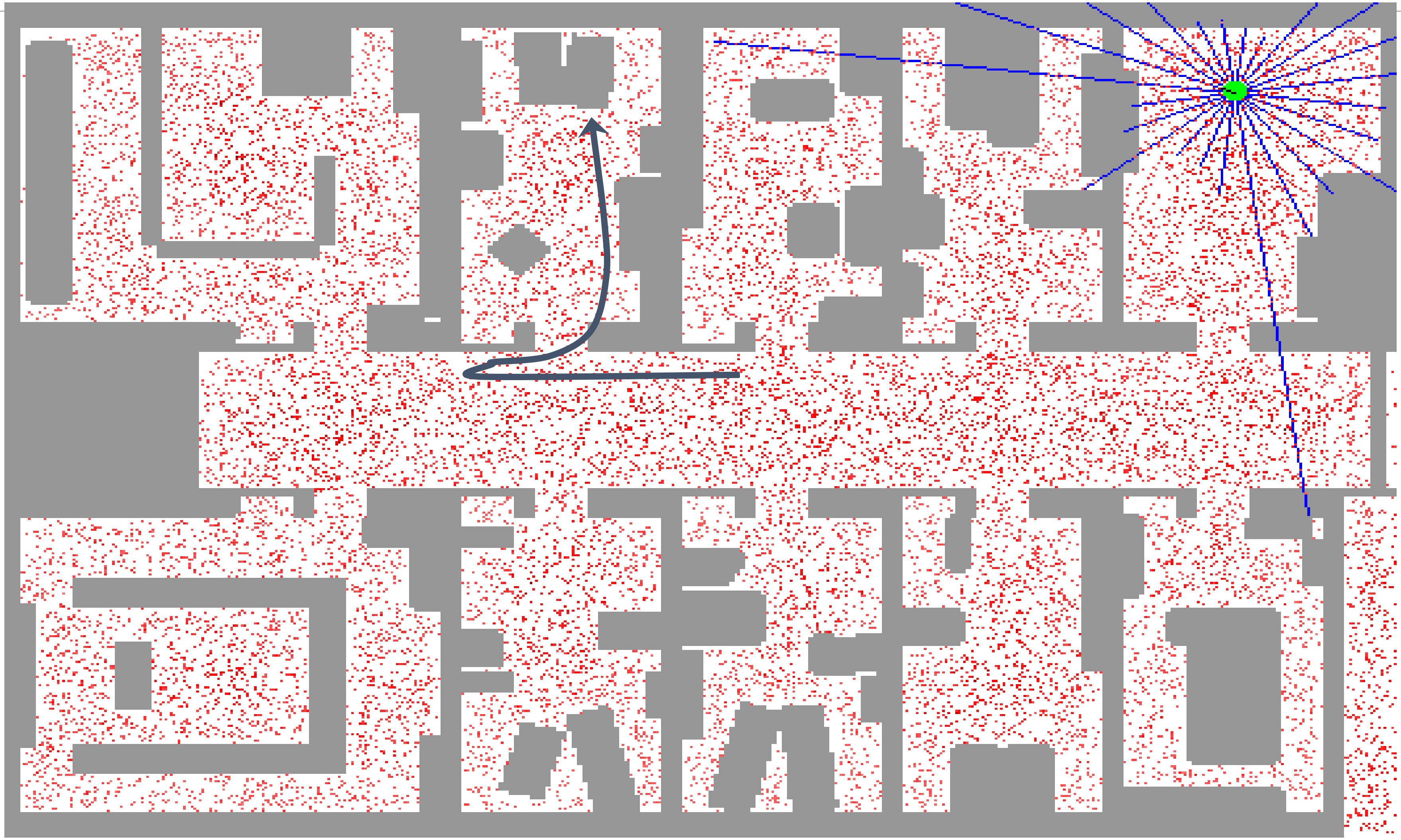
- A.K.A. Monte Carlo Localization (MCL)
  - Refers to techniques that are stochastic (random / non-deterministic)
  - Used in many modeling and simulation approaches

Sampling Advantages

- Arbitrary densities
- Memory = O(#samples)
- Only in “Typical Set”
- Great visualization tool!

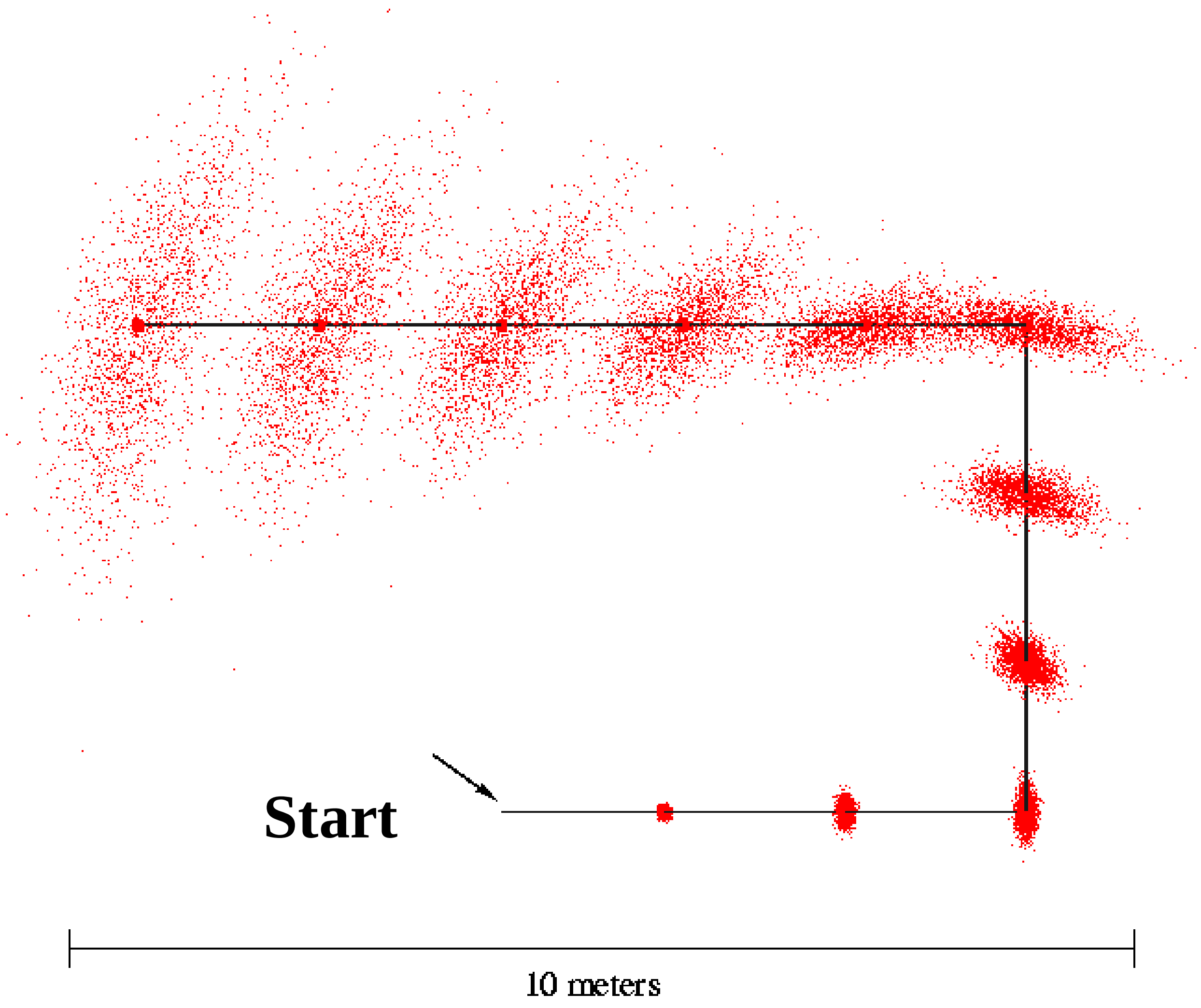
- Weakness: Approximate

Particle Filter Localization (using sonar)

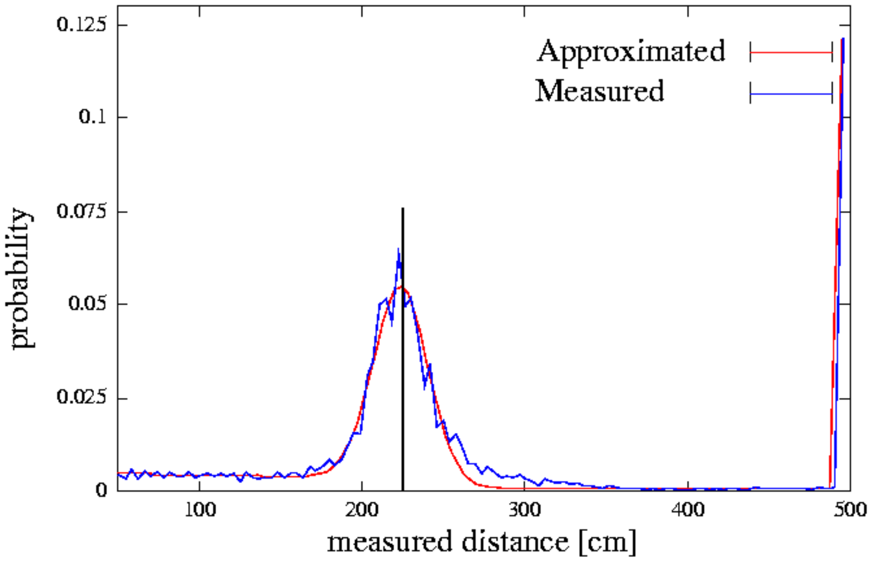


http://www.cs.washington.edu/ai/Mobile_Robotics//mcl/animations/global-floor-start.gif

