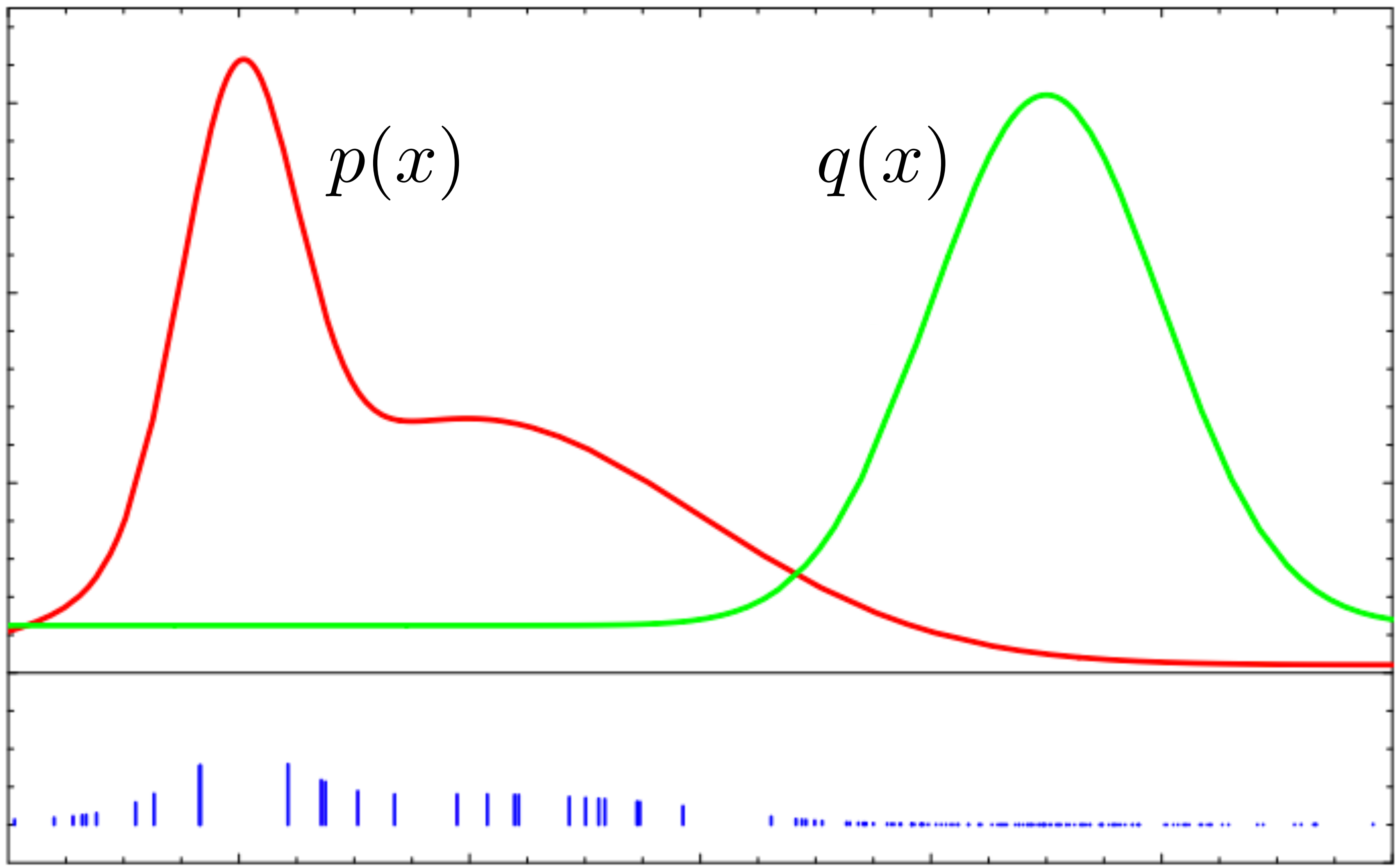
(Slides adapted from Drew Bagnell, Dieter Fox)

Particle Filtering

Sanjiban Choudhury

TAs: Matthew Rockett, Gilwoo Lee, Matt Schmittle

Question: What makes a good proposal distribution?



Applying importance sampling to Bayes filtering

Target distribution : Posterior

$$bel(x_t) = \eta P(z_t|x_t) \int p(x_t|u_t, x_{t-1}) bel(x_{t-1}) dx_{t-1}$$

Applying importance sampling to Bayes filtering

Target distribution : Posterior

$$bel(x_t) = \eta P(z_t|x_t) \int p(x_t|u_t, x_{t-1}) bel(x_{t-1}) dx_{t-1}$$

Proposal distribution : After applying motion model

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

Why is this easy to sample from?

Applying importance sampling to Bayes filtering

Target distribution : Posterior

$$bel(x_t) = \eta P(z_t|x_t) \int p(x_t|u_t, x_{t-1}) bel(x_{t-1}) dx_{t-1}$$

Proposal distribution : After applying motion model

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

Why is this easy to sample from?

Importance Ratio:

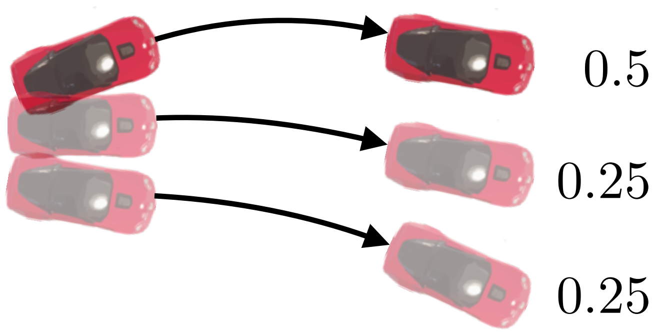
$$r = \frac{bel(x_t)}{\overline{bel}(x_t)} = \eta P(z_t|x_t)$$

Question: What are our options for non-parametric belief representations?

1. Histogram filter
2. Normalized importance sampling
3. Particle filter

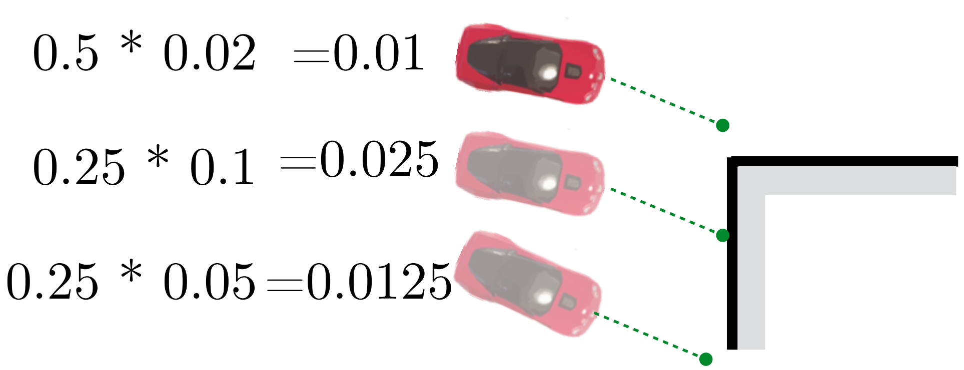
Approach 2: Normalized Importance Sampling

$$bel(x_{t-1}) = \left\{ \begin{matrix} x_{t-1}^1 \\ w_{t-1}^1 \end{matrix}, \begin{matrix} x_{t-1}^2 \\ w_{t-1}^2 \end{matrix}, \cdots, \begin{matrix} x_{t-1}^M \\ w_{t-1}^M \end{matrix} \right\}$$



for $i = 1$ to M

sample $\overline{x}_t^i \sim P(x_t | {\color{blue}u_t}, x_t^i)$



for $i = 1$ to M

$$w_t^i = P(z_t | \bar{x}_t^i) w_{t-1}^i$$

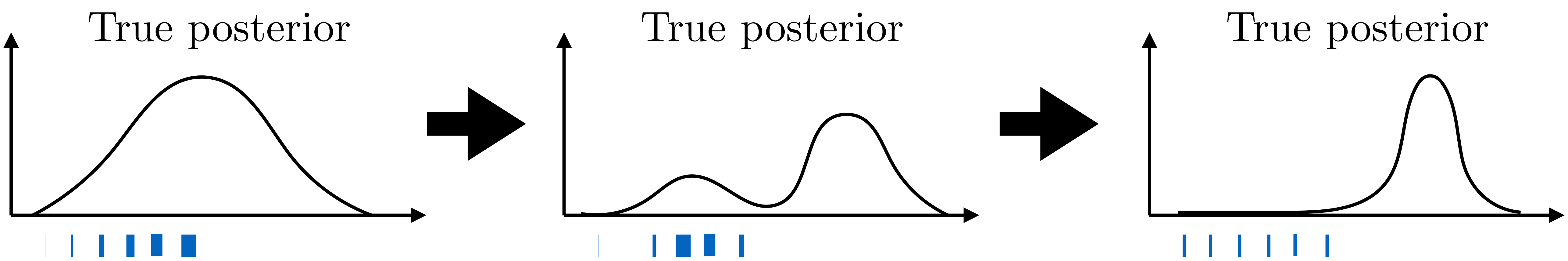
for $i = 1$ to M

$$w_t^i = \frac{w_t^i}{\sum_i w_t^i} \qquad bel(x_t) = \left\{ \begin{matrix} \bar{x}_t^1 \\ w_t^1 \end{matrix}, \cdots, \begin{matrix} \bar{x}_t^M \\ w_t^M \end{matrix} \right\}$$

Problem: What happens after enough iterations?

Problem: What happens after enough iterations?

Particles don't move - can get stuck in regions of low probability



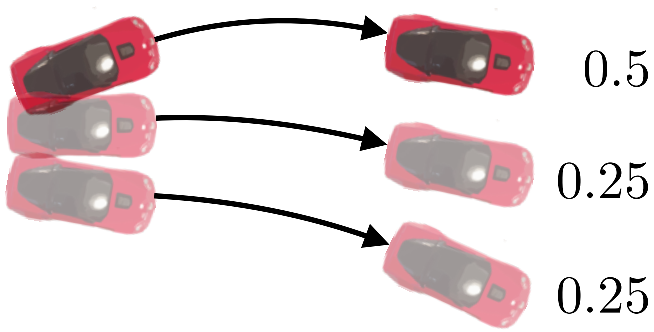
This is the same complaint we had about histogram filters!

