

Robotica Probabilistica

Filtri Bayesiani

Filtri discreti e Particle filters

Filtri ad Istogrammi e Filtri Discreti

- Decompongono lo spazio in regioni e rappresentano il post cumulativo di ogni regione con un valore di probabilità
- Se applicati a spazio finito sono detti Filtri Discreti: X_t prende valori in un insieme finito (es. occupancy grid X_{ij} ha 2 valori)

Filtro Discreto: Algoritmo

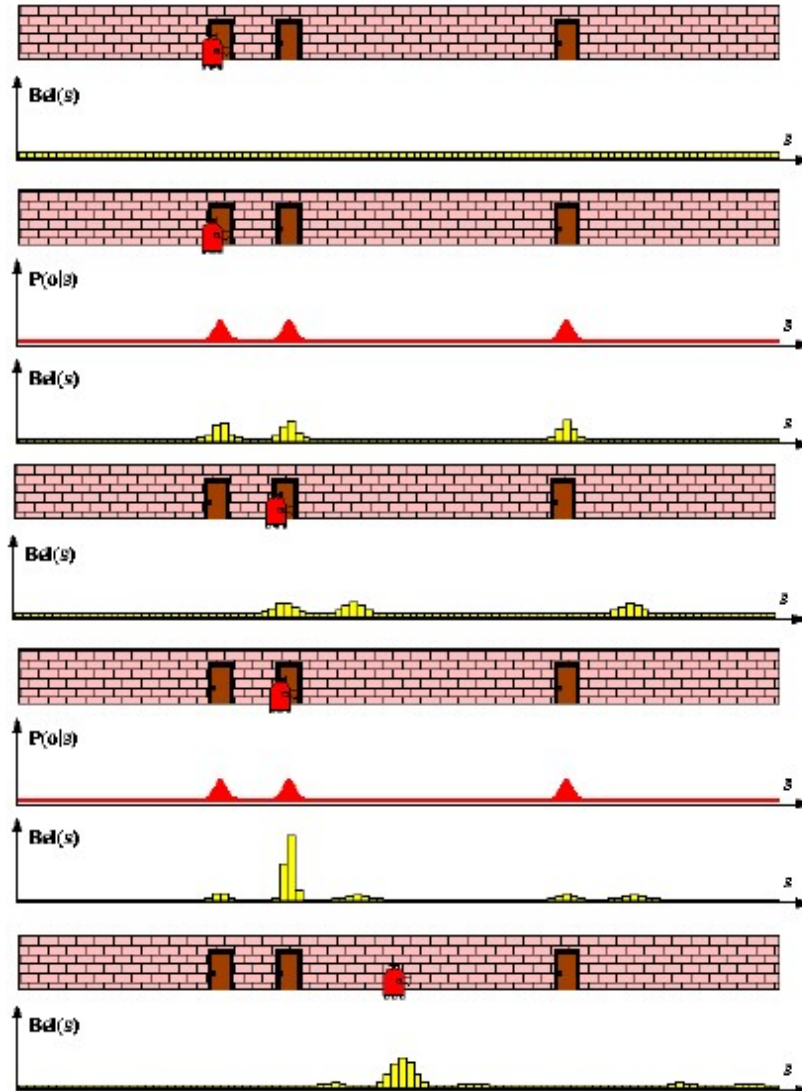
1. Algorithm **Discrete_Bayes_filter**($Bel(x), d$):
2. $\eta = 0$
3. If d is a **perceptual** data item z then
4. For all x do
5. $Bel'(x) = P(z | x)Bel(x)$
6. $\eta = \eta + Bel'(x)$
7. For all x do
8. $Bel'(x) = \eta^{-1}Bel'(x)$
9. Else if d is an **action** data item u then
10. For all x do
11. $Bel'(x) = \sum_{x'} P(x | u, x') Bel(x')$
12. Return $Bel'(x)$

Filtro ad Istogramma

- Se spazio continuo Filtri ad Istogramma (X_t prende valori in spazio continuo)
- Spazio diviso in regioni finite e convesse (bins) che partizionano lo stato: $X_t = x_{1,t} \vee \cdots \vee x_{n,t}$
- Per ogni regione è associata la probabilità $p_{k,t}$ quindi
$$p(x_t) = \frac{p_{k,t}}{|x_{k,t}|}$$