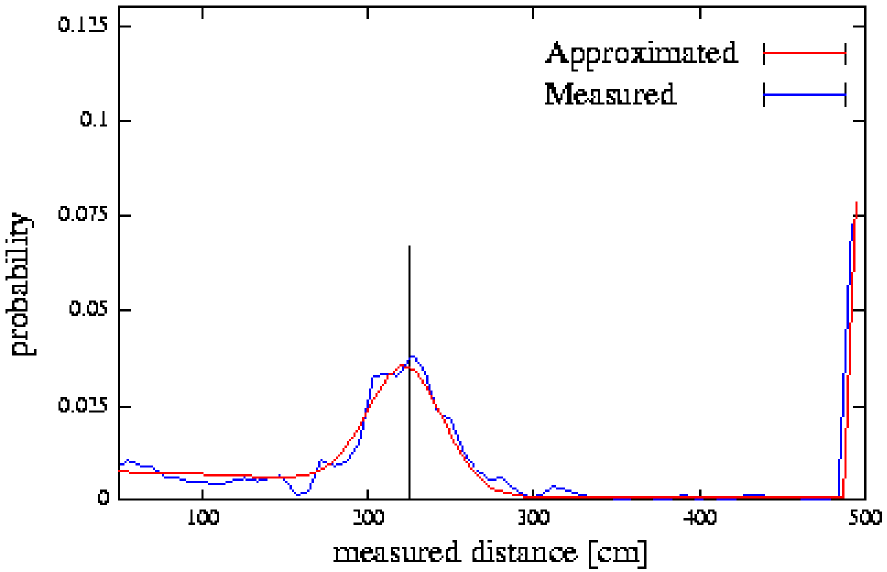
Motion Model for a Car-Like Robot



Sensor Model



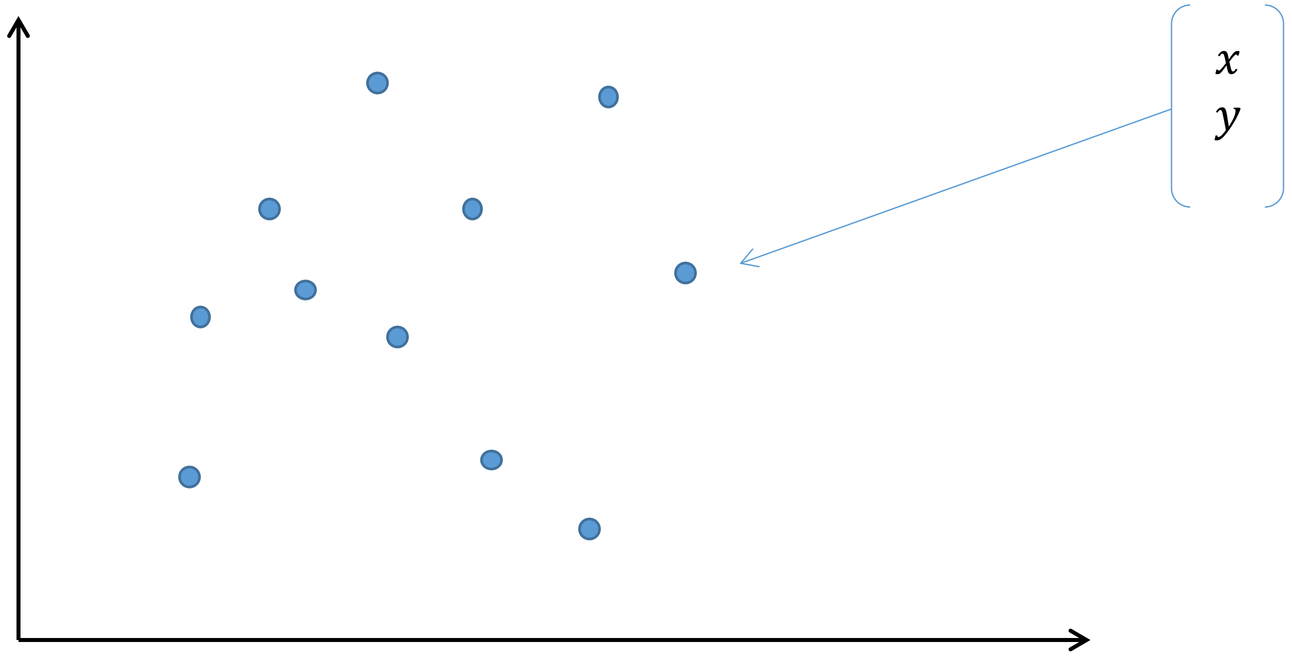
Laser sensor



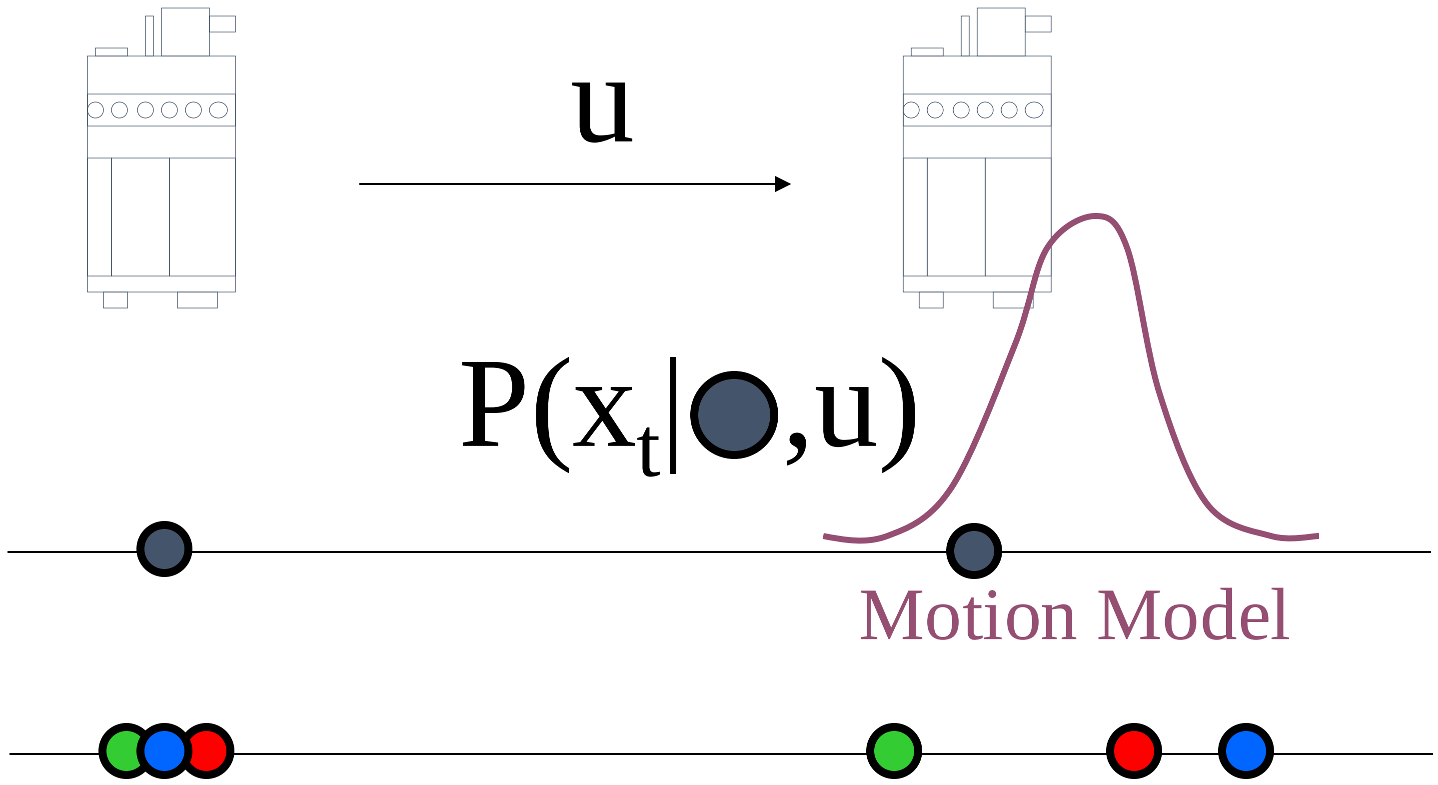
Sonar sensor

Particles

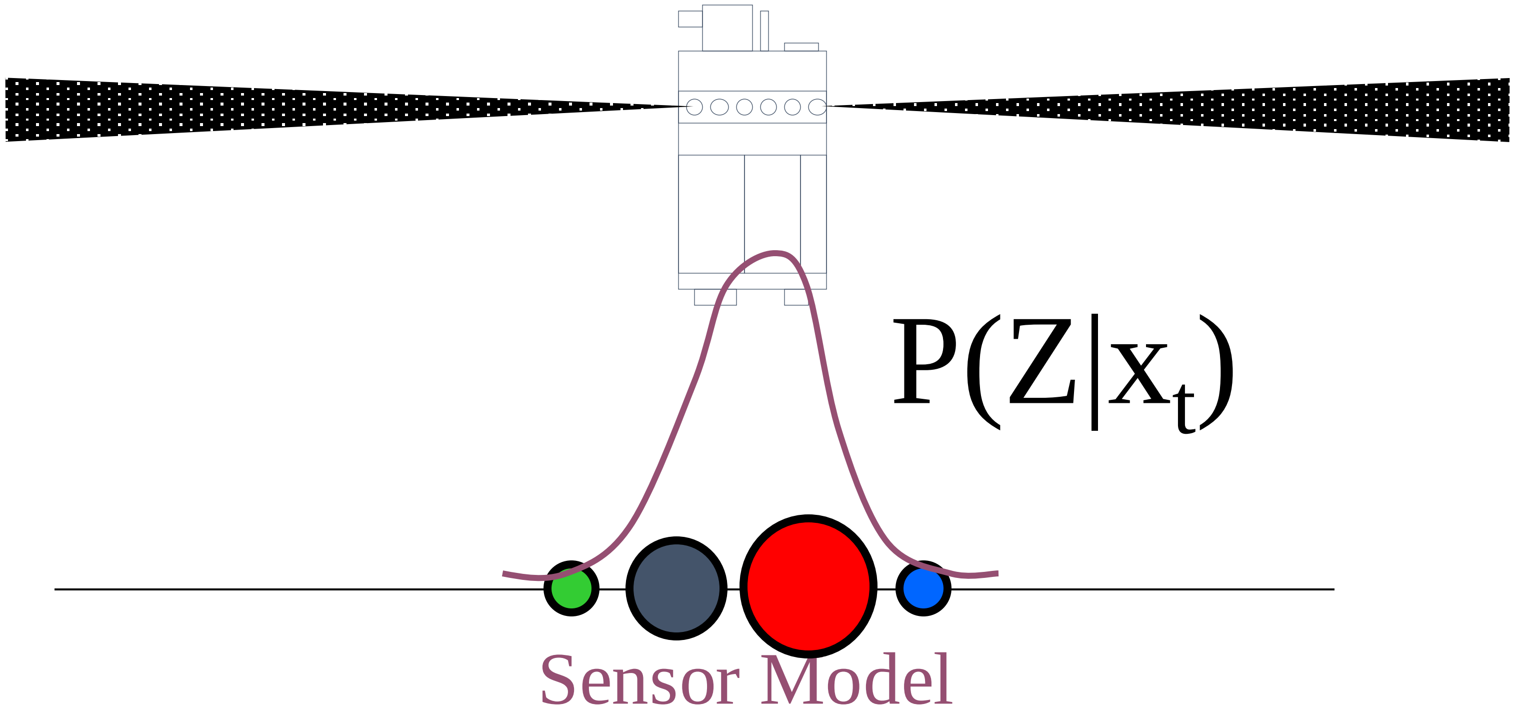
- Each particle is a guess about where the robot might be