Key Idea: Resample!

Why? Get rid of bad particles

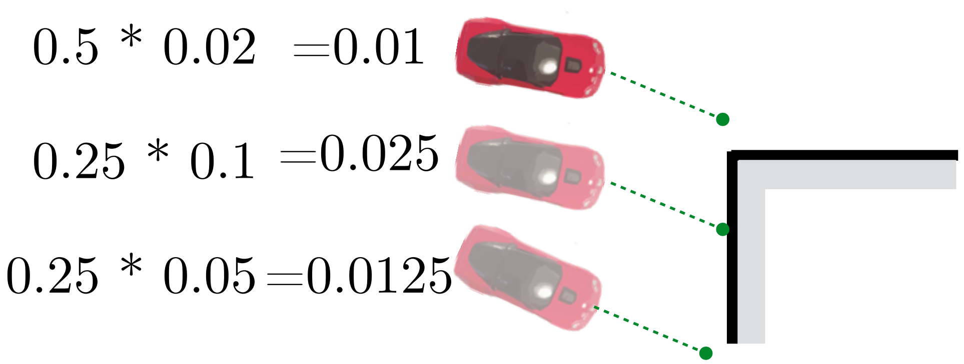
Approach 3: Particle Filtering

$$bel(x_{t-1}) = \left\{ \begin{matrix} x_{t-1}^1 \\ w_{t-1}^1 \end{matrix}, \begin{matrix} x_{t-1}^2 \\ w_{t-1}^2 \end{matrix}, \cdots, \begin{matrix} x_{t-1}^M \\ w_{t-1}^M \end{matrix} \right\}$$



for $i = 1$ to M

sample $\overline{x}_t^i \sim P(x_t | u_t, x_t^i)$



for $i = 1$ to M

$$w_t^i = P(z_t | \bar{x}_t^i) w_{t-1}^i$$

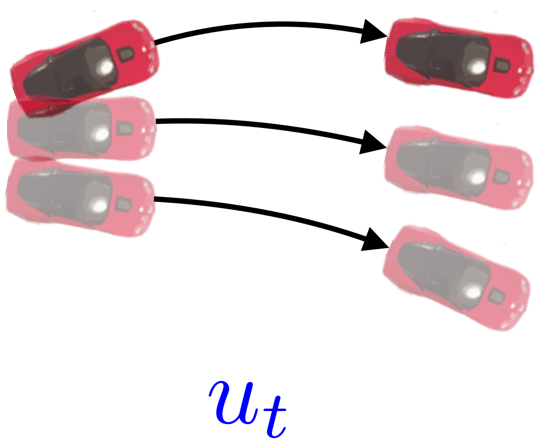
all weights = 1/M

for $i = 1$ to M

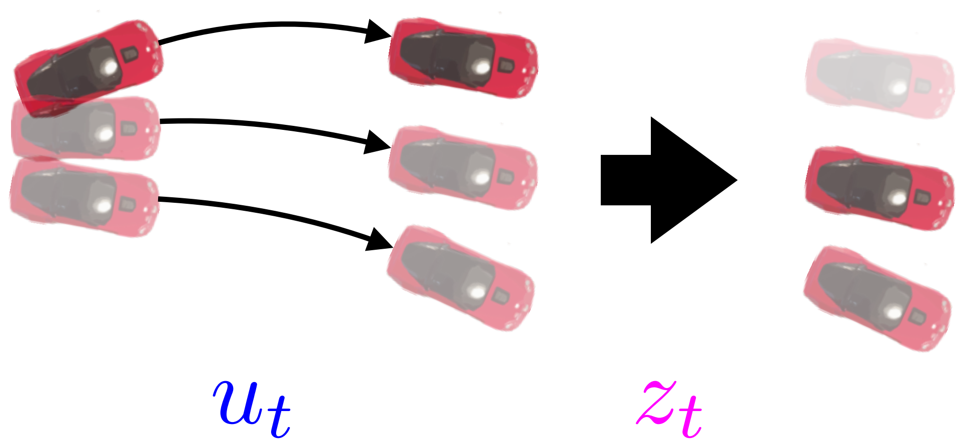
sample $x_t^i \sim w_t^i$

$$bel(x_t) = \left\{ \begin{matrix} x_t^1 \\ 1 \end{matrix}, \cdots, \begin{matrix} x_t^M \\ 1 \end{matrix} \right\}$$