Filtro ad Istogramma

Costante a tratti

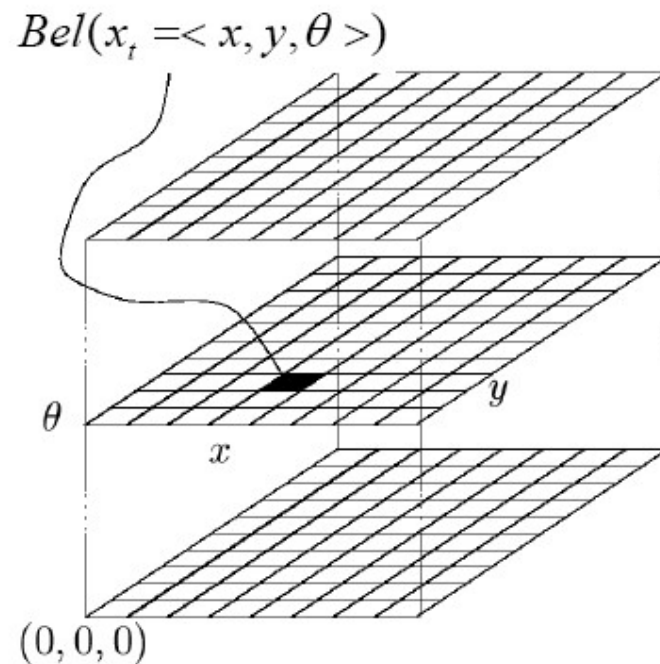


Implementazione

Statica: partizione statica dello spazio

Dinamica: partizione dello spazio
dipendente dal contesto

Rappresentazione
costante:
occupancy grid



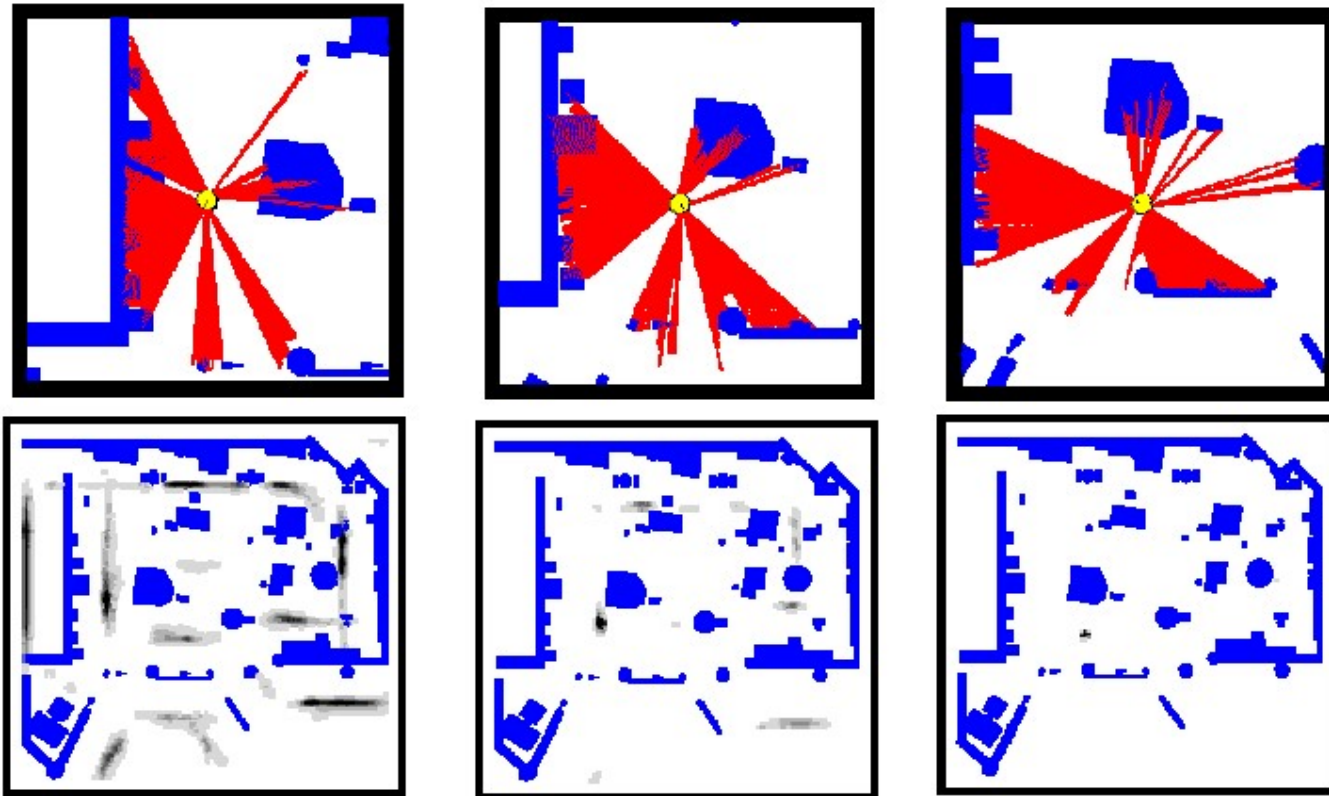
Rappresentazione Statica

Utili quando lo stato è binario

Problemi:

- Per update e normalizzazione occorre lo scan di tutta la griglia
- Se le credenze sono concentrate si vorrebbe evitare
- Si può fare l'update della sottogriglia "attiva", ma non gestisce la delocalizzazione

Localizzazione Grid-based

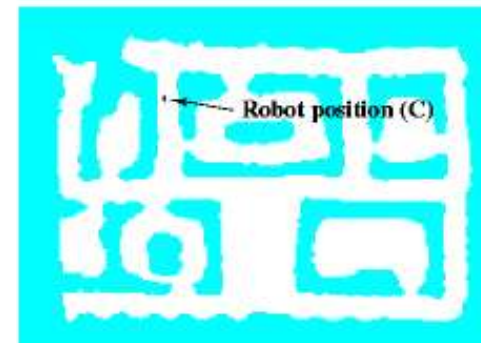
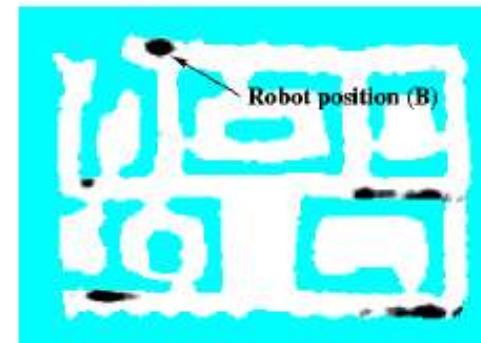
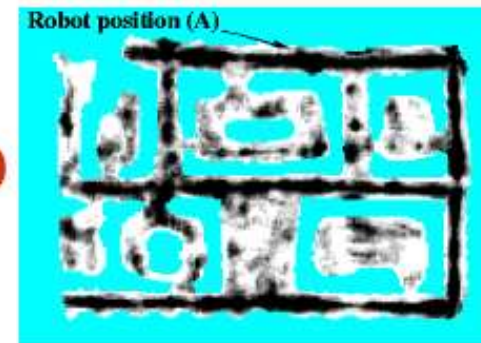
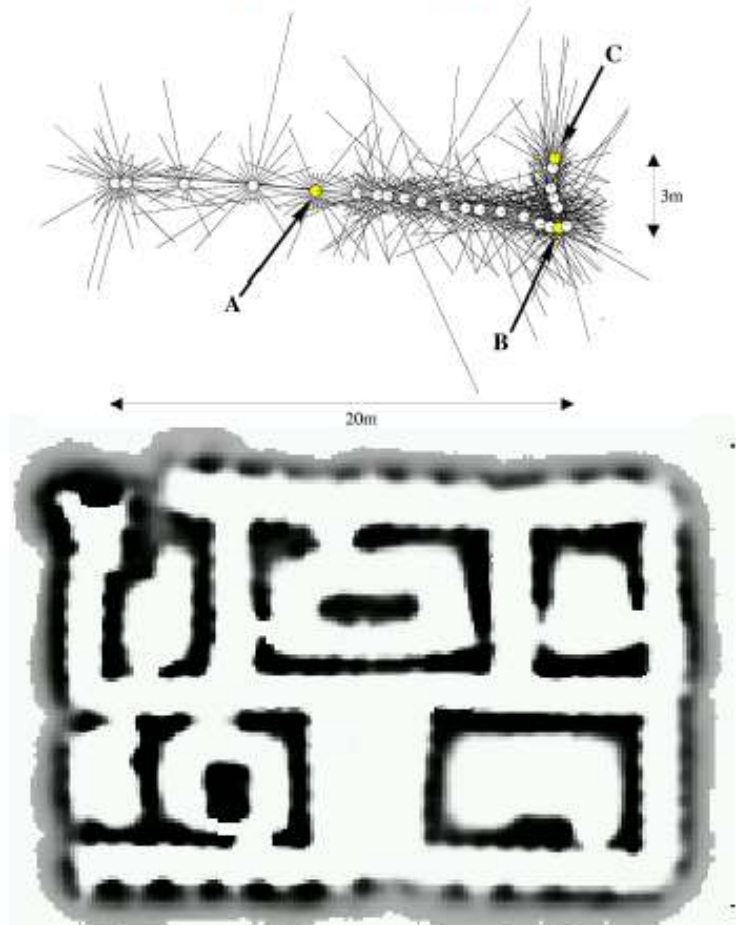


Gridmap risoluzione 15 cm, 15 g, 2LRF, beam model

Posizione dopo 3 scan

Localizzazione Grid-based

Sonars and Occupancy Grid Map



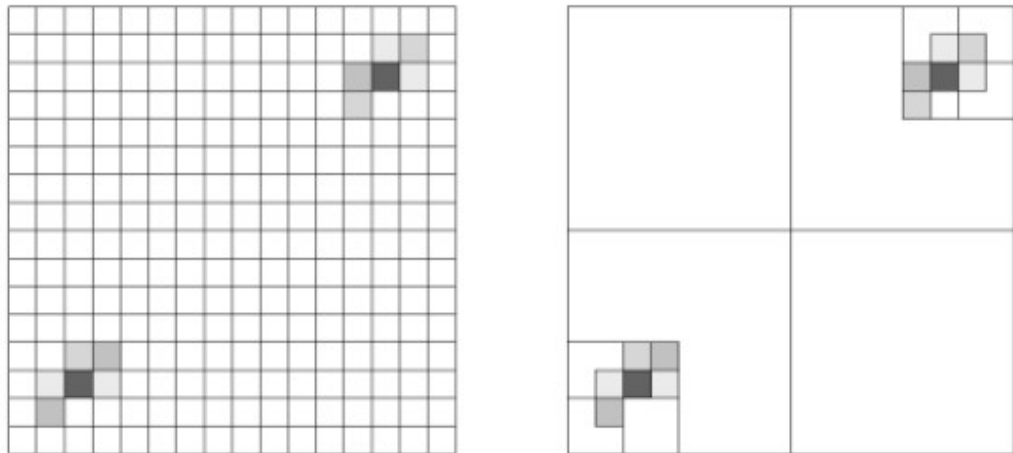
Decomposizione Dinamica

Density Tree: decomposizione ricorsiva adatta alla risoluzione del post

Meno è probabile una regione, minore è la risoluzione

Più precisa ed efficiente (ordini di grandezza)