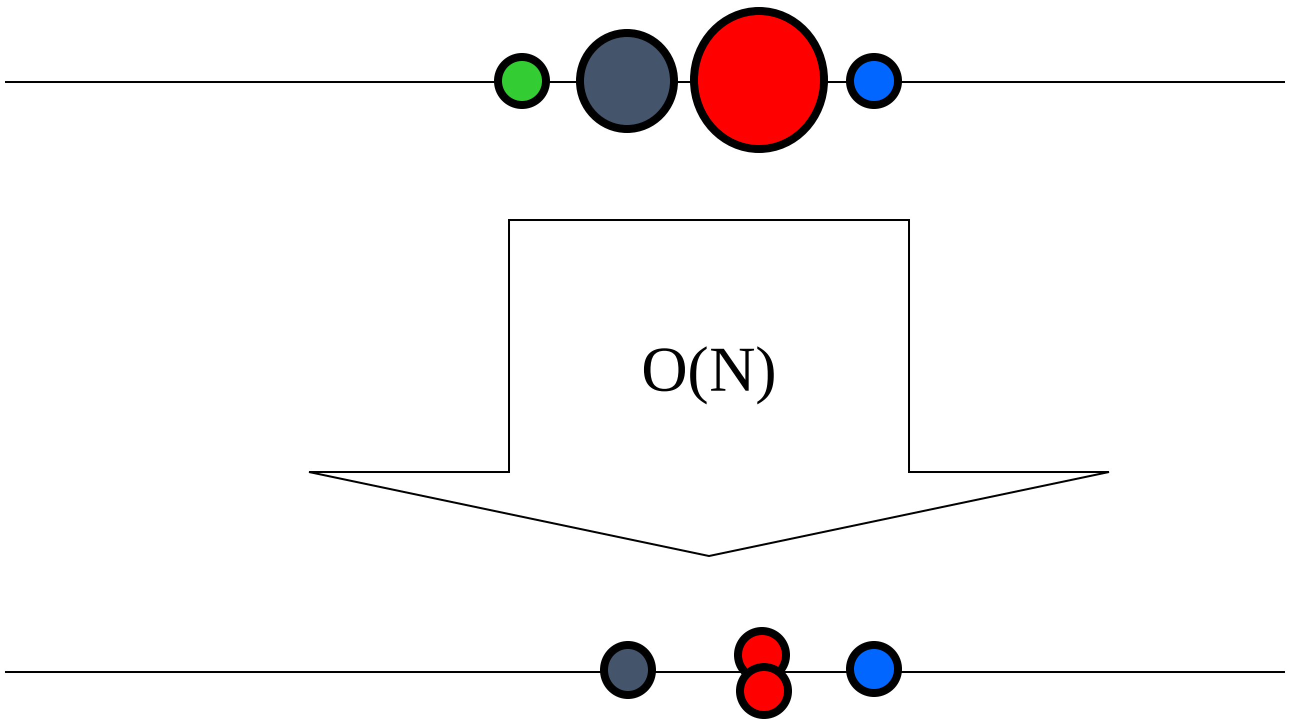
1. Prediction Phase

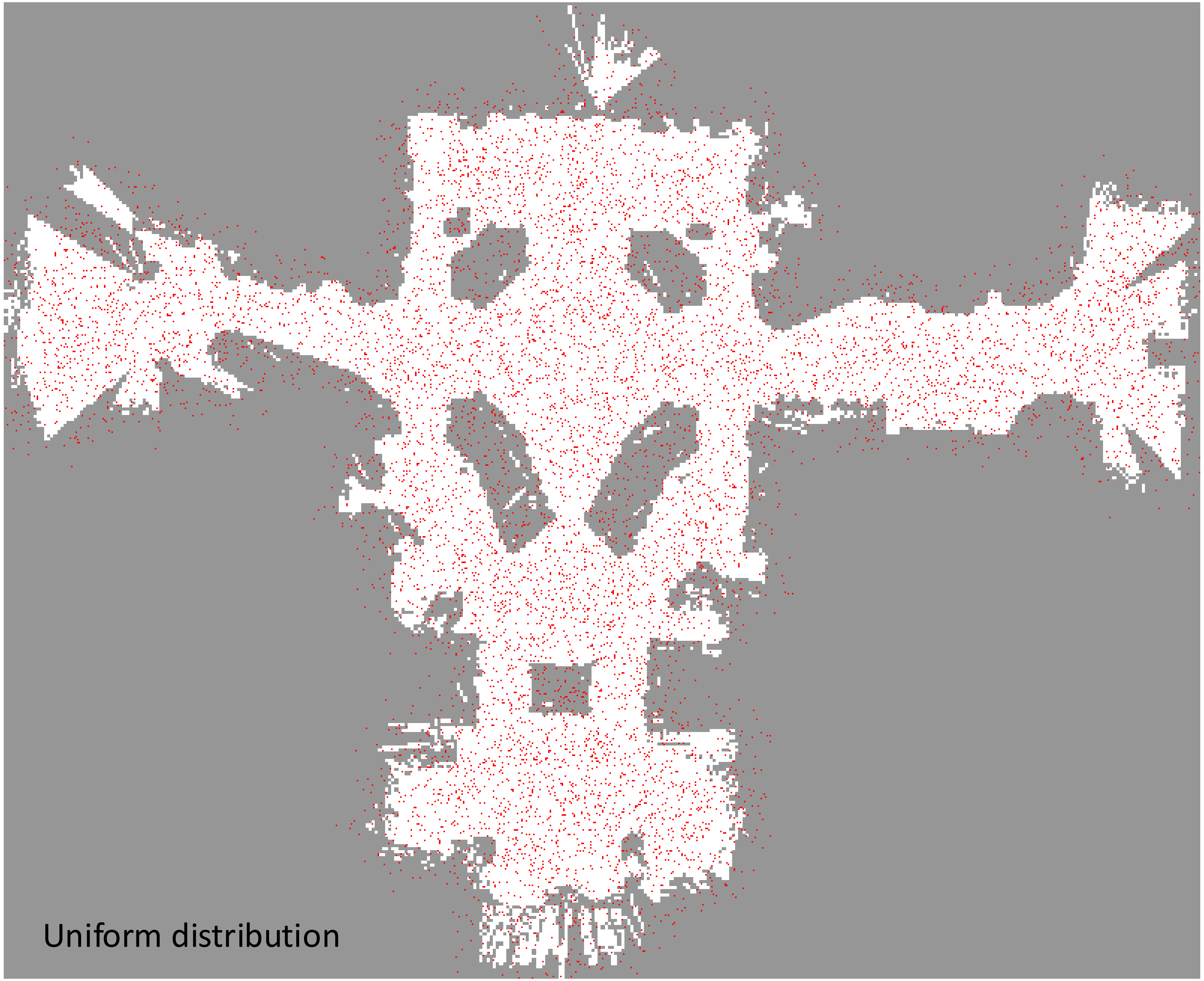


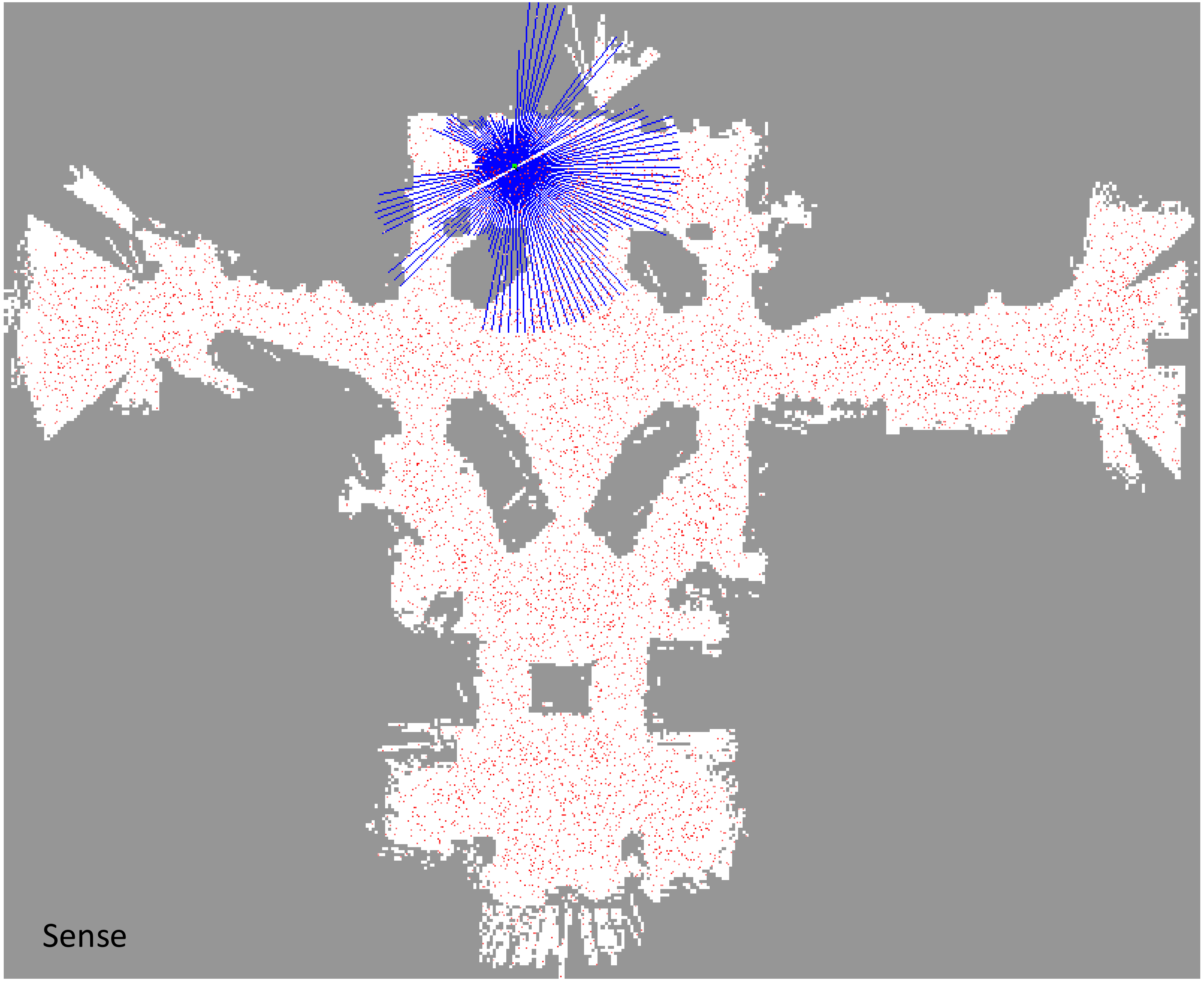
2. Measurement Phase

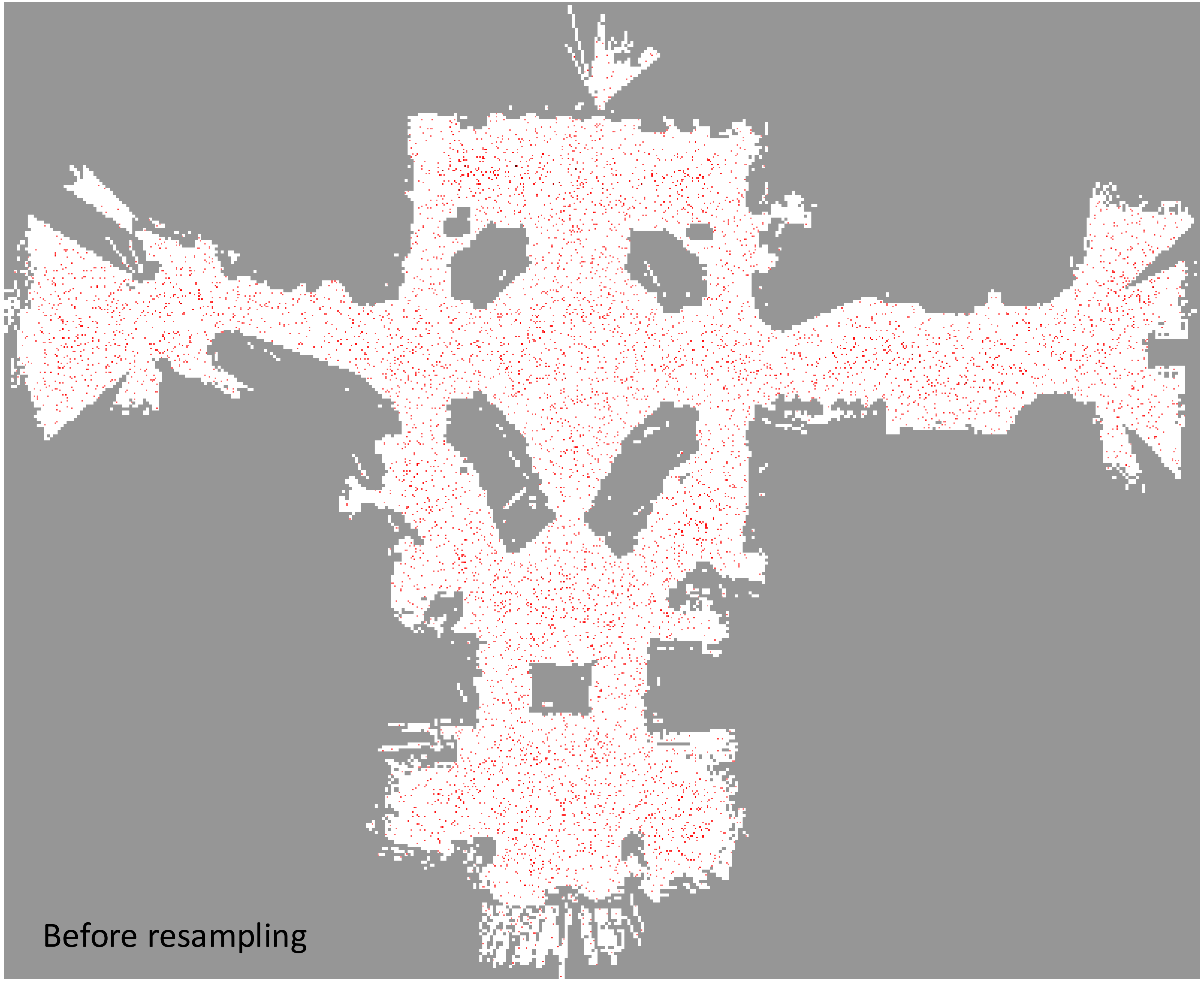


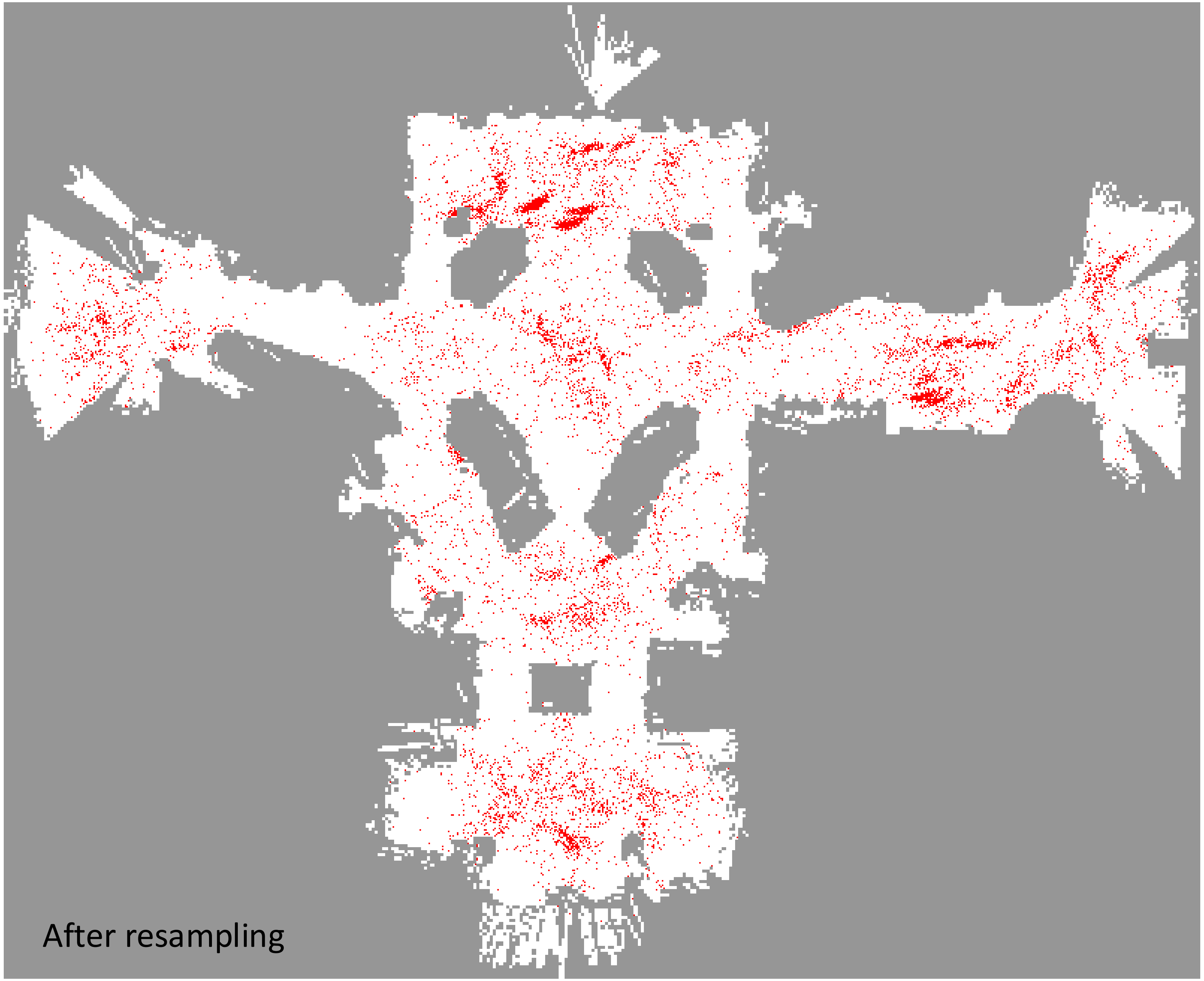