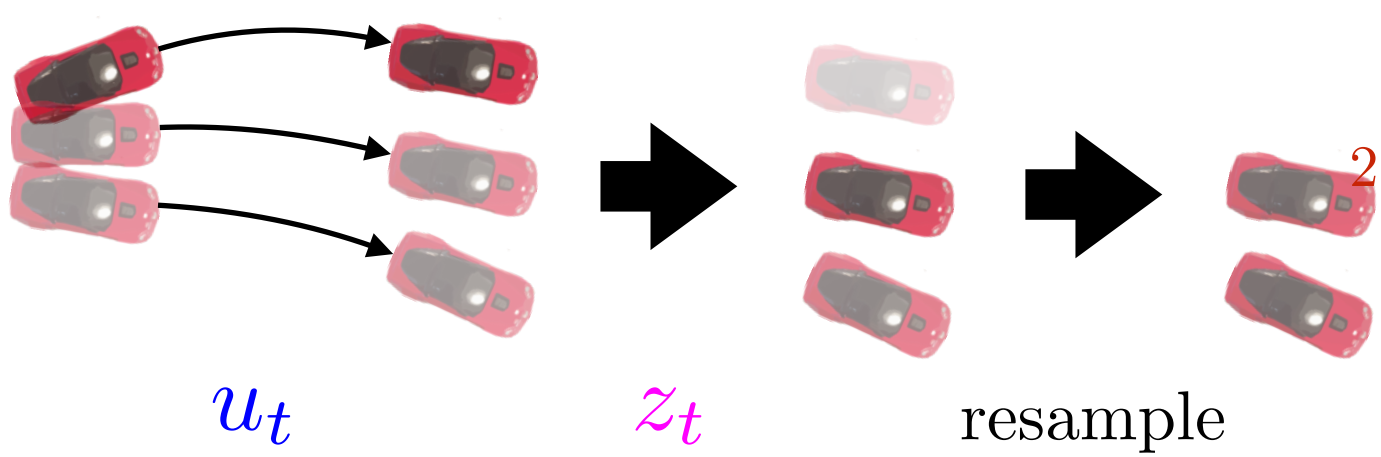
Virtues of resampling



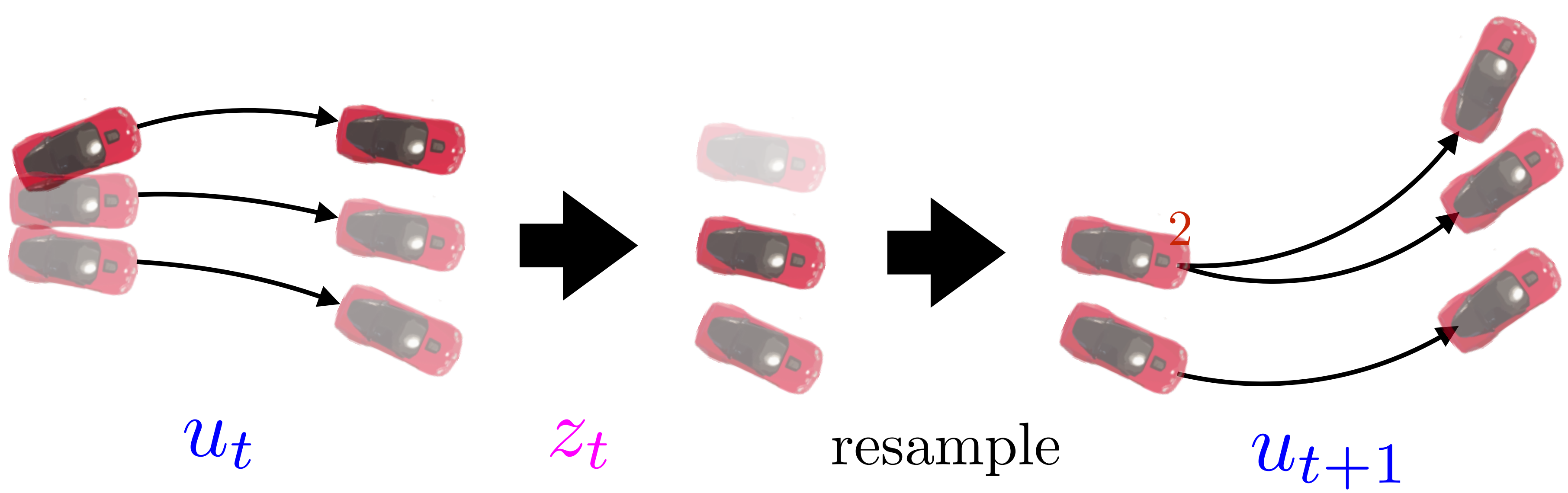
Virtues of resampling



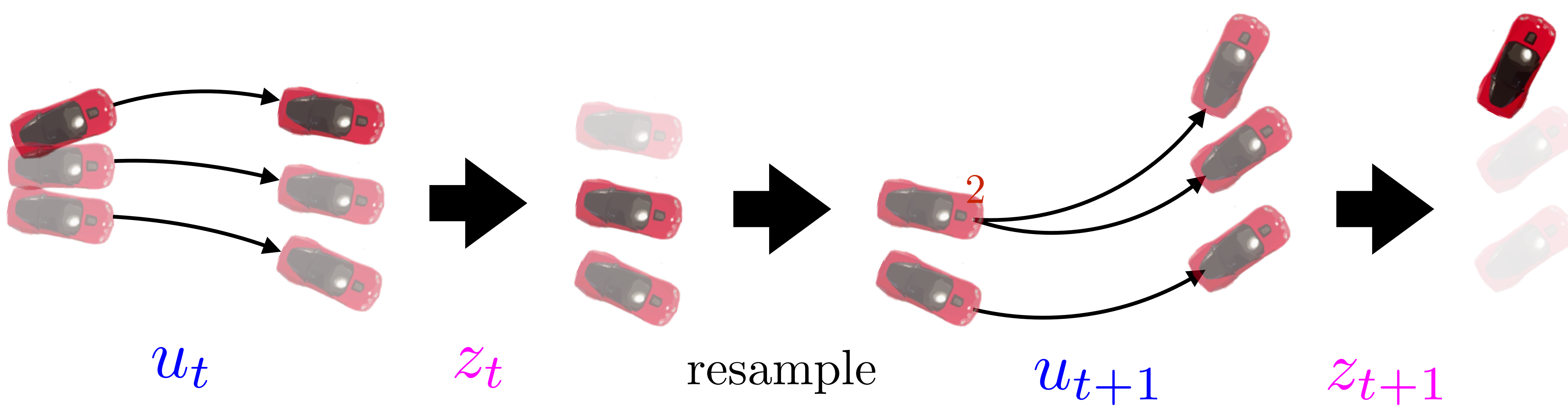
Virtues of resampling



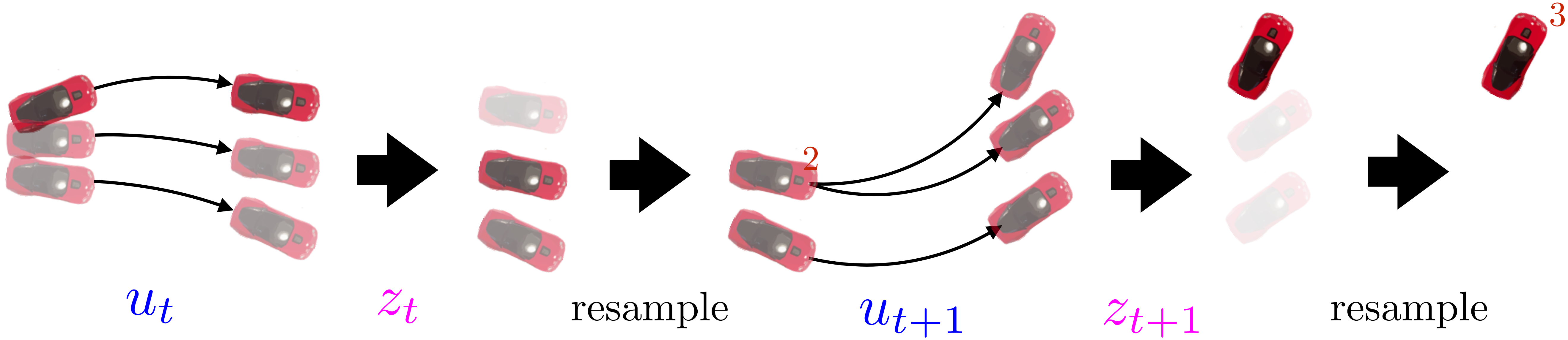
Virtues of resampling