Quadtree
o Octree



Particle Filters

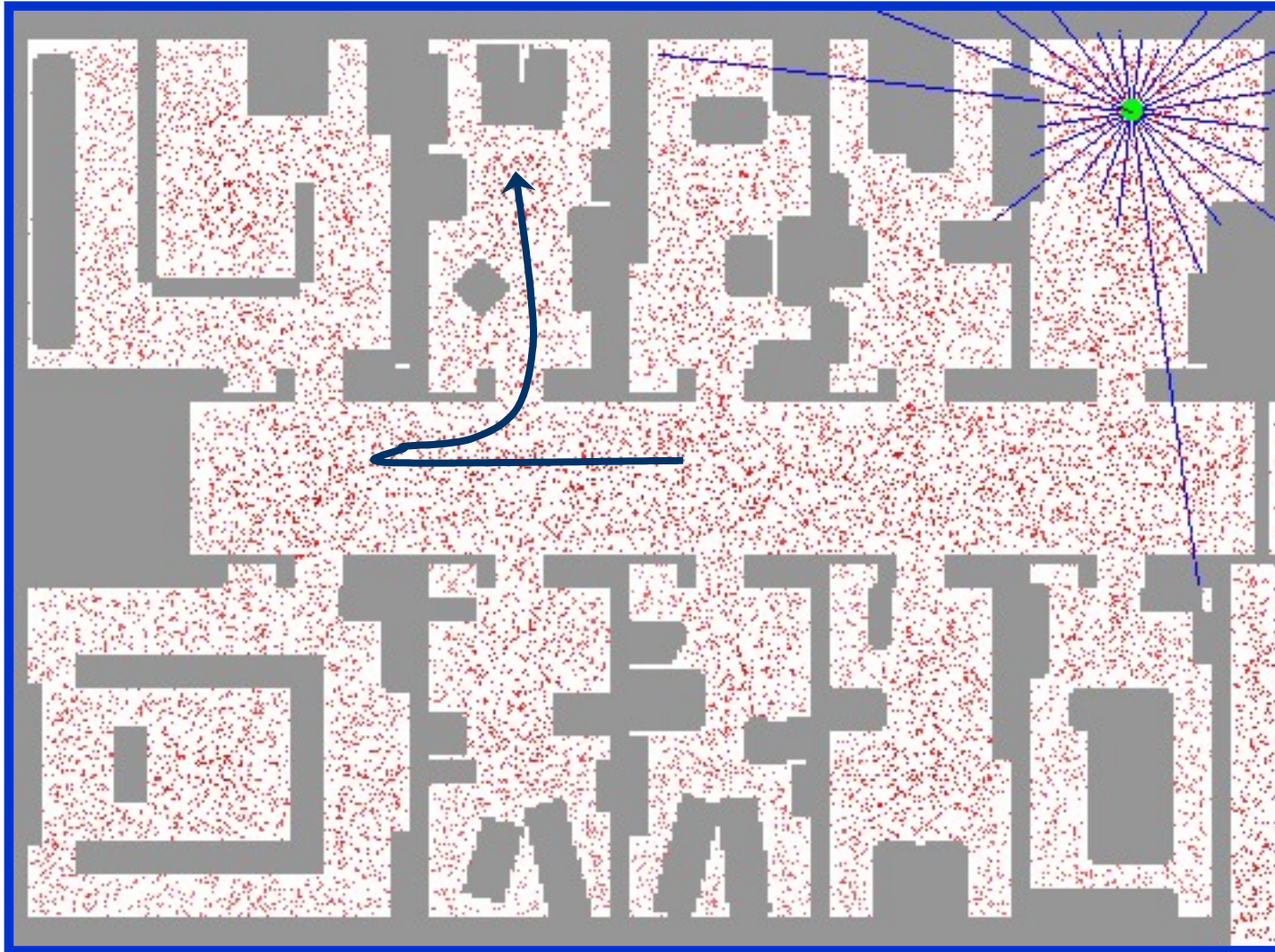
- Filtri ad Istogrammi:
 - Discretizzazione dello spazio (statica o dinamica)
- Problemi di risoluzione ed efficienza

Alternativa più efficiente: **Particle Filters**

Approssimano il post con un numero finito di parametri, ma tecnica diversa:

- Insiemi di ipotesi (particles) rappresentano il post
- Sopravvivono le migliori

Localizzazione basata su Campioni (sonar)



Particle Filters

- Rappresenta i belief con campioni random
- Stima di **non-Gaussiane, processi nonlineari**
- Monte Carlo filter, Survival of the fittest, Condensation, Bootstrap filter, Particle filter
- Adattivi: numero di campioni dipendenti dalle risorse e dalla complessità del task.
- Filtering: [Rubin, 88], [Gordon et al., 93], [Kitagawa 96]
- Computer vision: [Isard and Blake 96, 98]
- Dynamic Bayesian Networks: [Kanazawa et al., 95]d

Particle Filter

- Lo stato al tempo t è rappresentato da un insieme di campioni (particles):

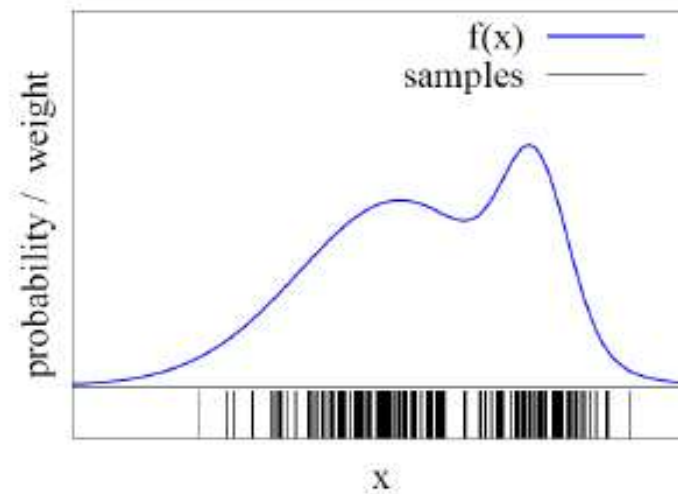
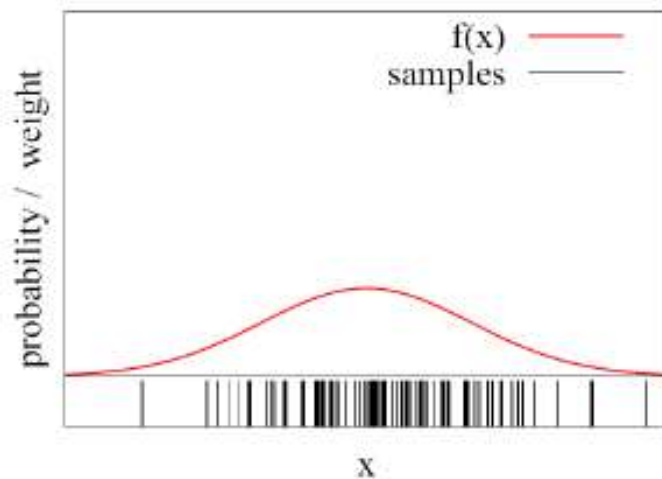
$$X_t = x_{t,1}, x_{t,2}, \dots, x_{t,M}$$

- Ogni $x_{t,i}$ è una istanza dello stato al tempo t , M è il numero delle particles
- Ogni particle è una ipotesi sullo stato
- Prob di " $x_{t,m}$ appartenente ad X_t " approssima $\text{bel}(x_{t,m}) : x_{t,m} \sim p(x_t | z_{1:t}, u_{1:t})$