3. Resampling Step

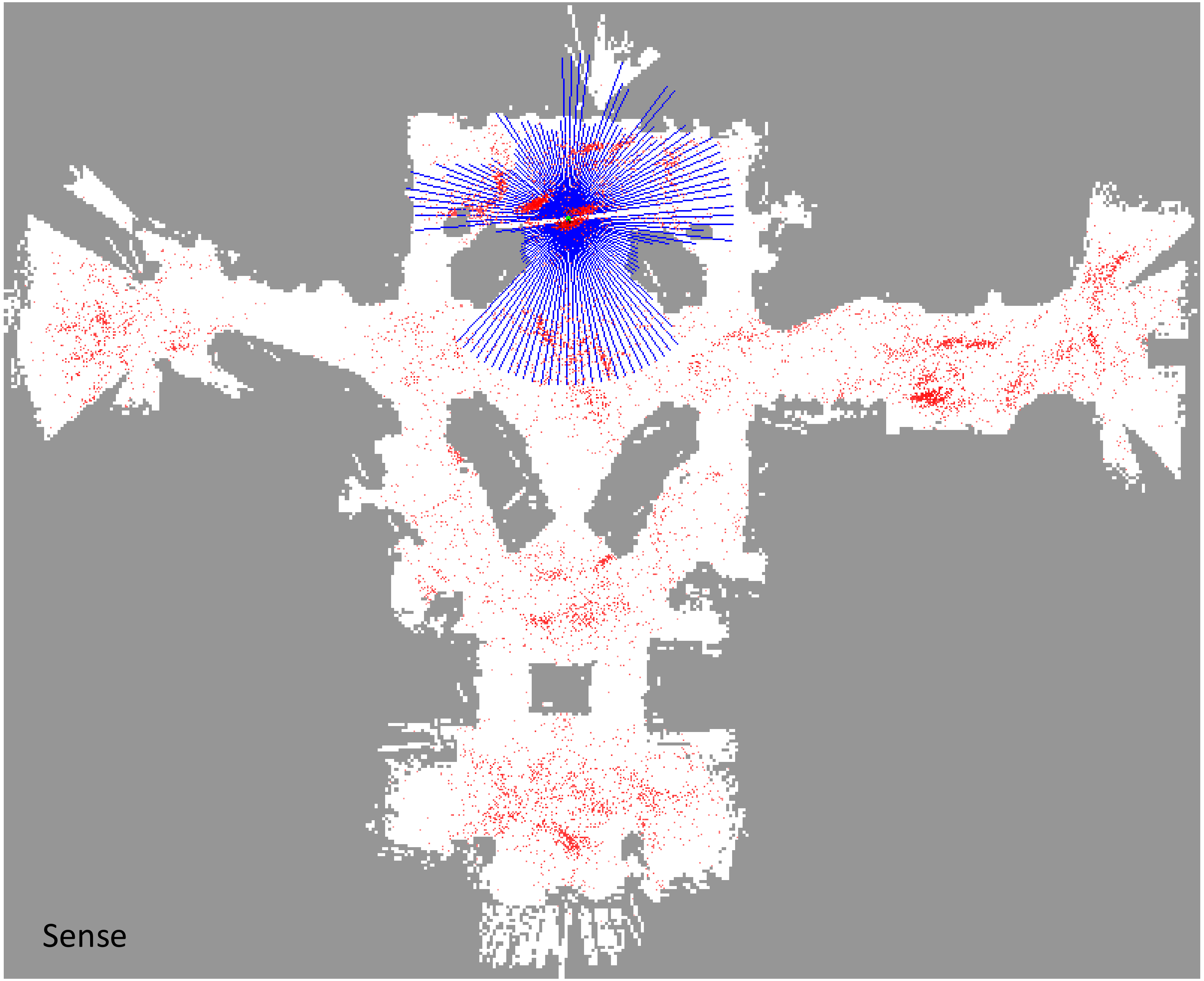


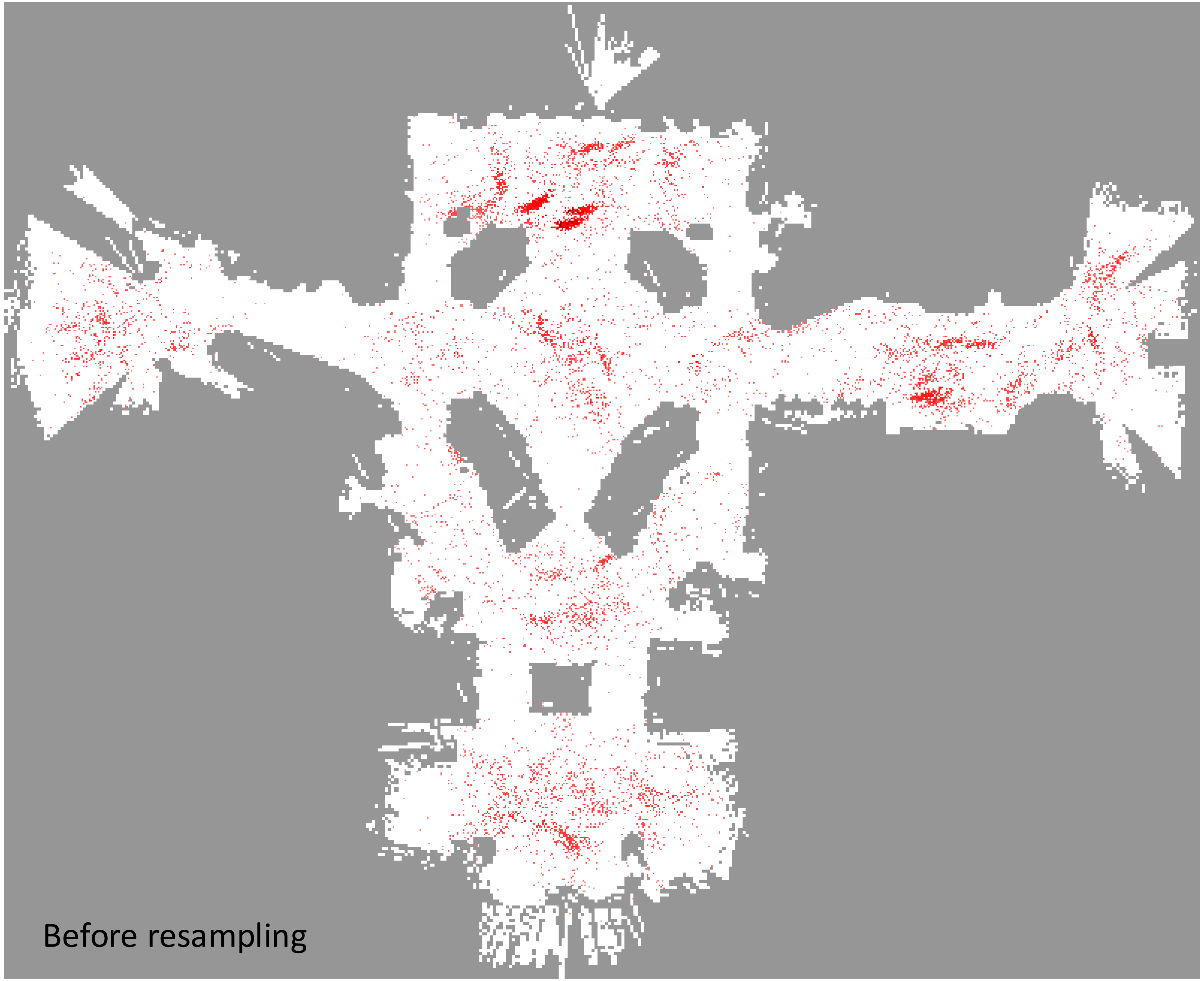

Uniform distribution

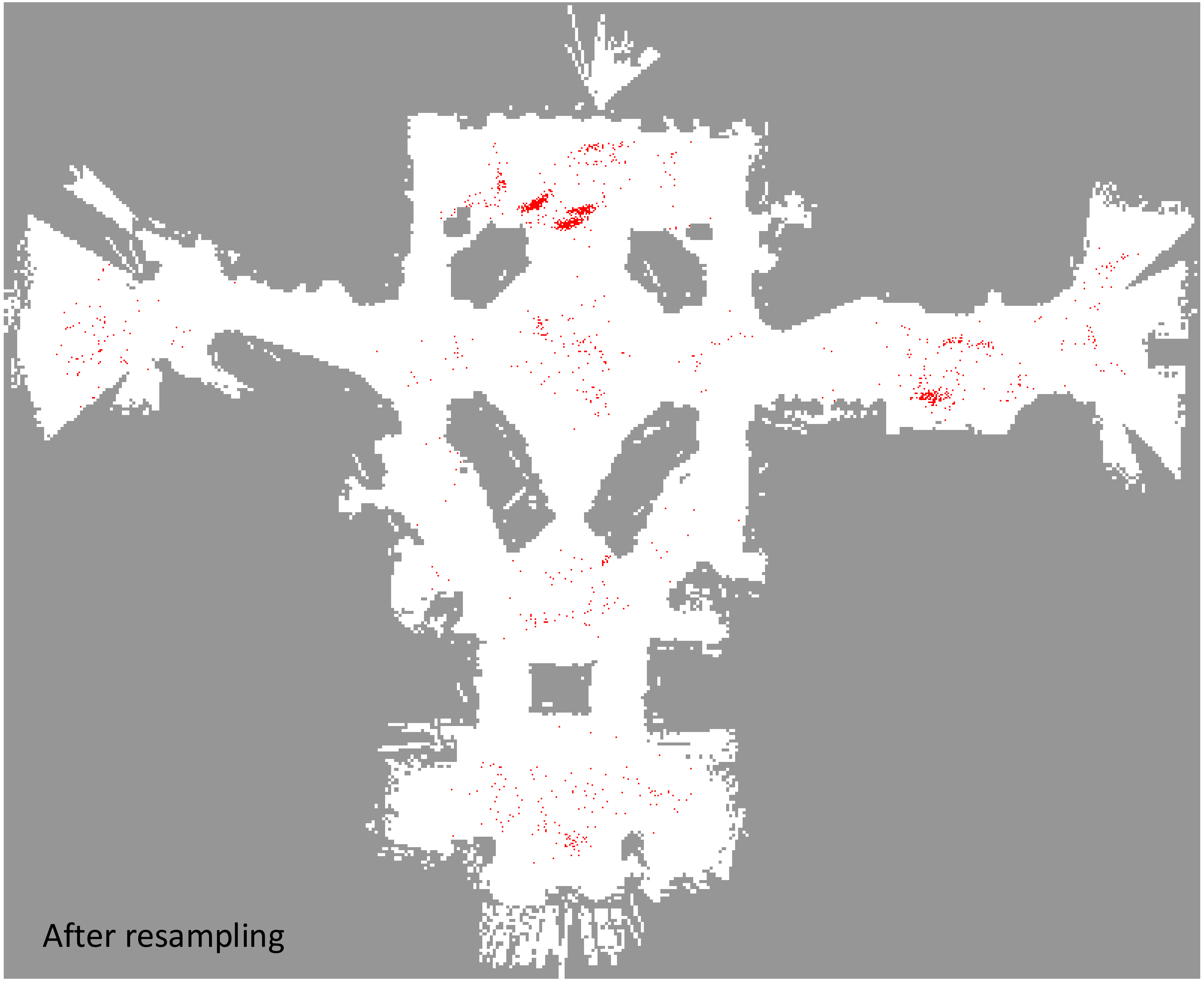

Sense


Before resampling

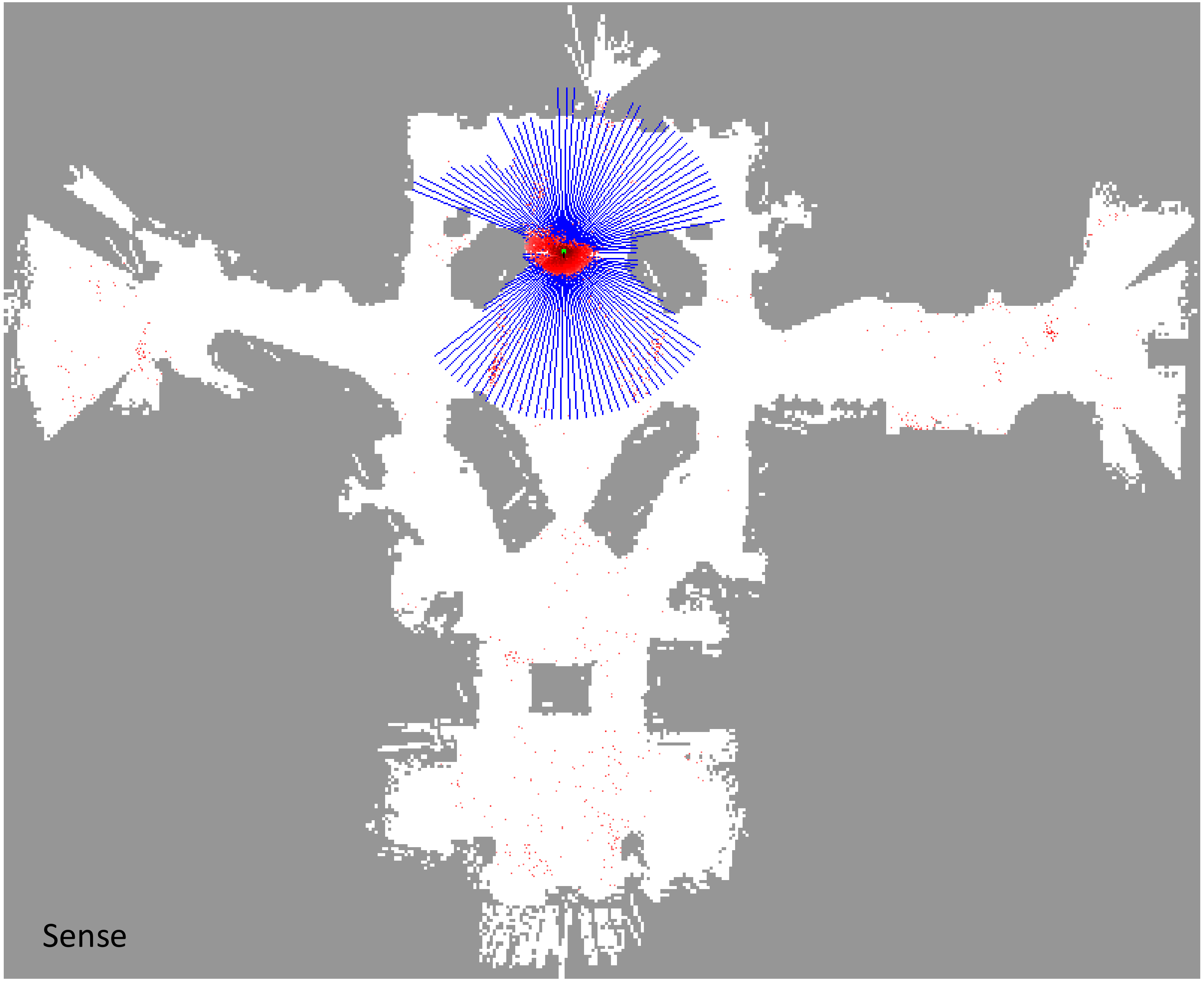