Virtues of resampling

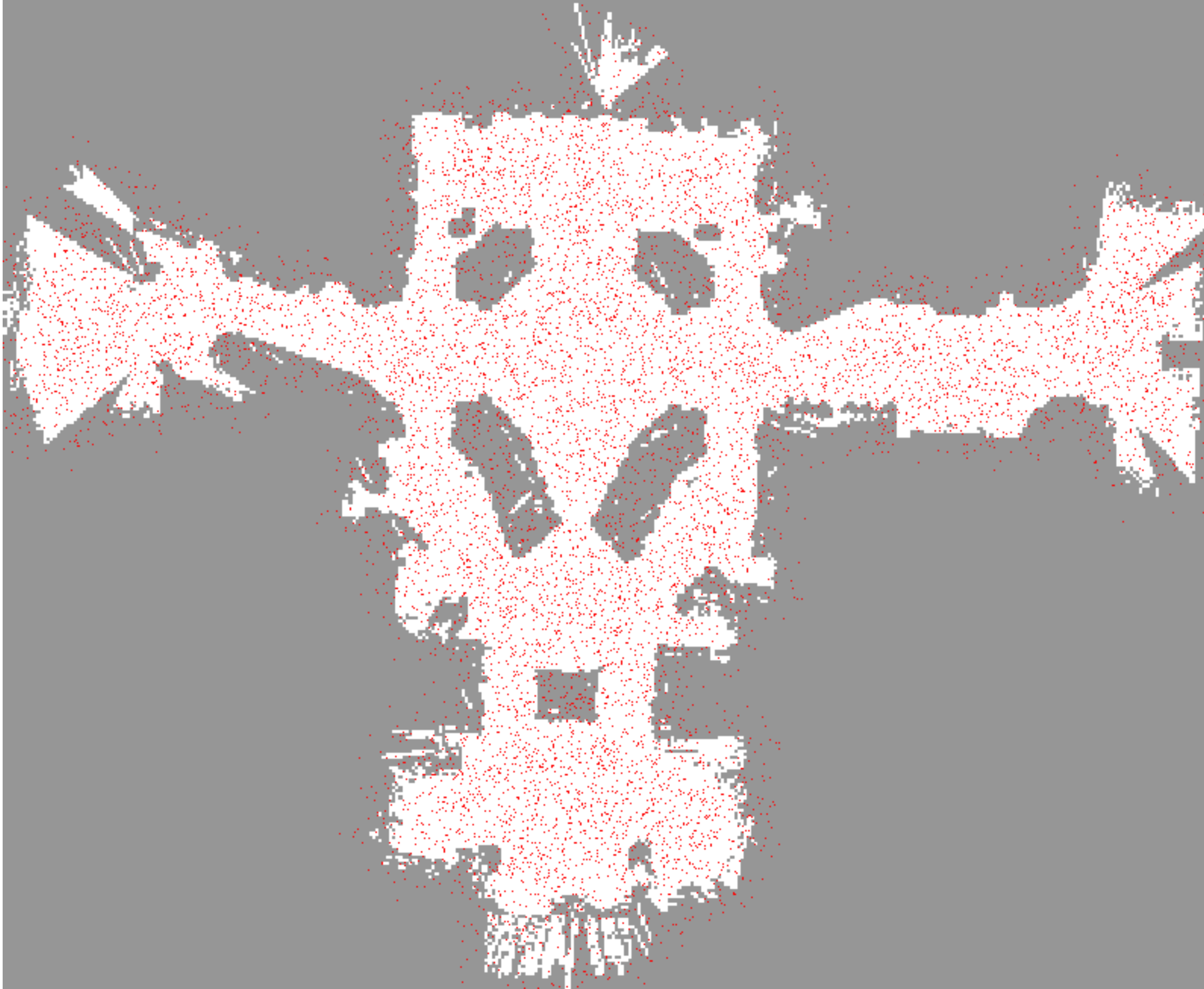


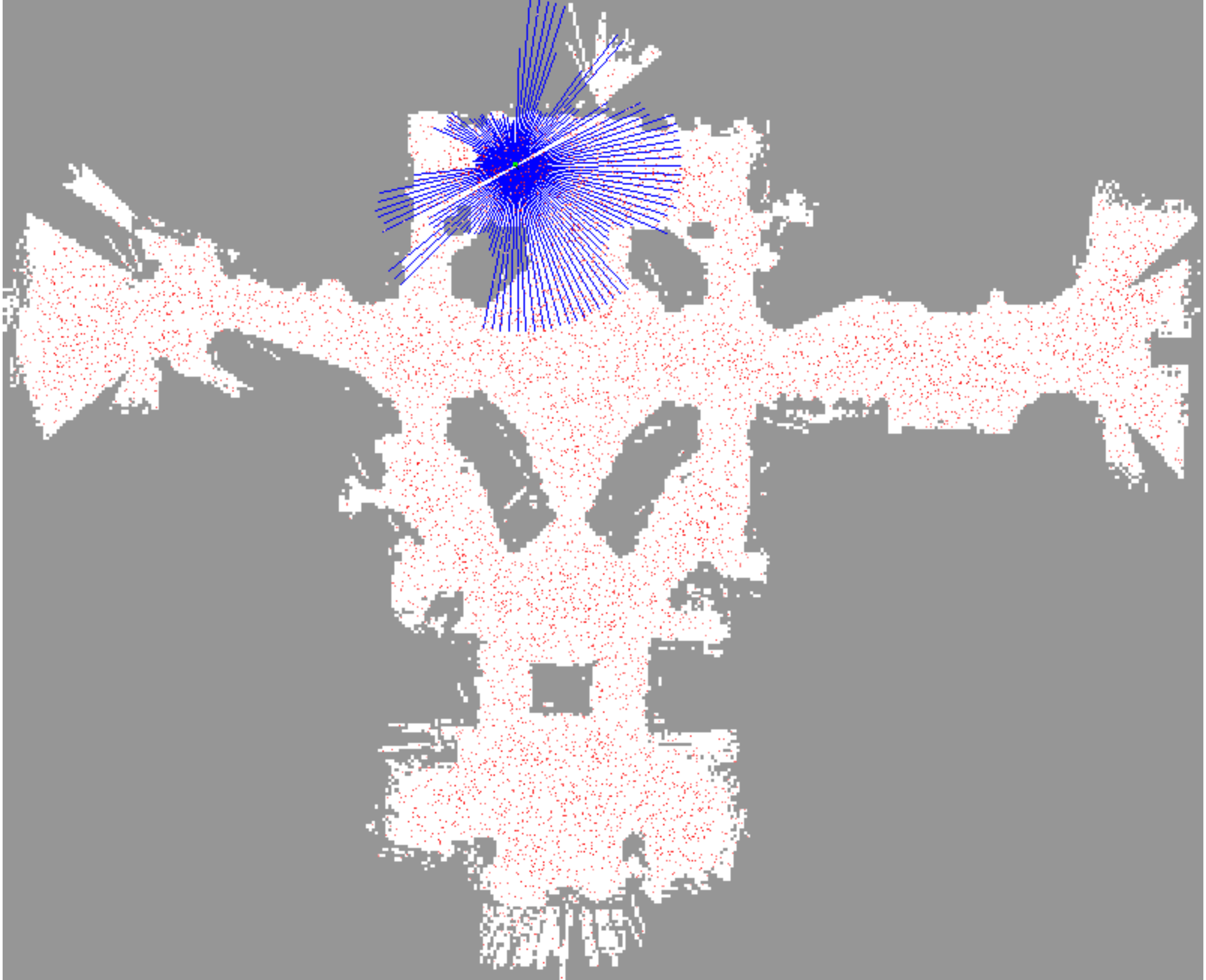
Virtues of resampling

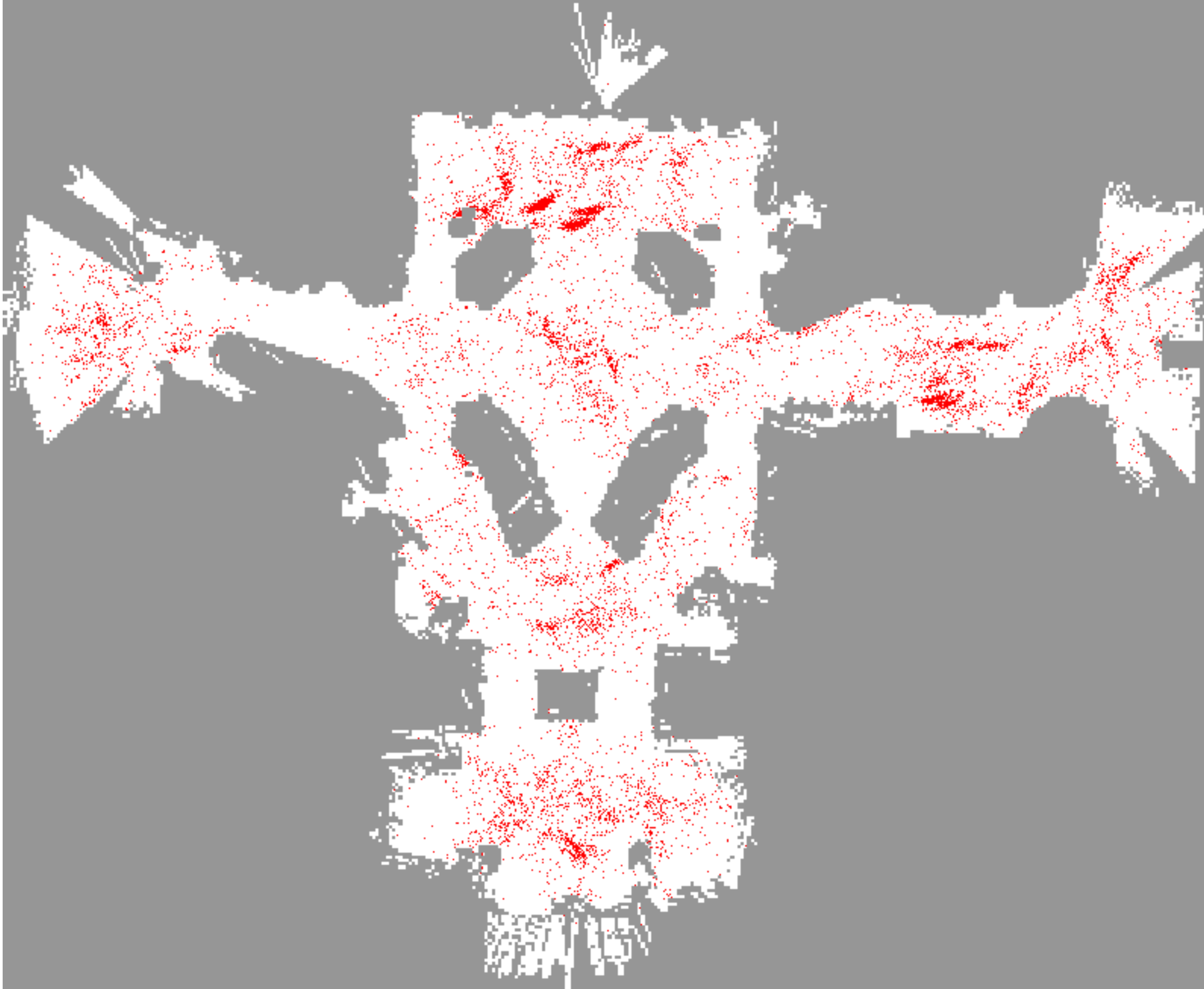


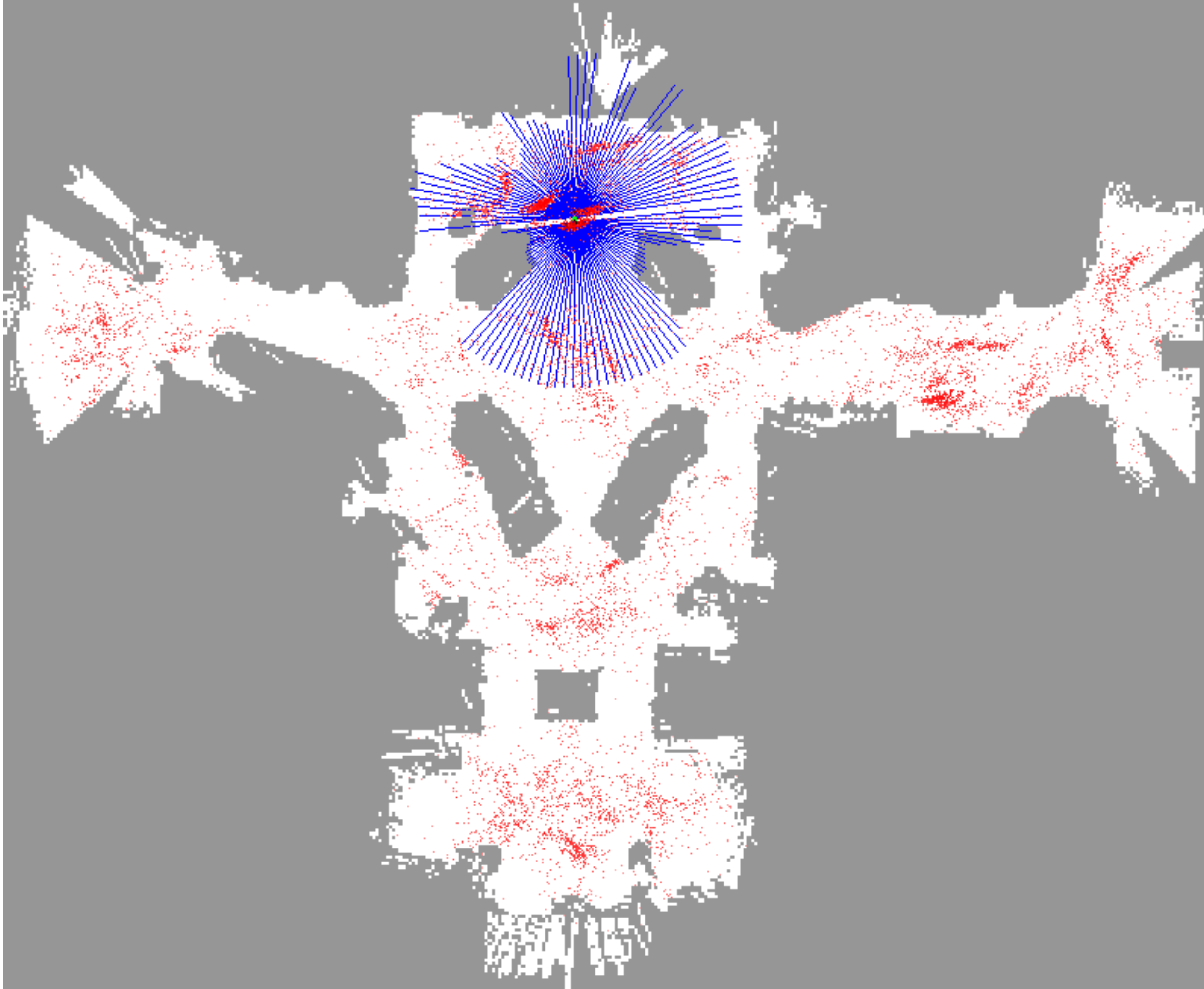
Why use particle filters?