Particle Filter

Gli insiemi di particles X_t approssimano una funzione di distribuzione

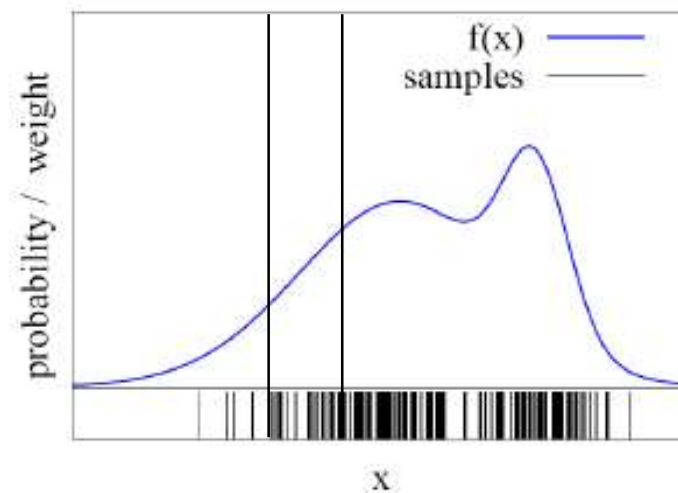
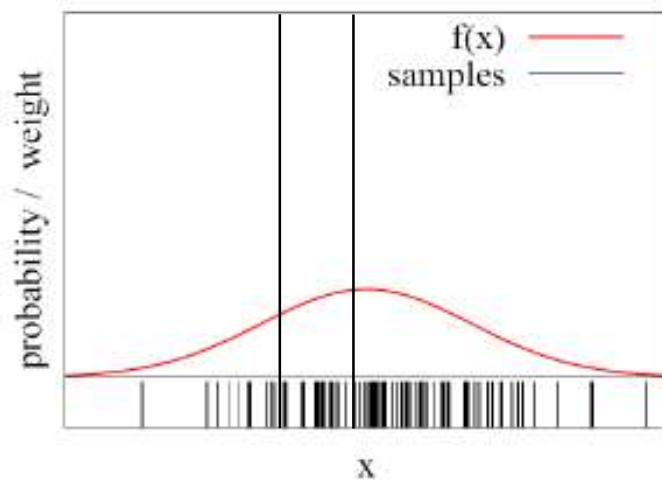
Prob di " $x_{t,m}$ appartenente ad X_t " approssima
 $\text{bel}(x_{t,m}) : x_{t,m} \sim p(x_t | z_{1:t}, u_{1:t})$



Particle Filter

Gli insiemi di particles X_t approssimano una funzione di distribuzione

Prob di " $x_{t,m}$ appartenente ad X_t " approssima
 $\text{bel}(x_{t,m}) : x_{t,m} \sim p(x_t | z_{1:t}, u_{1:t})$



Più particle cadono in un intervallo più è probabile l'intervallo

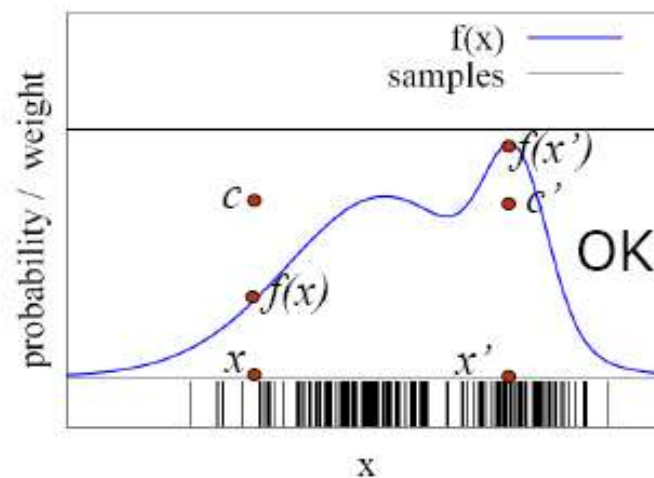
Campionamento per Rifiuto (rejection sampling)

Si assume $f(x) < 1$ per ogni x

Campionamento da distribuzione uniforme

Si campiona c da $[0,1]$ $c \sim U(0,1)$

Se $f(x) > c$ allora si mantiene il campione
altrimenti si scarta



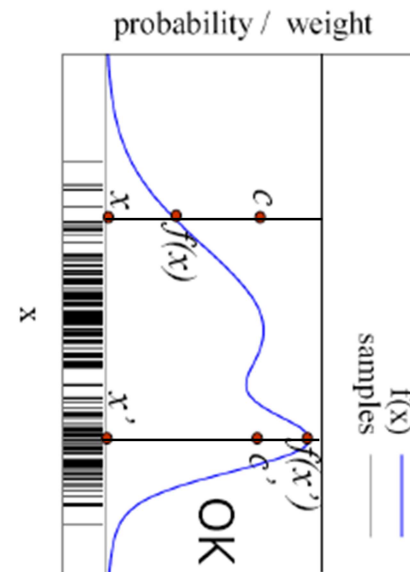
Campionamento per Rifiuto (rejection sampling)

Si assume $f(x) < 1$ per ogni x

Campionamento da distribuzione uniforme

Si campiona c da $[0,1]$ $c \sim U(0,1)$

Se $f(x) > c$ allora si mantiene il campione
altrimenti si scarta

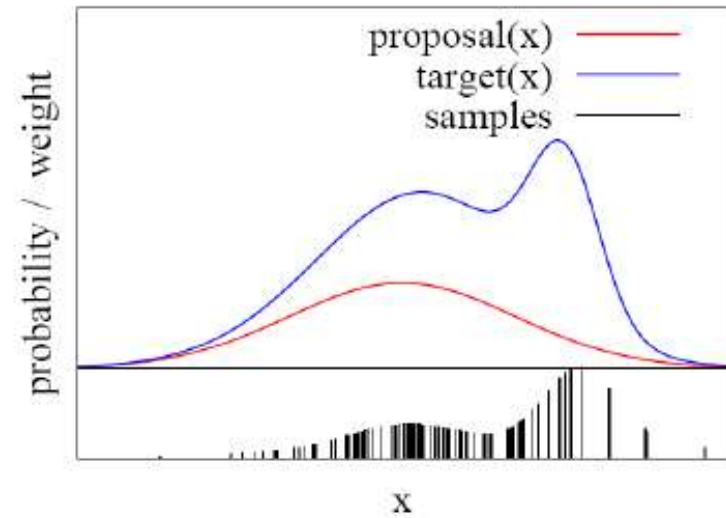


Importance Sampling

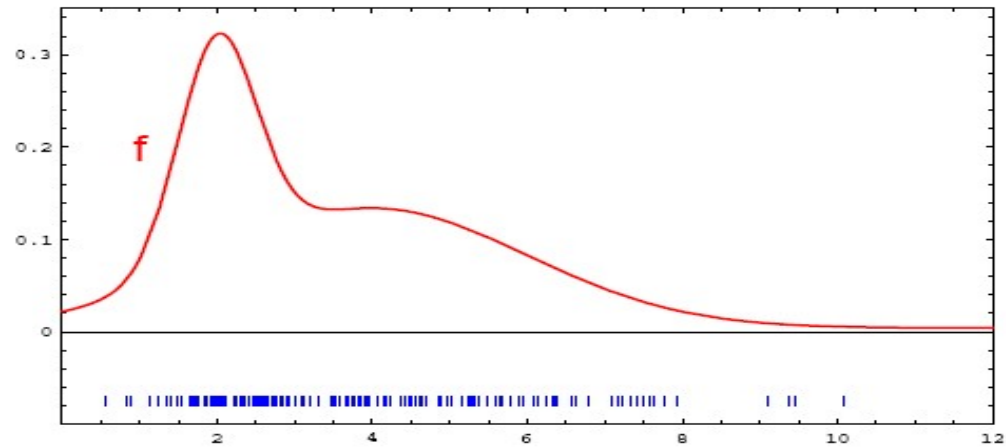
Si vuole fare sampling dalla $\text{target}(x)$ ma si può fare da $\text{proposal}(x)$