After resampling


Sense

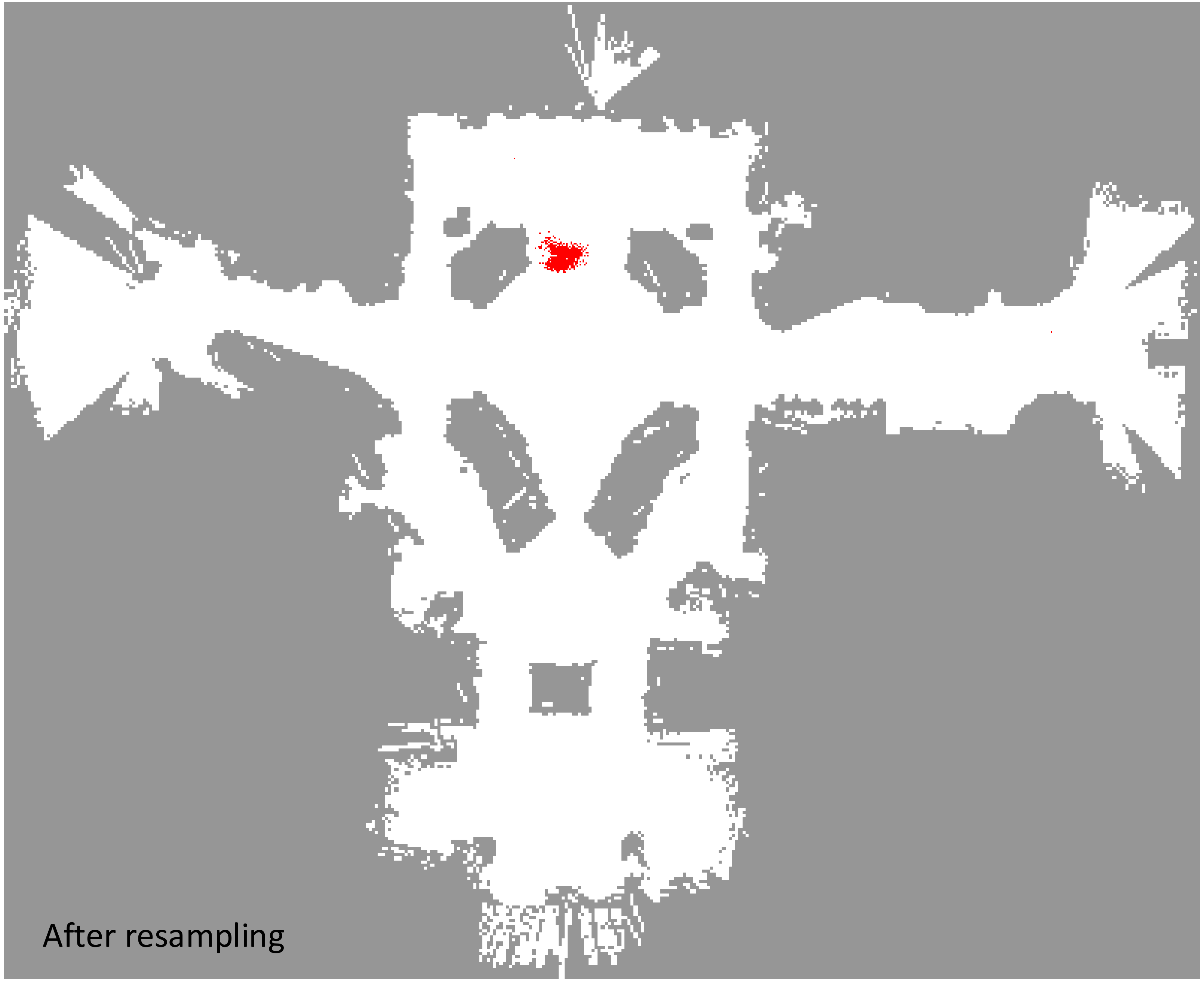

Before resampling

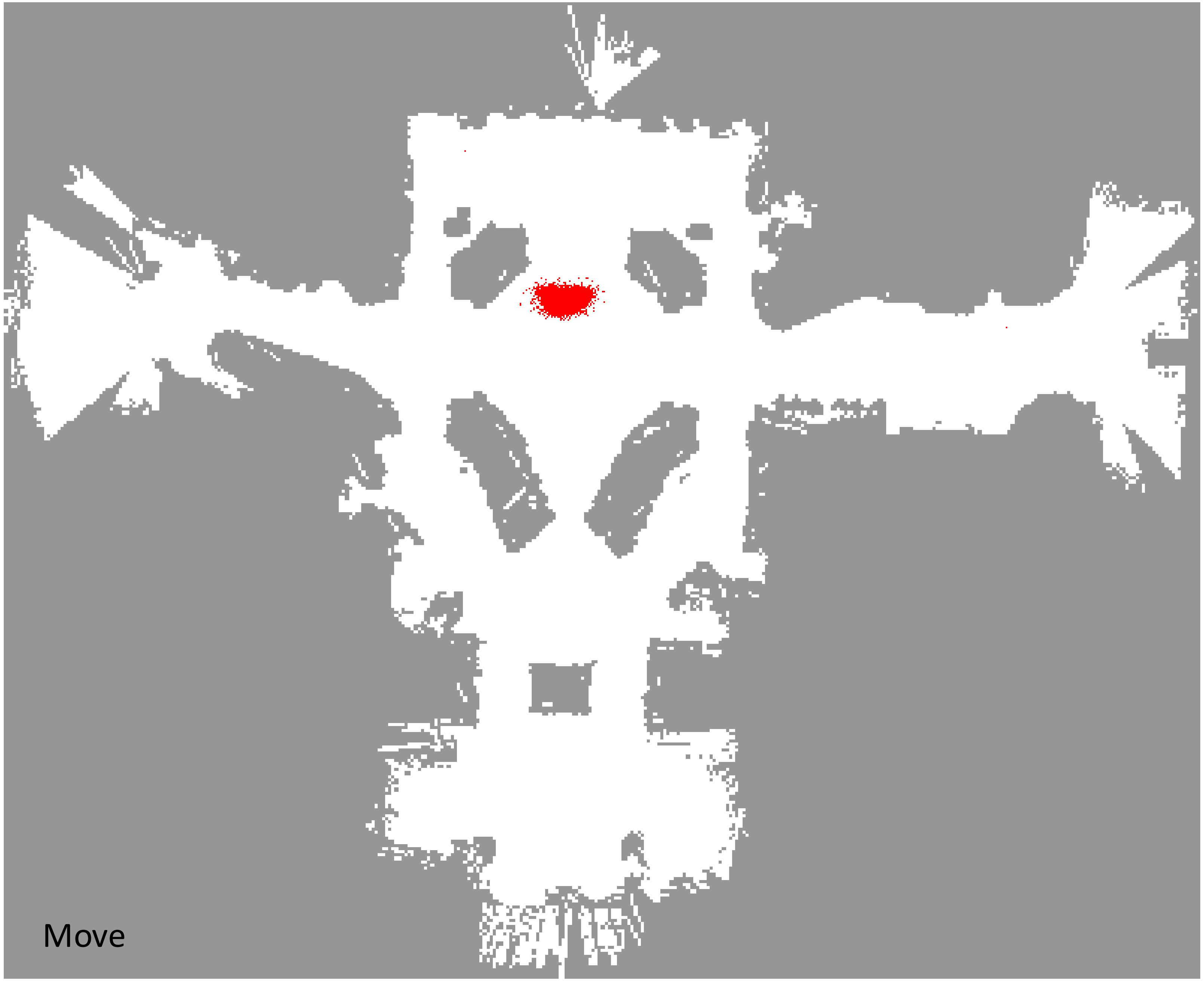

After resampling

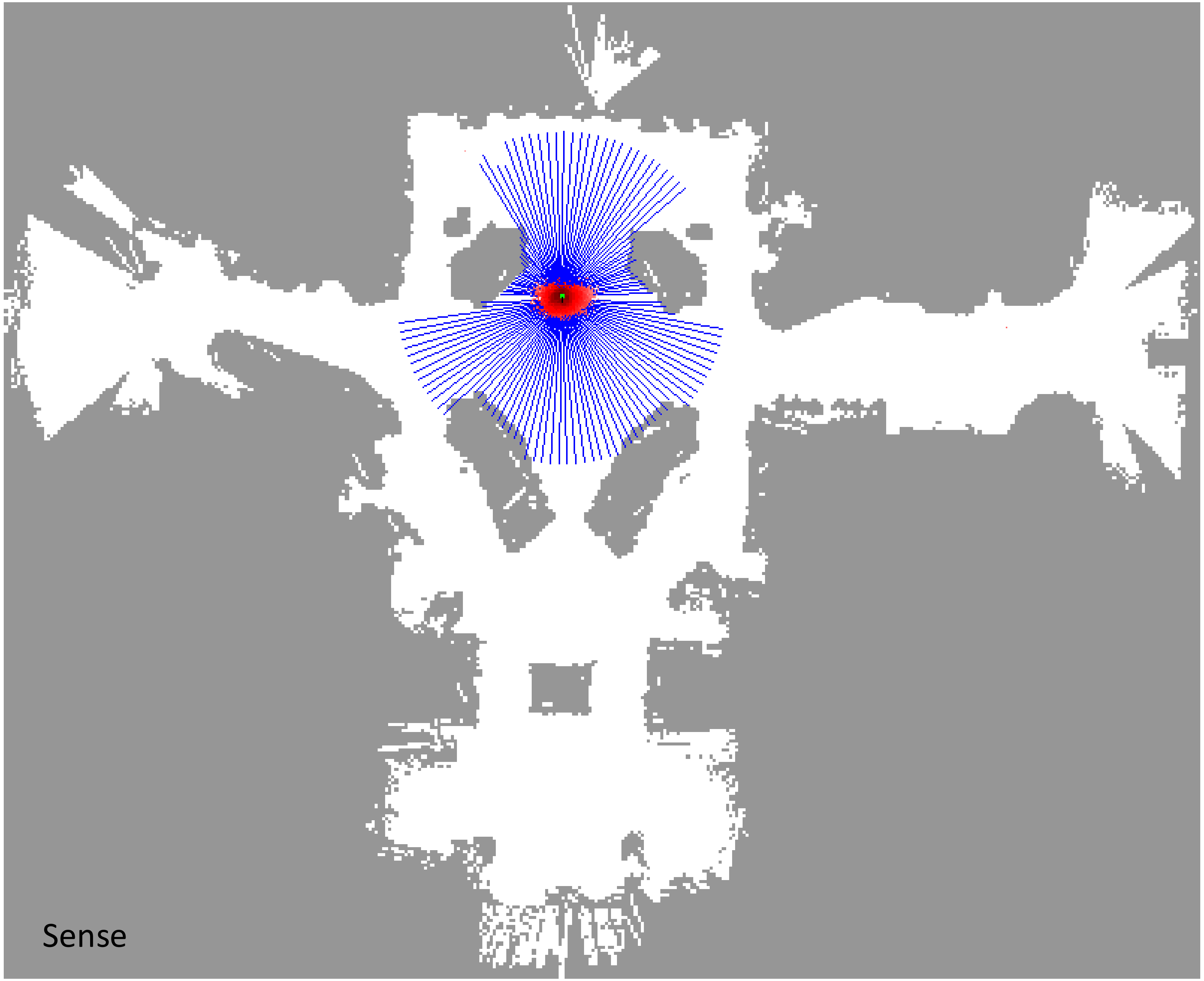

Move

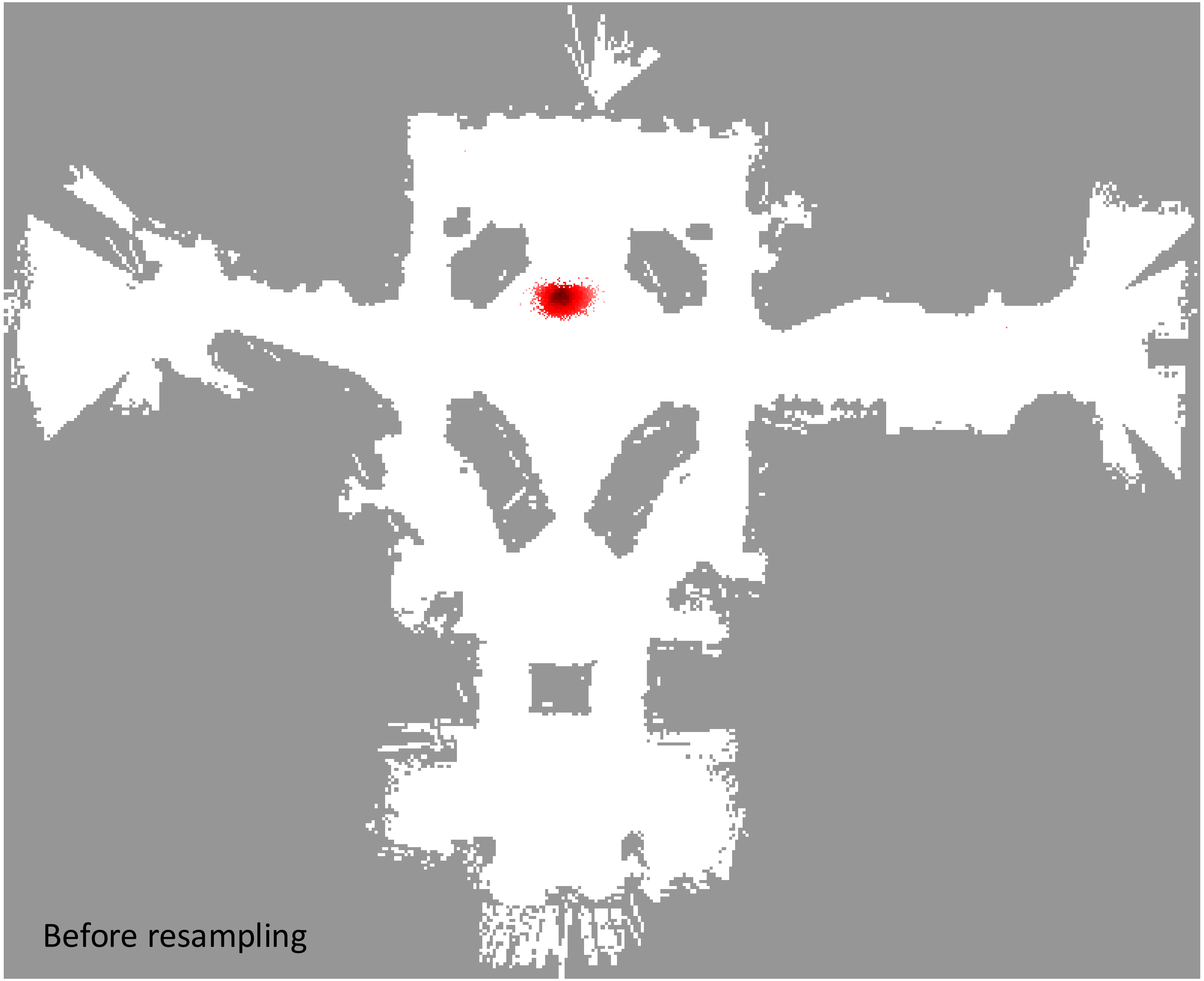

Sense