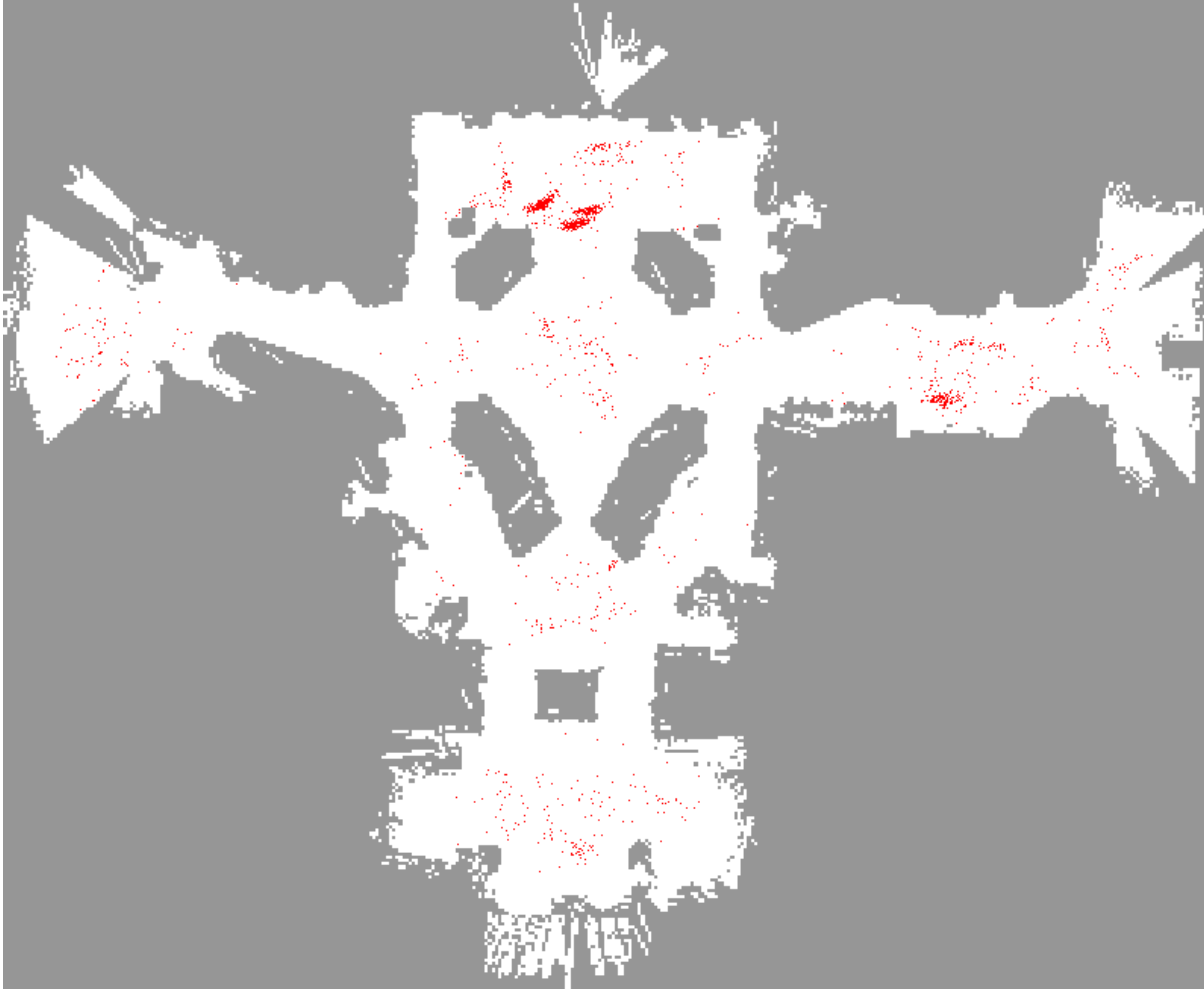
1. Can answer any query
2. Will work for any distribution, including multi-modal (unlike Kalman filter)
3. Scale well in computational resources (embarassingly parallel)
4. Easy to implement!

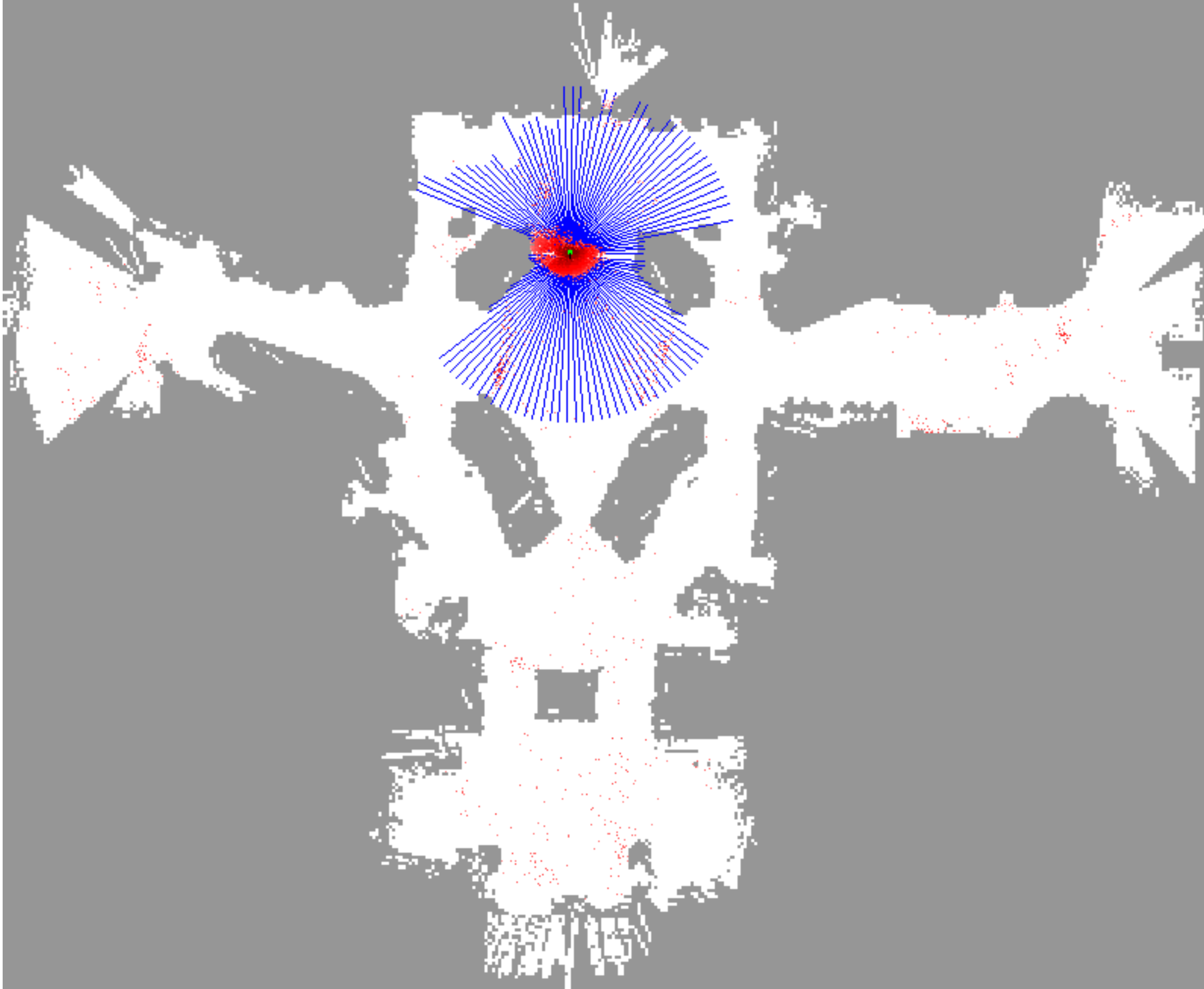


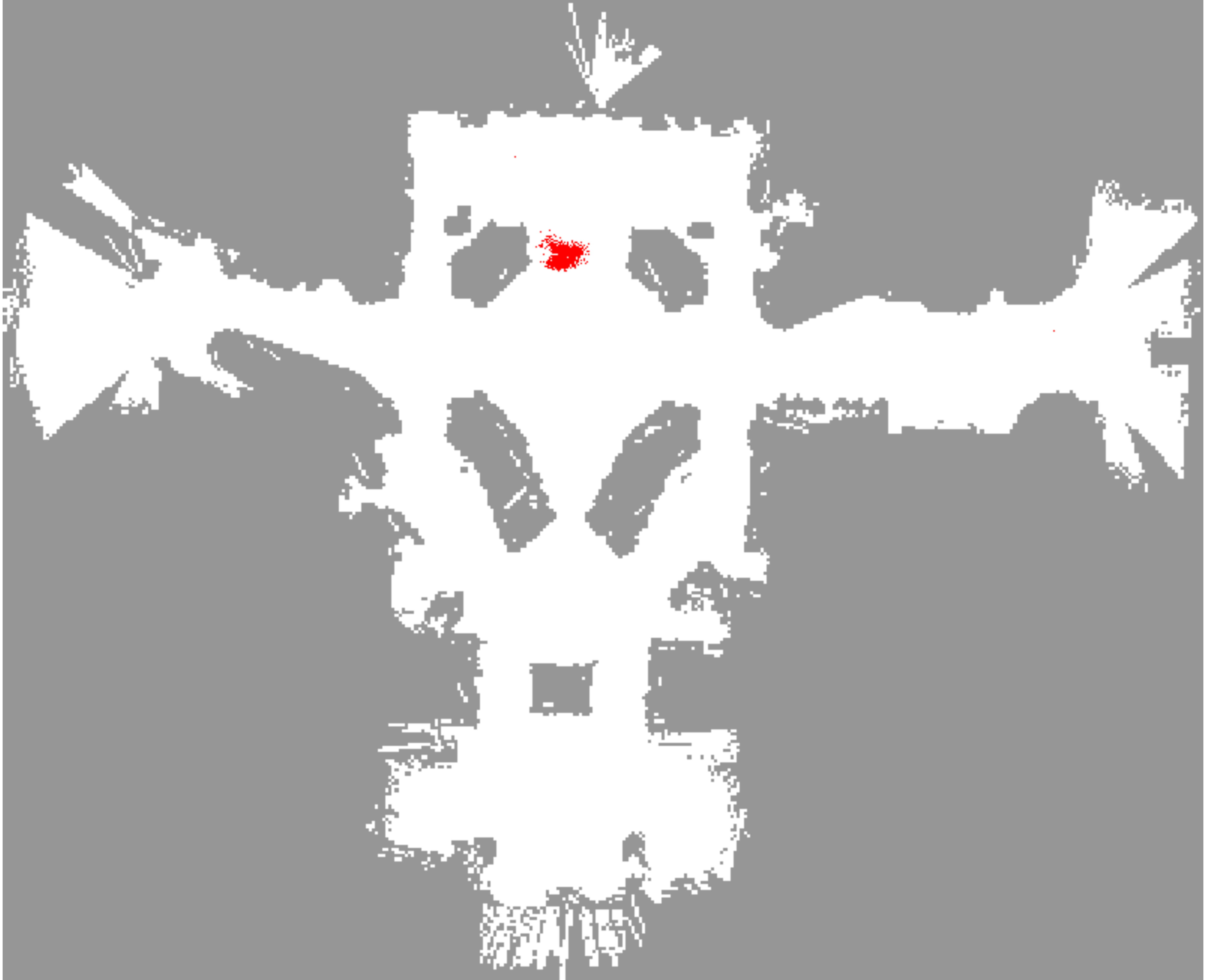












Non-parametric Filters

Grid up state space

Histogram Filter

Use a fixed set of samples

Normalized Importance Sampling

Resample

Particle Filter

Same fundamental Bayes rule again and again ...