Un sample da $\text{proposal}(x)$ viene pesato con:
 $w = \text{target}(x) / \text{proposal}(x)$

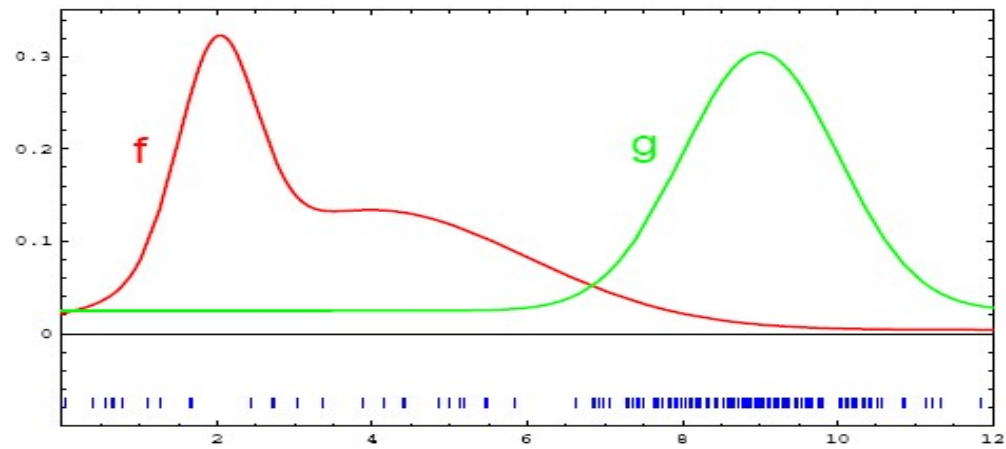
Condizione:
 $\text{target}(x) \rightarrow \text{proposal}(x)$