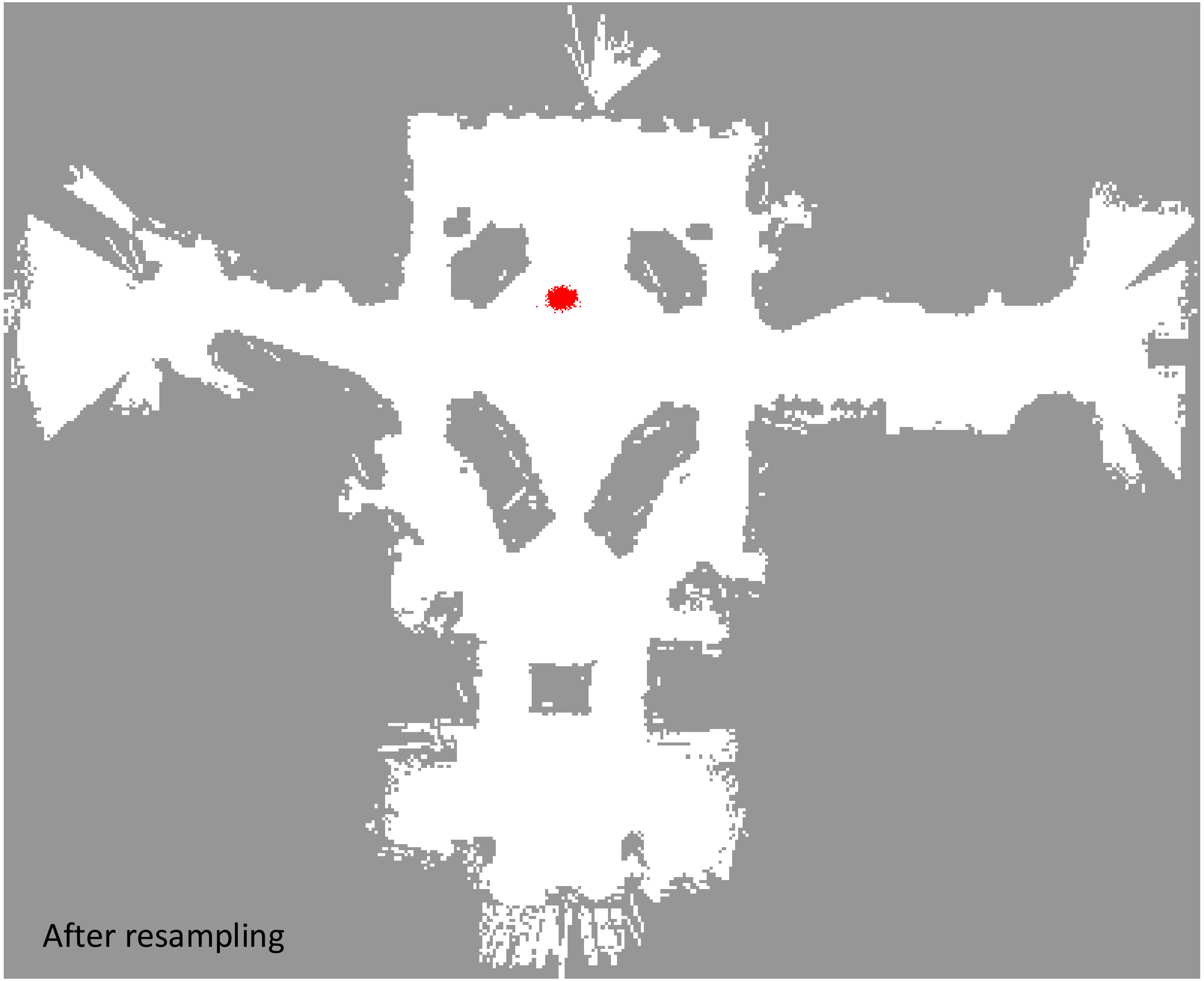

Before resampling

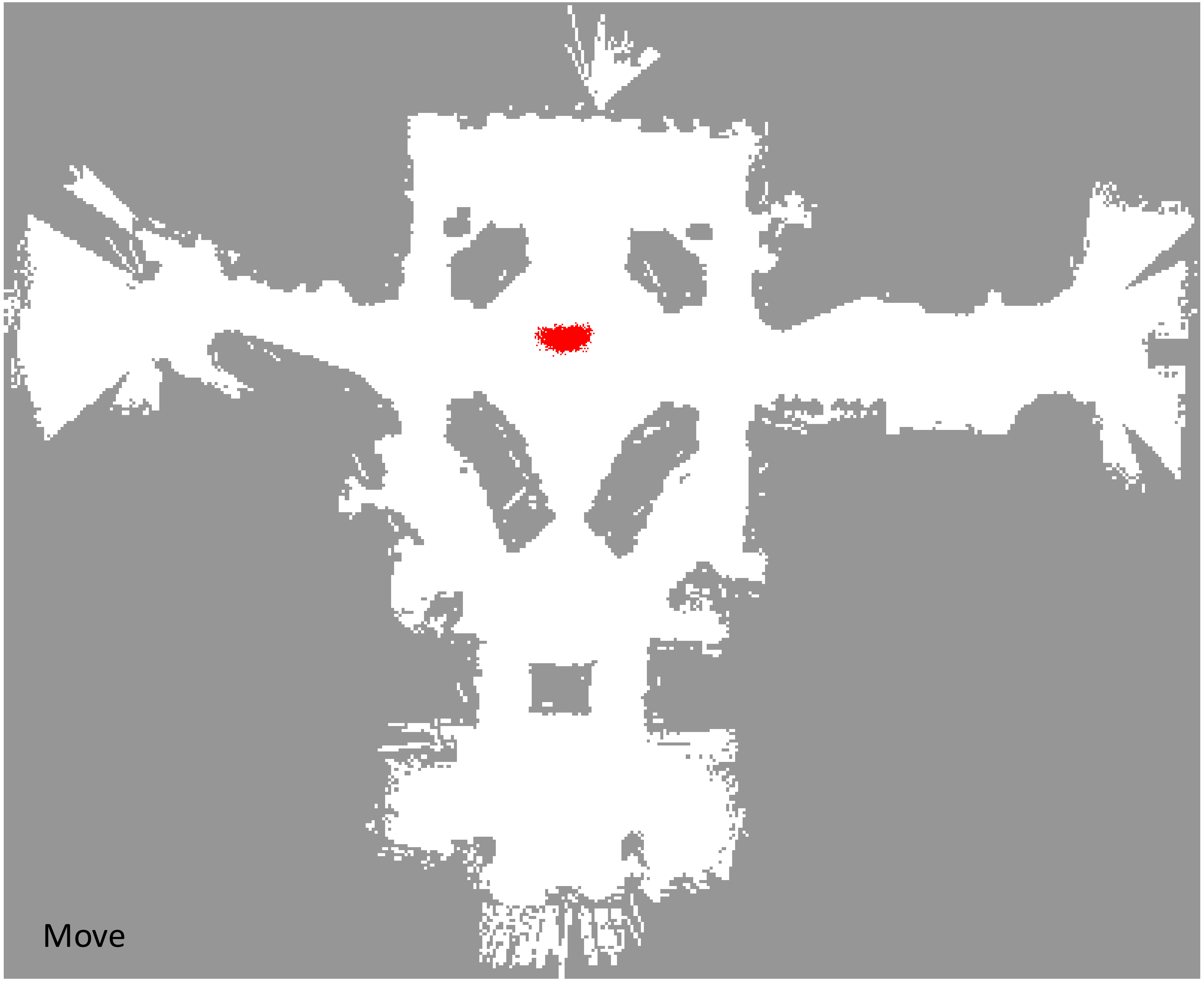

After resampling


Move


Sense

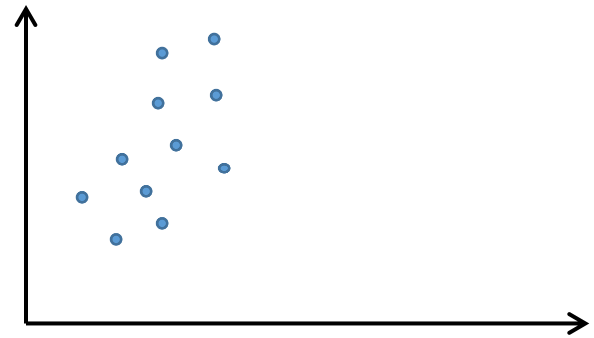

Before resampling


After resampling

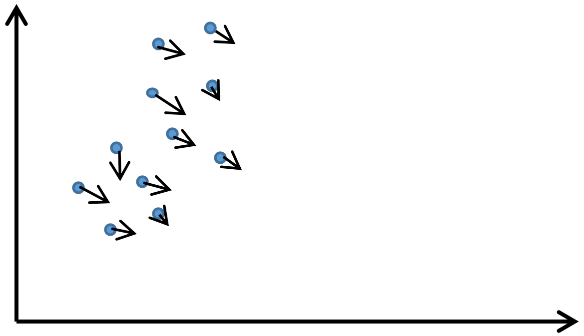

Move

Motion Model

- When the command u_{t-1} is executed, each particle is updated to approximate the robot's movement by ***sampling*** from $p(x_t | x_{t-1}, u_{t-1})$.
- At this stage, typically all particles have equal weight ($w = \frac{1}{N}$).