Are we done?

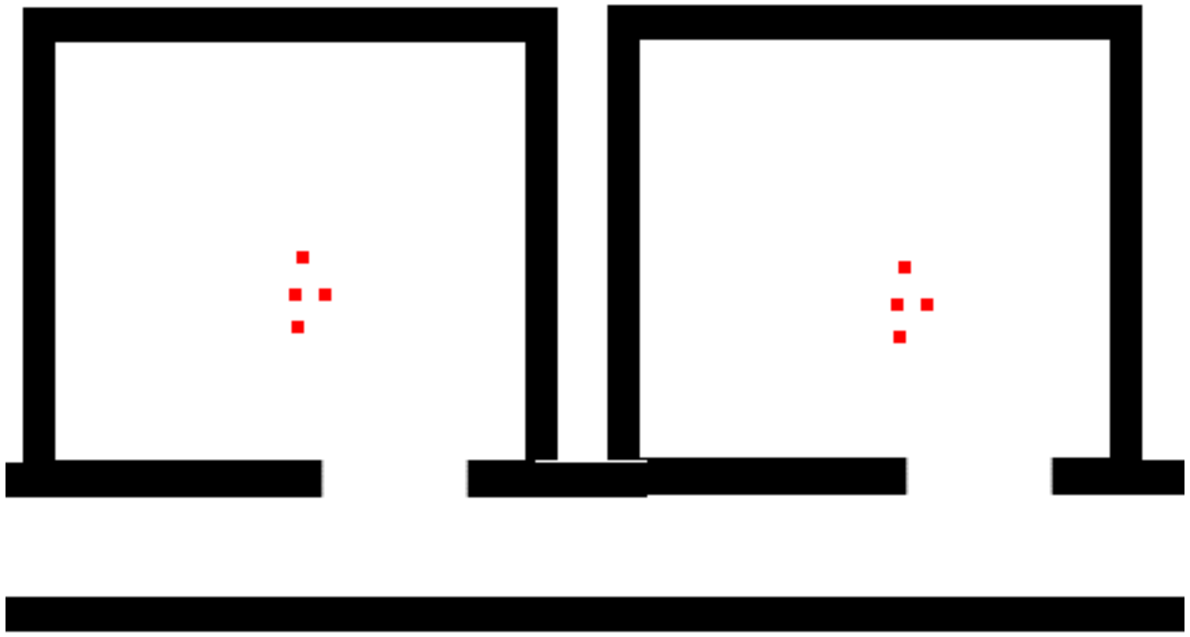
No!

Lots of practical problems to deal with

Problem 1: Two room challenge

Given: Particles equally distributed, no motion, no observation

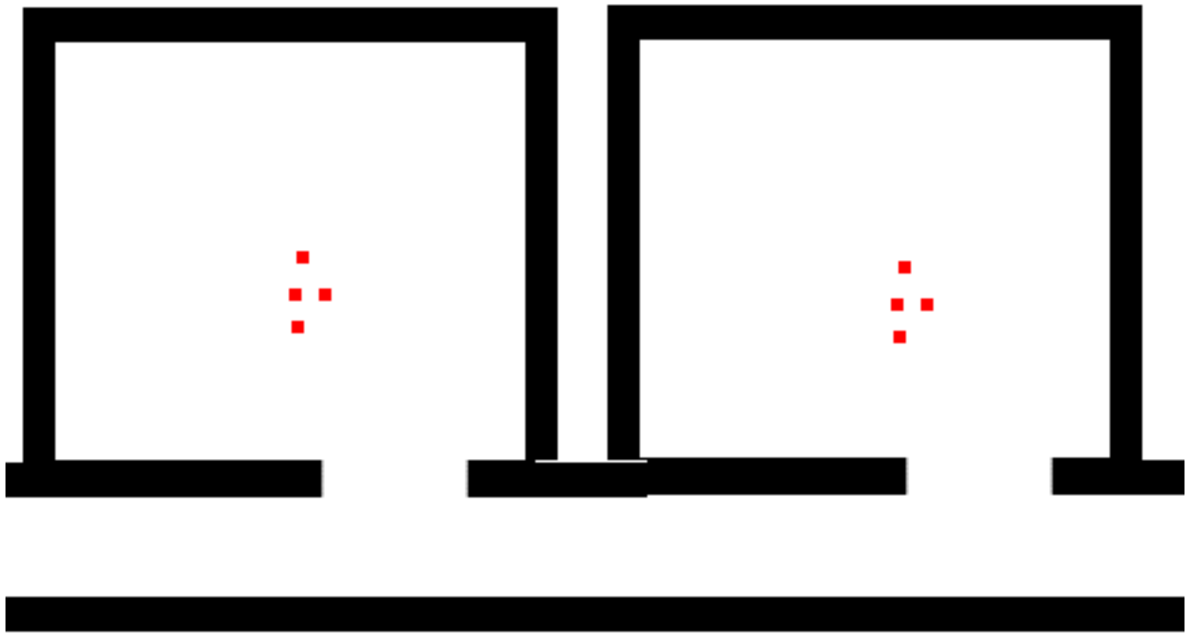
What happens?

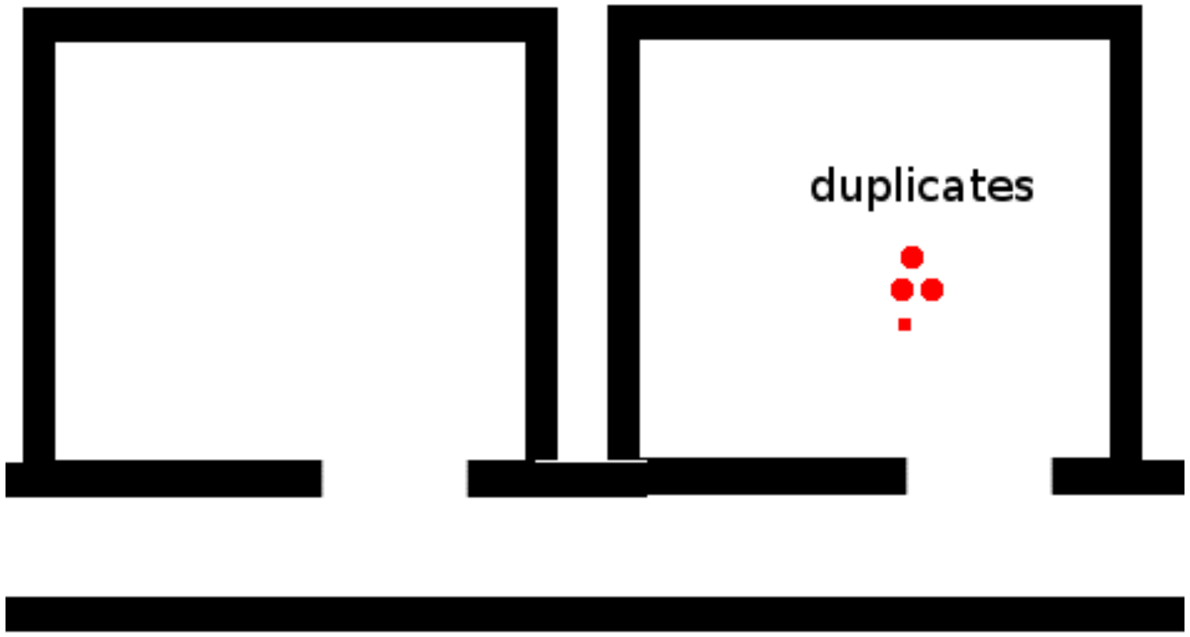


Problem 1: Two room challenge

Given: Particles equally distributed, no motion, no observation

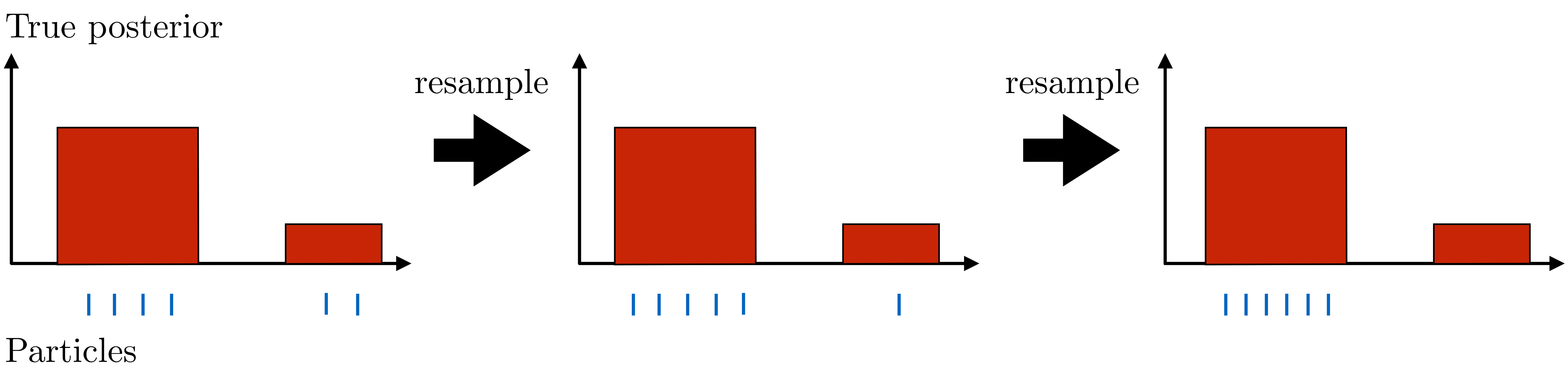
What happens?





All particles migrate to the other room!!