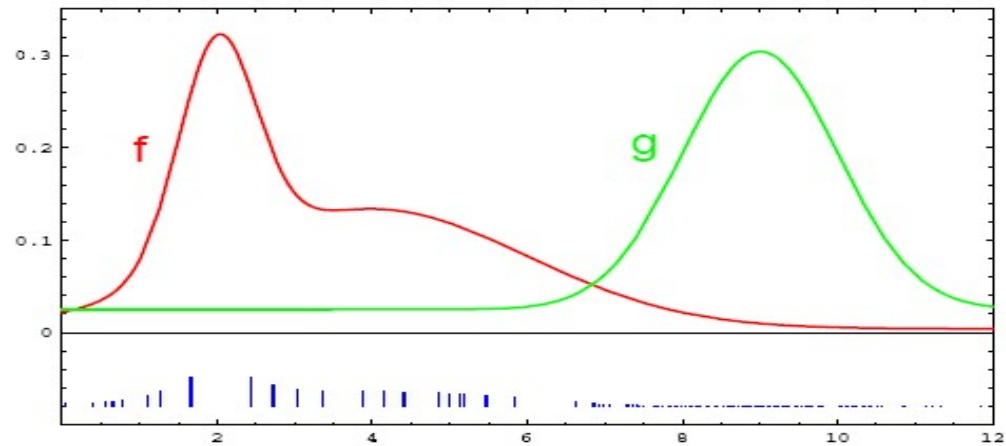
Campioni da $f(x)$,
ma non possibile



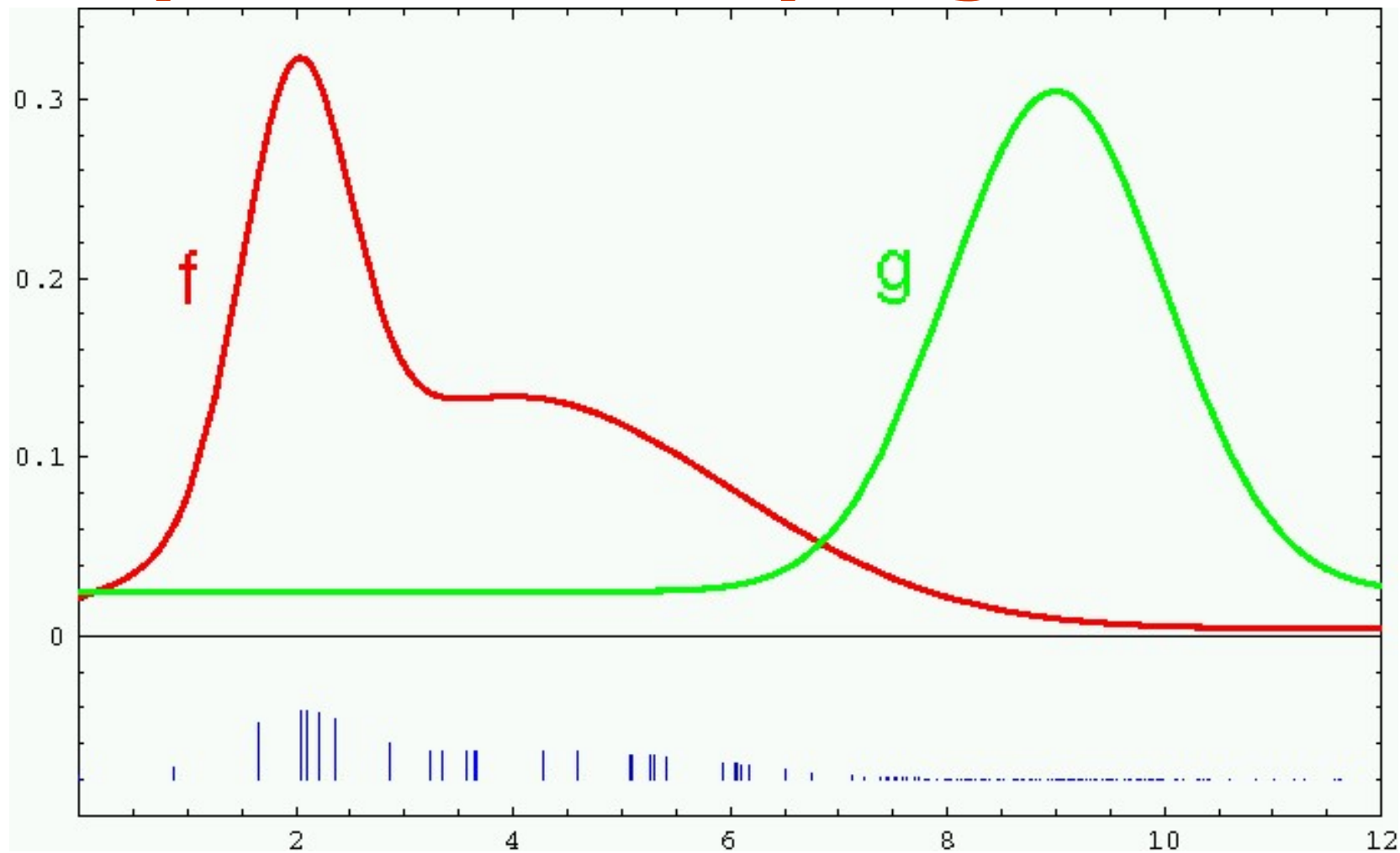
Campioni da $g(x)$



Campioni pesati
con $f(x)/g(x)$



Importance Sampling

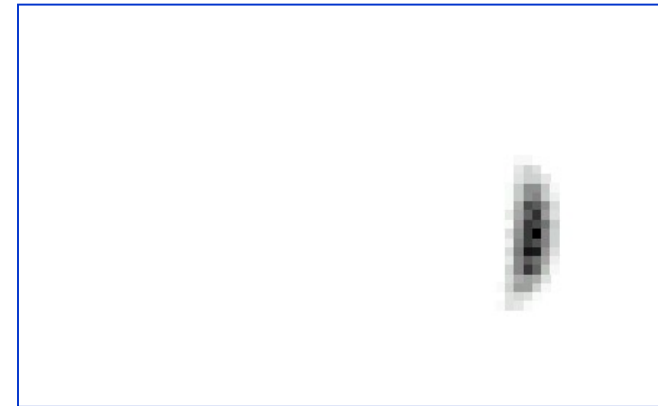
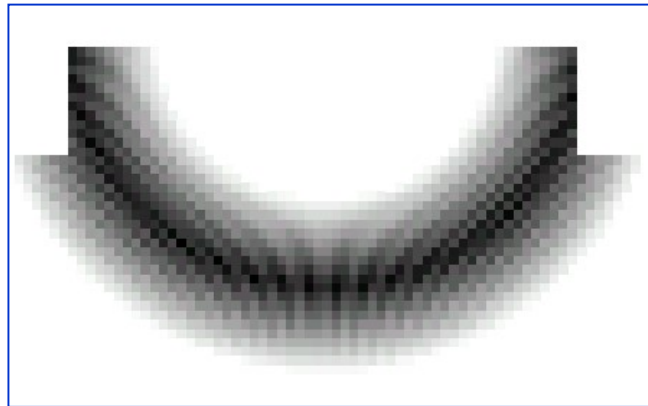
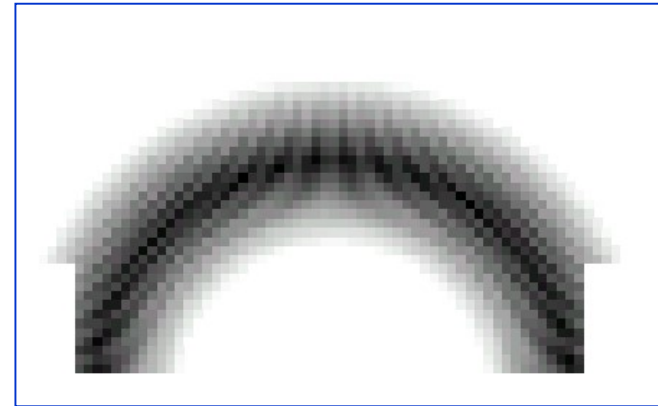
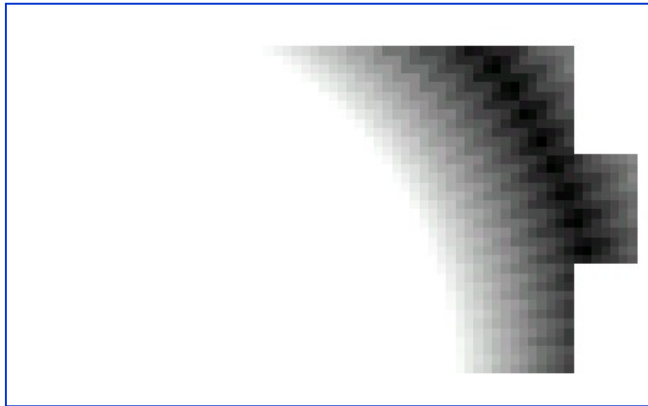
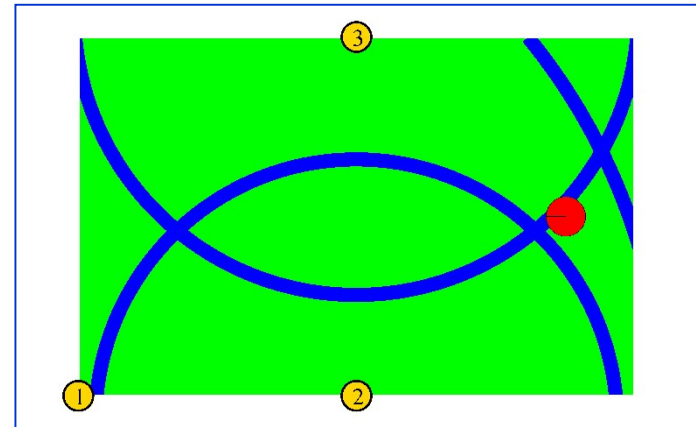


Peso dei campioni: $w = f/g$

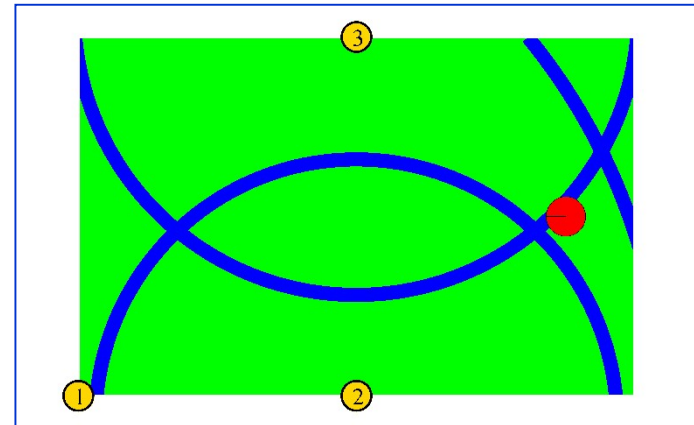
Importance Sampling con Resampling: Esempio



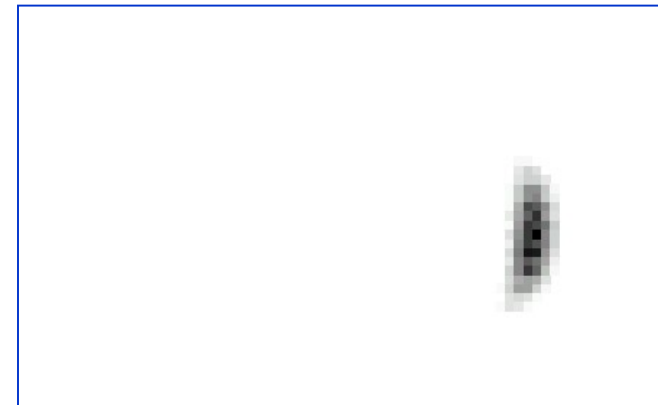
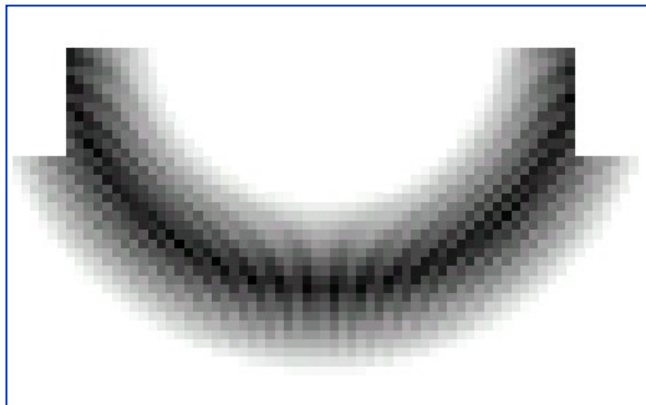
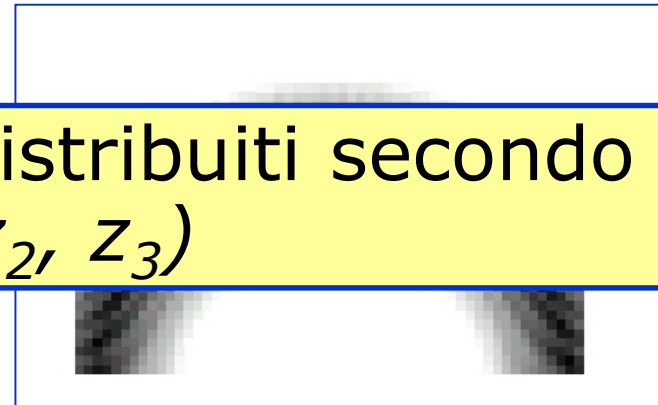
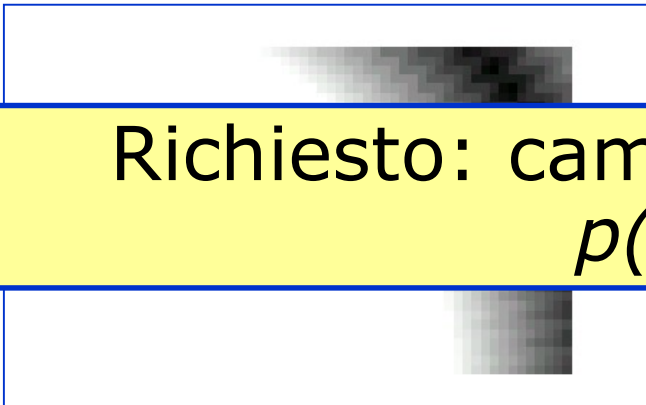
Distribuzioni