n particles



n particles

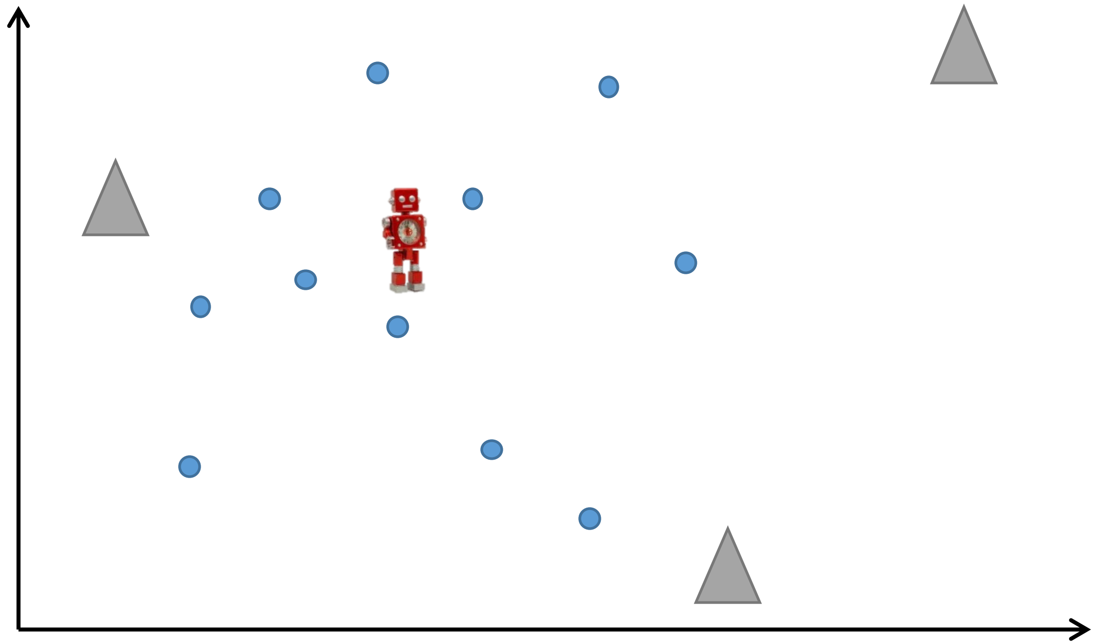
Sensing Model

- ***Re-weight sample set***, according to the likelihood that robot's current sensors match what would be seen at a given location

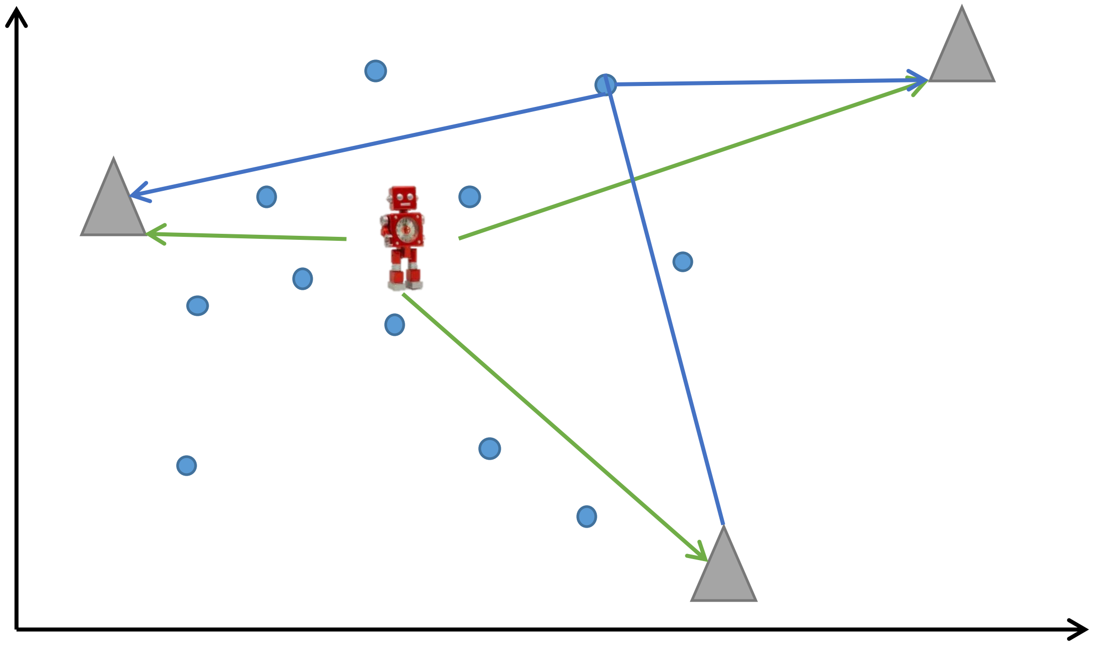
  - Let $< \ x, w >$ be a sample.
  - Then, $w \leftarrow \eta P(z|x)$

    - z is the sensor measurement;
    - η a normalization constant to enforce the sum of w's equaling 1

Incorporating Sensing



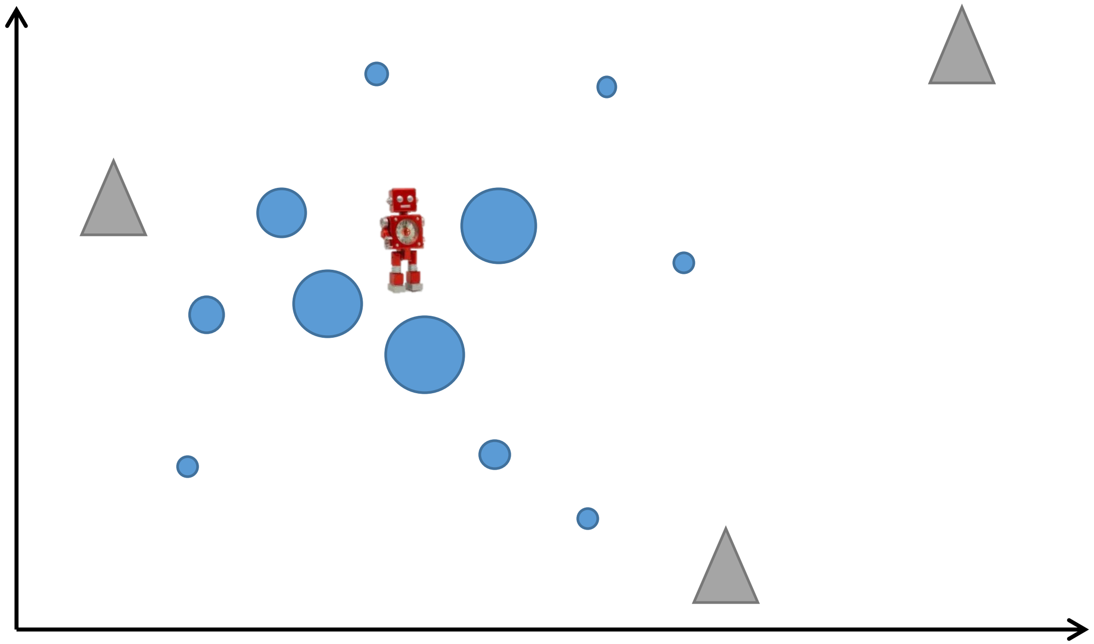
Incorporating Sensing



Difference between the
actual measurement
and the
estimated measurement

Importance weight

Incorporating Sensing



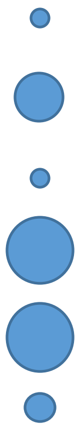
Resampling

- After applying the motion update and sensing update, we end up with new positions and weights for particles

- We want to eliminate particles that have very low weight (unlikely to represent robot position) and generate more particles in the more likely areas of the state space.

- ***Resample***, according to latest weights
- Add a few uniformly distributed, **random samples**
    - Very helpful in case robot completely loses track of its location

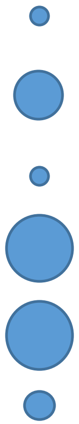
Resampling

n original particles	Importance Weight $w(x_i)$
	0.2 0.6 0.2 0.8 0.8 0.2

$$\sum = 2.8$$

$$\eta = \frac{1}{2.8}$$

Resampling



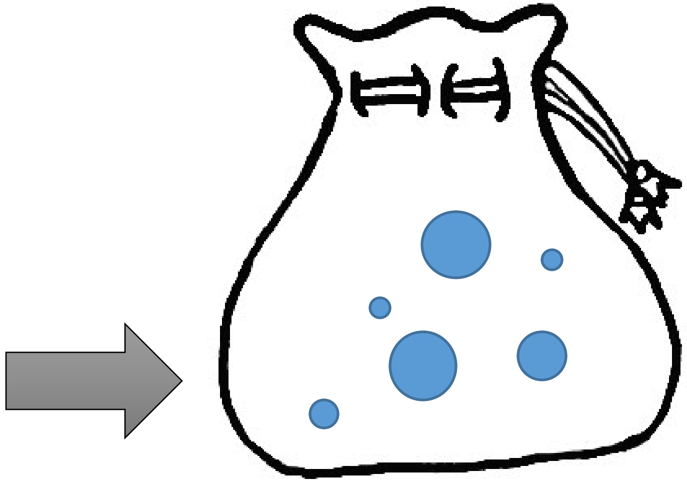
n original particles	Importance Weight $w(x_i)$	Normalized Probability $p(x_i)$
	0.2	0.07
	0.6	0.21
	0.2	0.07
	0.8	0.29
	0.8	0.29
	0.2	0.07

$$\sum = 2.8$$

Resampling

n original particles	Importance Weight $w(x_i)$	Normalized Probability $p(x_i)$
	0.2	0.07
	0.6	0.21
	0.2	0.07
	0.8	0.29
	0.8	0.29
	0.2	0.07

$$\sum = 2.8$$



Sample n new particles from the previous set.

- Each particle is chosen with probability $p(x_i)$, with replacement. Add a little random noise to each resampled particle to avoid identical duplicates.

Resampling

Is it possible that one of the particles is never chosen?

Yes!

Is it possible that one of the particles is chosen more than once?

Yes!

n original particles	Importance Weight $w(x_i)$	Normalized Probability $p(x_i)$
	0.2	0.07
	0.6	0.21
	0.2	0.07
	0.8	0.29
	0.8	0.29
	0.2	0.07

$$\sum = 2.8$$

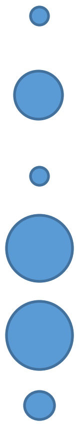
Sample n new particles from the previous set.

- Each particle is chosen with probability $p(x_i)$, <u>with replacement.</u>

Resampling

What is the probability that this particle is not chosen during the resampling of the six new particles?

$$(0.71)^6 = 0.13$$



n original particles	Importance Weight $w(x_i)$	Normalized Probability $p(x_i)$
	0.2	0.07
	0.6	0.21
	0.2	0.07
	0.8	0.29
	0.8	0.29
	0.2	0.07

$$\sum = 2.8$$

Sample n new particles from the previous set.

- Each particle is chosen with probability $p(x_i)$, <u>with replacement.</u>

Resampling

What is the probability that this particle is not chosen during the resampling of the six new particles?

$$(0.93)^6 = .65$$

n original particles	Importance Weight $w(x_i)$	Normalized Probability $p(x_i)$
	0.2	0.07
	0.6	0.21
	0.2	0.07
	0.8	0.29
	0.8	0.29
	0.2	0.07

$$\sum = 2.8$$

Sample n new particles from the previous set.

- Each particle is chosen with probability $p(x_i)$, <u>with replacement.</u>