Reason: Resampling increases variance



Resampling collapses particles, reduces diversity, increases variance w.r.t true posterior

Fix 1: Choose when to resample

Key idea: If variance of weights low, don't resample

Fix 1: Choose when to resample

Key idea: If variance of weights low, don't resample

We can implement this condition in various ways

Fix 1: Choose when to resample

Key idea: If variance of weights low, don't resample

We can implement this condition in various ways

1. All weights are equal - don't resample

Fix 1: Choose when to resample

Key idea: If variance of weights low, don't resample

We can implement this condition in various ways

1. All weights are equal - don't resample

2. Entropy of weights high - don't resample

Fix 1: Choose when to resample

Key idea: If variance of weights low, don't resample

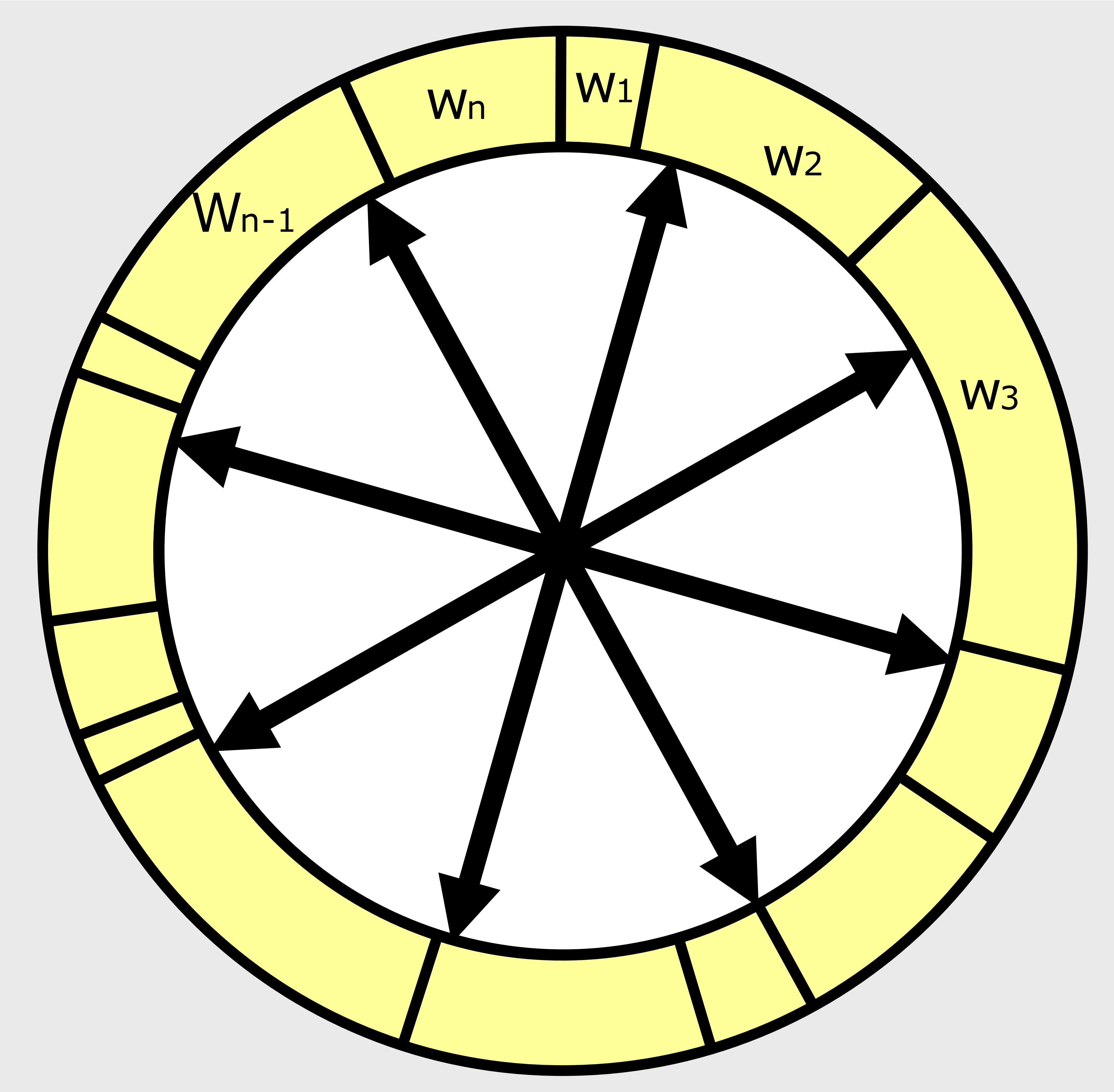
We can implement this condition in various ways

1. All weights are equal - don't resample

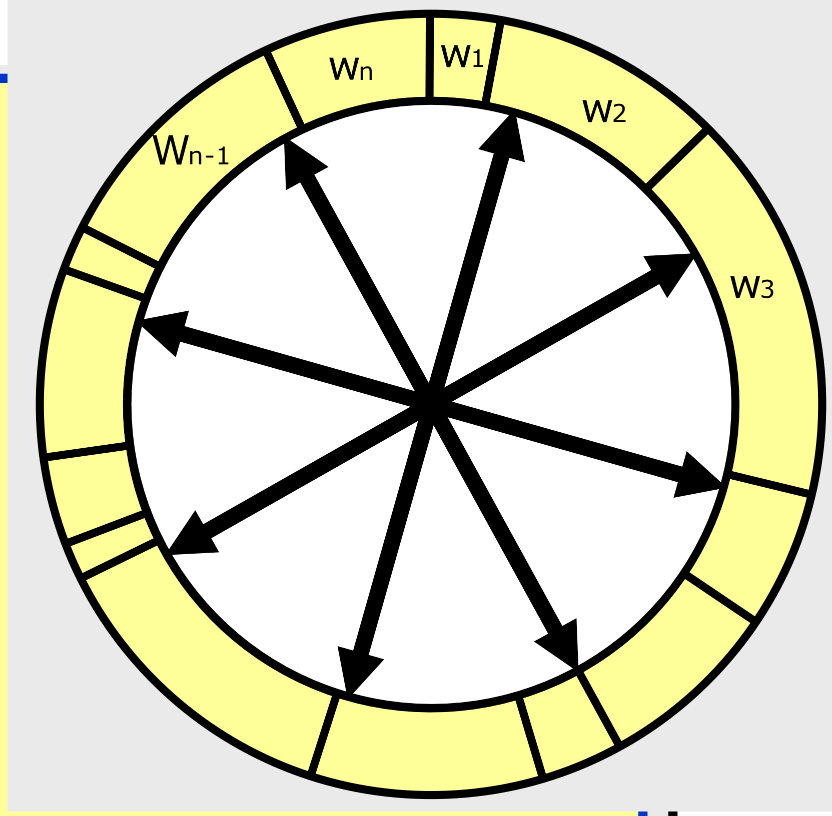
2. Entropy of weights high - don't resample

3. Ratio of max to min weights low - don't resample

Fix 2: Low variance sampling



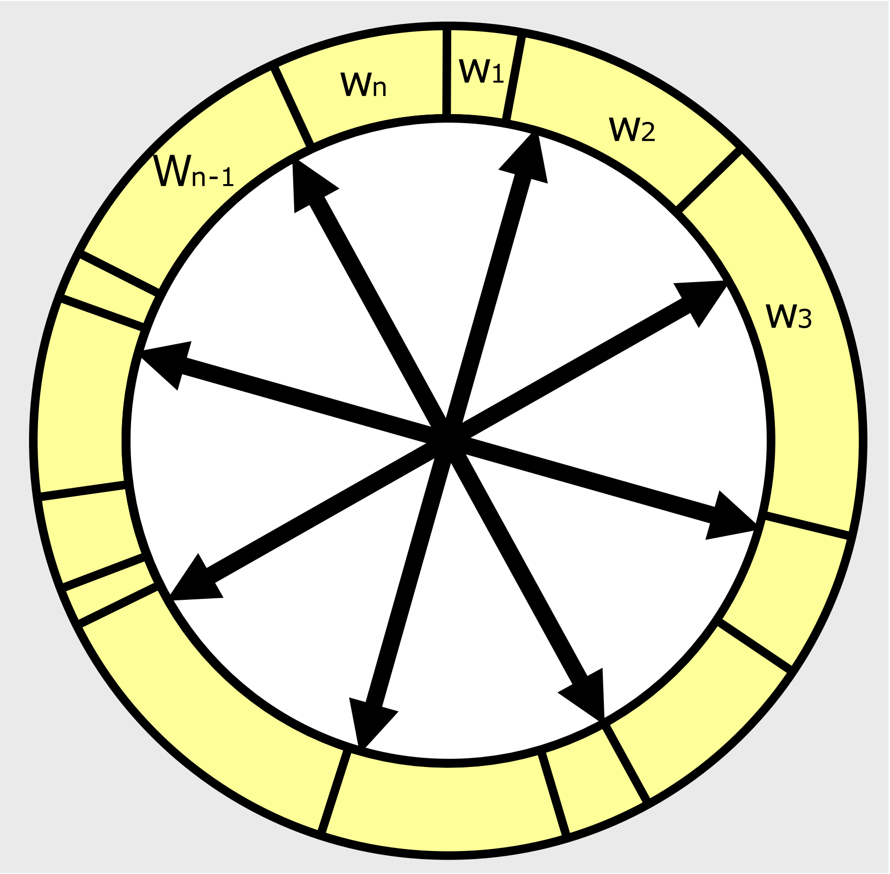
Fix 2: Low variance sampling




```
Algorithm systematic_resampling(S,n):

S' = ∅, c_1 = w^1                      Assumption: weights sum to 1
For i = 2…n                            Generate cdf
    c_i = c_{i-1} + w^i
u_1 ~ U[0, n^{-1}], i = 1              Initialize threshold

For j = 1…n                            Draw samples …
    While ( u_j > c_i )                Skip until next threshold reached
        i = i + 1
    S' = S' ∪ {< x^i, n^{-1} >}        Insert
    u_j = u_j + n^{-1}                 Increment threshold

Return S'
```

Also called **stochastic universal sampling**

Why does this work?