Distribuzioni

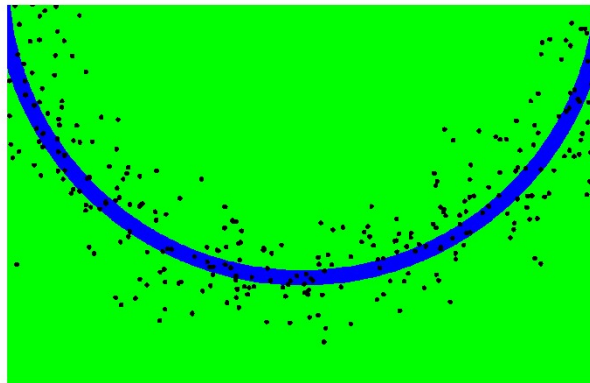
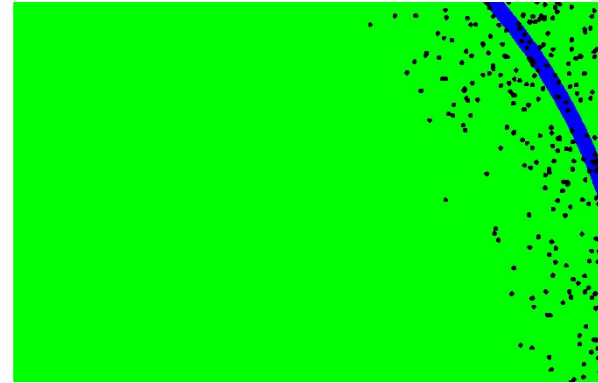
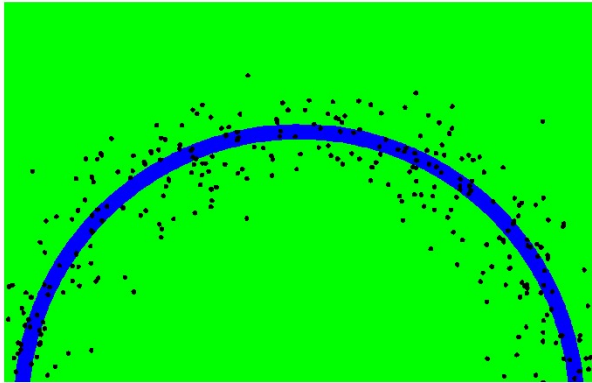


Richiesto: campioni distribuiti secondo
 $p(x | z_1, z_2, z_3)$



Campionamento

Estrarre campioni da $p(x|z_i)$



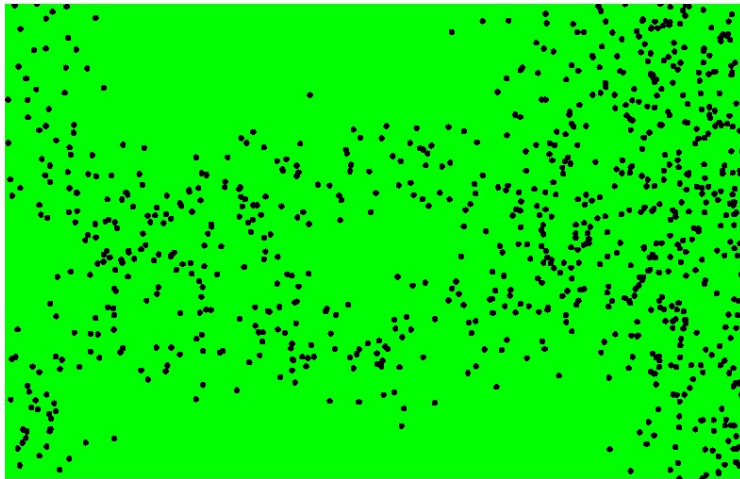
Importance Sampling con Resampling

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

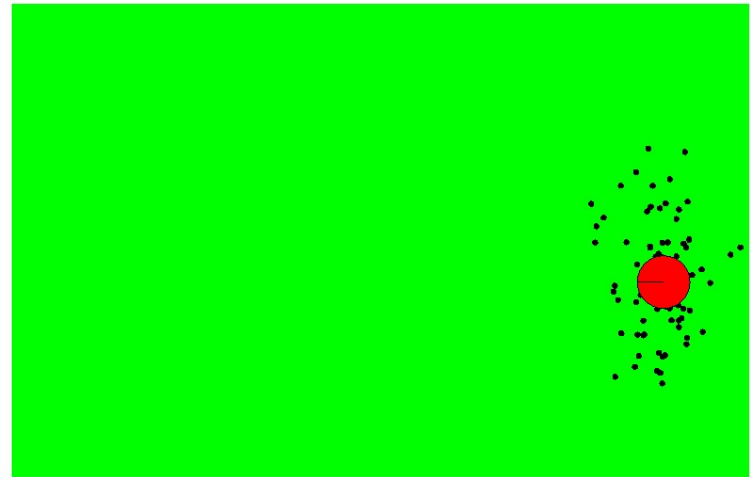
$$\text{Sampling distribution } g : p(x | z_l) = \frac{p(z_l | x) p(x)}{p(z_l)}$$

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

Importance Sampling con Resampling



Campioni pesati



Dopo il resampling

Importance Sampling

Campionamento di g

$$\frac{1}{M} \sum_{m=1}^M I(x^{[m]} \in A) \longrightarrow \int_A g(x) dx$$

Rapporto tra g ed f

$$w^{[m]} = \frac{f(x^{[m]})}{g(x^{[m]})}$$

Permette di ricostruire f

$$\left[\sum_{m=1}^M w^{[m]} \right]^{-1} \sum_{m=1}^M I(x^{[m]} \in A) w^{[m]} \longrightarrow \int_A f(x) dx$$

Importance Sampling

Nel caso del filtro particellare:

$$\begin{aligned} \text{Target}(x) &= \text{Bel}(x_t) && \rightarrow X_t \\ \text{Proposal}(x) &= p(x_t | u_t, x_{t-1}) \text{Bel}(x_{t-1}) && \rightarrow X'_t \\ w &= p(z_t | x_t) \end{aligned}$$

Particle Filter

Come $\text{bel}(x_t)$ si aggiorna da $\text{bel}(x_{t-1})$

Così X_t si aggiorna da X_{t-1}

Le ipotesi sono pesate:

$$S = \left\{ \left\langle s^{[i]}, w^{[i]} \right\rangle \mid i = 1, \dots, N \right\}$$