1. What happens when all weights equal?

2. What happens if you have ONE large weight and many tiny weights?

$$\mathrm{w1} = 0.5, \;\; \mathrm{w2} = 0.5/1000, \; \mathrm{w3} = 0.5/1000, \; .... \; \mathrm{w1001} = 0.5/1000$$

Problem 2: Particle Starvation

No particles in the vicinity of the current state

Problem 2: Particle Starvation

No particles in the vicinity of the current state

Why?

1. Unlucky set of samples

2. Committed to the wrong mode in a multi-modal scenario

3. Bad set of measurements

Fix: Add new particles

Fix: Add new particles

Which distribution should be used to add new particles?

Fix: Add new particles

Which distribution should be used to add new particles?

1. Uniform distribution

2. Biased around last good measurement

3. Directly from the sensor model

Fix: Add new particles

When should we add new samples?

Fix: Add new particles

When should we add new samples?

Key Idea: As soon as importance weights become too small, add more samples

Fix: Add new particles

When should we add new samples?

Key Idea: As soon as importance weights become too small, add more samples

1. Threshold the total sum of weights

2. Fancy estimator that checks rate of change.

Problem 3: Observation model too good!

Observation model is so peaky, that all particles die!

Problem 3: Observation model too good!

Observation model is so peaky, that all particles die!

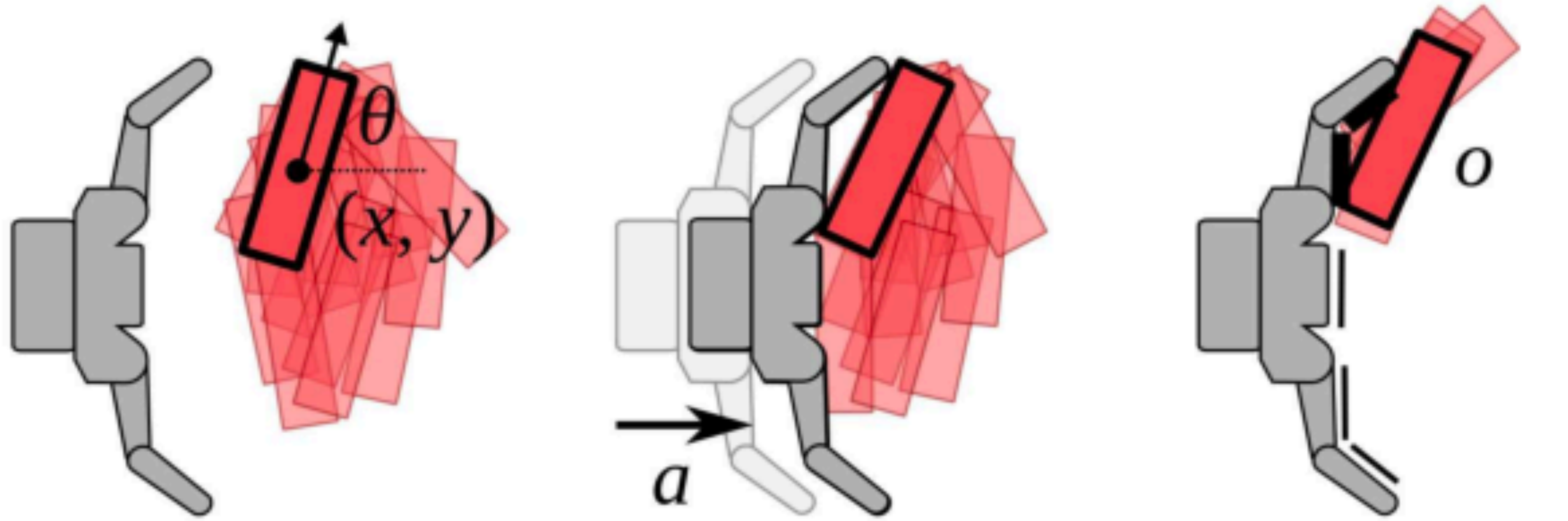
Fixes

1. Sample from a better proposal distribution than motion model!

2. Squash the observation model (apply a power of $1/m$ to all probabilities. m observations count as one)

3. Last resort: Smooth your observation model with a Gaussian (you are pretending your observation model is worse than it is)

Fix 1: Sample from a better proposal distribution



Koval et al. 2017

Contact observation may kill ALL particles!

Key Idea: Sample and weigh particles correctly

$$bel(x_t) = \eta P(z_t|x_t) \int P(x_t|x_{t-1}, u_t)\, bel(x_{t-1}) dx_{t-1}$$

(Sample) (Reweigh)

Problem 4: How many samples is enough?

Example: We typically need more particles at the beginning of run

Key idea: KLD Sampling (Fox et al. 2002)

1. Partition the state-space into bins

2. When sampling, keep track of the number of bins

3. Stop sampling when you reach a statistical threshold that depends on the number of bins

(If all samples fall in a small number of bins -> lower threshold)

KLD sampling

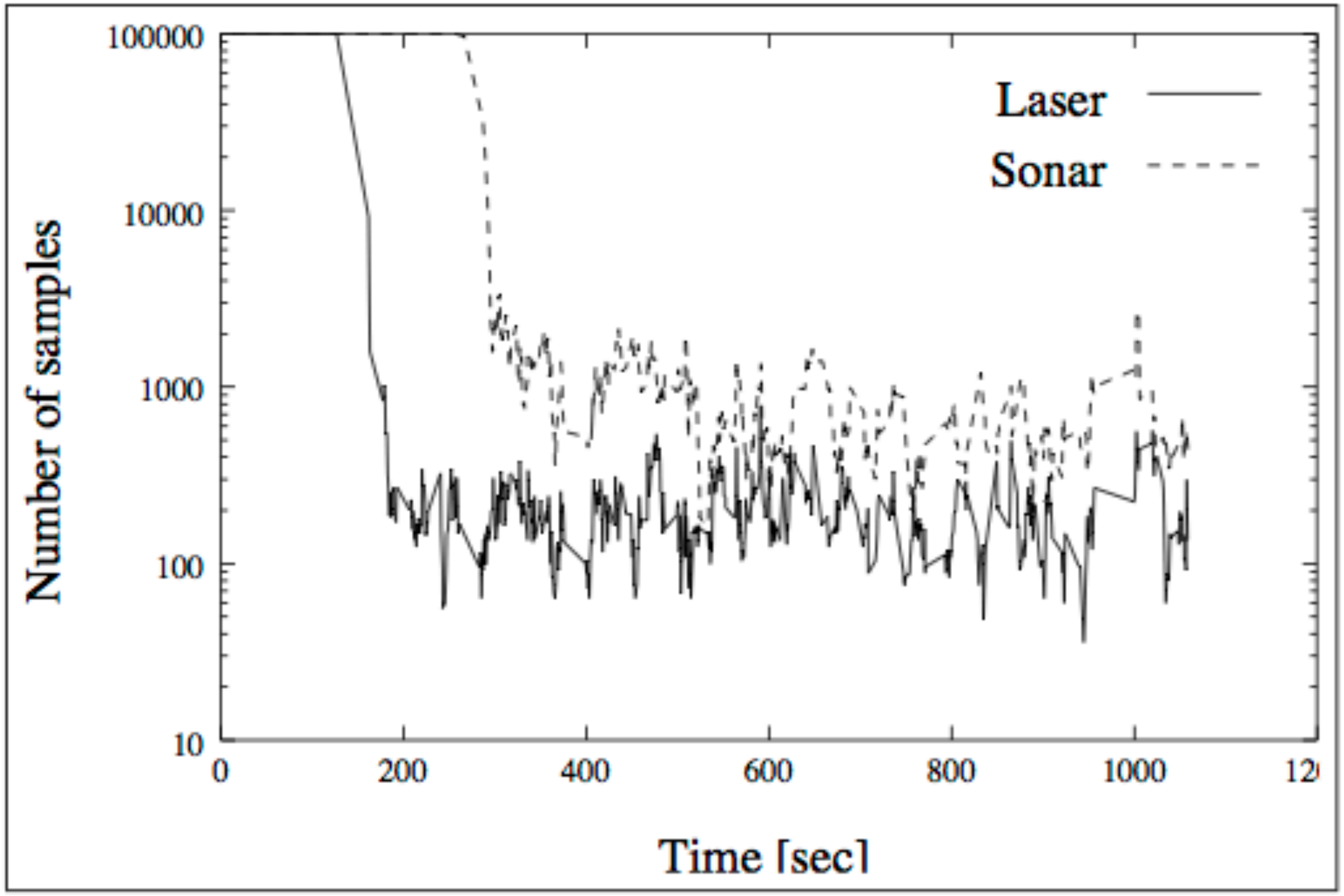


Figure 8.18 KLD-sampling: Typical evolution of number of samples for a global localization run, plotted against time (number of samples is shown on a log scale). The solid line shows the number of samples when using the robot's laser range-finder, the dashed graph is based on sonar sensor data.

Closing: Myth busting Particle filters

1. Particle Filter = Sample from motion model, weight by observation

2. Particle filters are for localization

3. Particle filters are to do with samples

Closing: Myth busting Particle filters

1. Particle Filter = Sample from motion model, weight by observation

(sample from any good proposal distribution)

2. Particle filters are for localization

3. Particle filters are to do with samples

Closing: Myth busting Particle filters

1. Particle Filter = Sample from motion model, weight by observation

BUSTED

(sample from any good proposal distribution)

2. Particle filters are for localization

BUSTED

(any continuous space estimation problem)

3. Particle filters are to do with samples

Closing: Myth busting Particle filters

1. Particle Filter = Sample from motion model, weight by observation

BUSTED

(sample from any good proposal distribution)

2. Particle filters are for localization

BUSTED

(any continuous space estimation problem)

3. Particle filters are to do with samples

BUSTED

(normalized importance sampling also uses samples but no resampling)