State hypothesis Importance weight

I campioni pesati usati per rappresentare il post:

$$p(x) = \sum_{i=1}^N w_i \cdot \delta_{s^{[i]}}(x)$$

Particle Filter: Algoritmo

Prima crea X'_t dal moto u_t rappresenta $bel'(t)$, quindi corregge con le osservazioni z_t

```
1:   Algorithm Particle_filter( $\mathcal{X}_{t-1}, u_t, z_t$ ):  
2:      $\bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset$   
3:     for  $m = 1$  to  $M$  do  
4:       sample  $x_t^{[m]} \sim p(x_t | u_t, x_{t-1}^{[m]})$   
5:        $w_t^{[m]} = p(z_t | x_t^{[m]})$   
6:        $\bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \langle x_t^{[m]}, w_t^{[m]} \rangle$   
7:     endfor  
8:     for  $m = 1$  to  $M$  do  
9:       draw  $i$  with probability  $\propto w_t^{[i]}$   
10:      add  $x_t^{[i]}$  to  $\mathcal{X}_t$   
11:    endfor  
12:    return  $\mathcal{X}_t$ 
```

Generation
Importance
factor

Resampling

Particle Filter: Algoritmo

Si parte da X_{t-1} che rapp $Bel_{t-1}(x)$

Si applica il modello di moto
 $p(x_{t,i} | u_t, x_{t-1,i})$ e si campiona X'_t

Si pesano i campioni rispetto a $P(z|x)$

Si effettua il ricampionamento per avere X_t

Particle Filter: Algoritmo

Target

$$\begin{aligned}
 & p(x_{0:t} \mid z_{1:t}, u_{1:t}) \\
 & \stackrel{\text{Bayes}}{=} \eta p(z_t \mid x_{0:t}, z_{1:t-1}, u_{1:t}) p(x_{0:t} \mid z_{1:t-1}, u_{1:t}) \\
 & \stackrel{\text{Markov}}{=} \eta p(z_t \mid x_t) p(x_{0:t} \mid z_{1:t-1}, u_{1:t}) \\
 & = \eta p(z_t \mid x_t) p(x_t \mid x_{0:t-1}, z_{1:t-1}, u_{1:t}) p(x_{0:t-1} \mid z_{1:t-1}, u_{1:t}) \\
 & \stackrel{\text{Markov}}{=} \eta p(z_t \mid x_t) p(x_t \mid x_{t-1}, u_t) p(x_{0:t-1} \mid z_{1:t-1}, u_{1:t-1})
 \end{aligned}$$

Proposal

$$\begin{aligned}
 & p(x_t \mid x_{t-1}, u_t) \text{bel}(x_{0:t-1}) \\
 & = p(x_t \mid x_{t-1}, u_t) p(x_{0:t-1} \mid z_{0:t-1}, u_{0:t-1})
 \end{aligned}$$

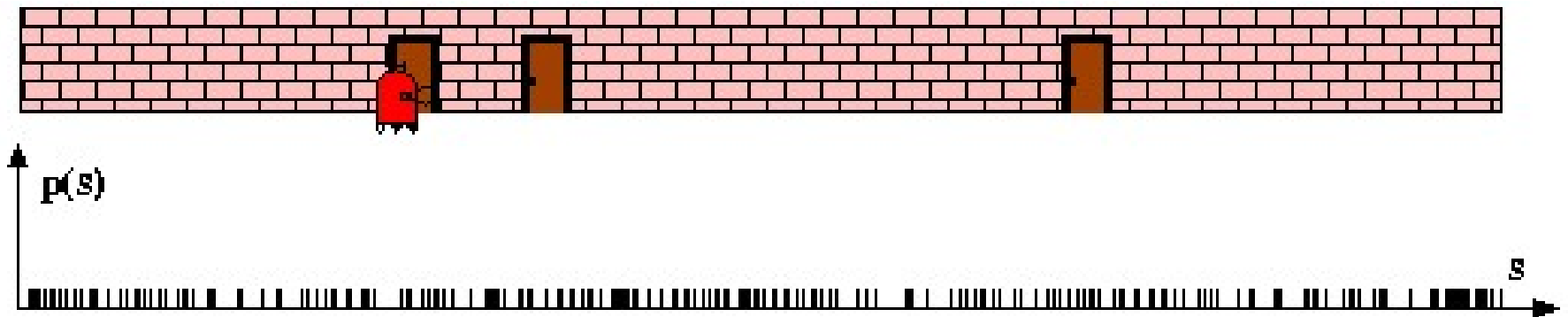
Peso

$$\begin{aligned}
 w_t^{[m]} & = \frac{\text{target distribution}}{\text{proposal distribution}} \\
 & = \frac{\eta p(z_t \mid x_t) p(x_t \mid x_{t-1}, u_t) p(x_{0:t-1} \mid z_{1:t-1}, u_{1:t-1})}{p(x_t \mid x_{t-1}, u_t) p(x_{0:t-1} \mid z_{0:t-1}, u_{0:t-1})} \\
 & = \eta p(z_t \mid x_t)
 \end{aligned}$$

Target

$$\eta w_t^{[m]} p(x_t \mid x_{t-1}, u_t) p(x_{0:t-1} \mid z_{0:t-1}, u_{0:t-1}) = \text{bel}(x_{0:t})$$

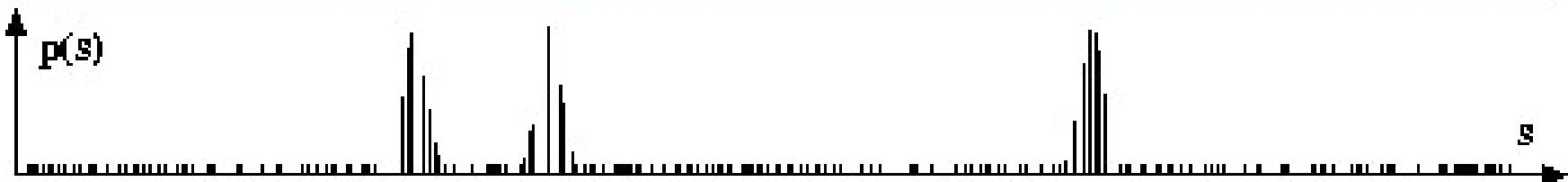
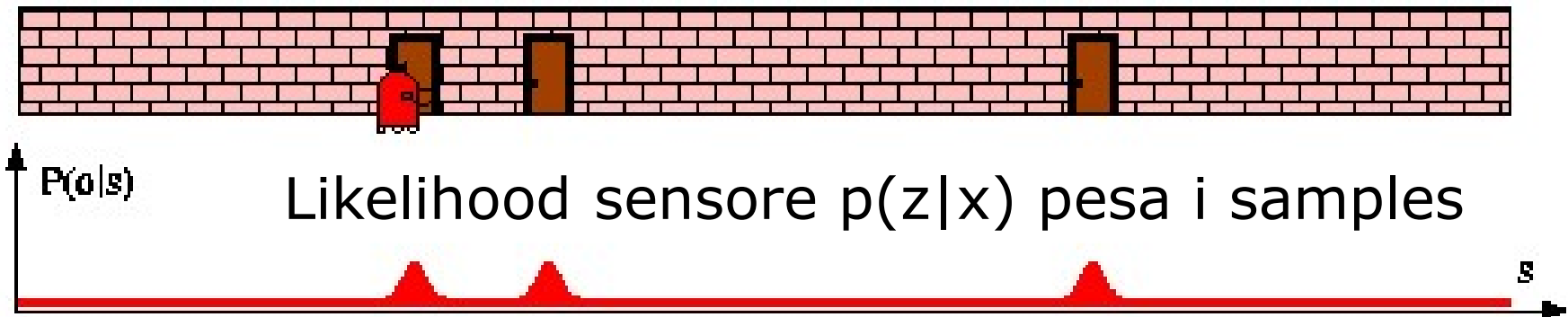
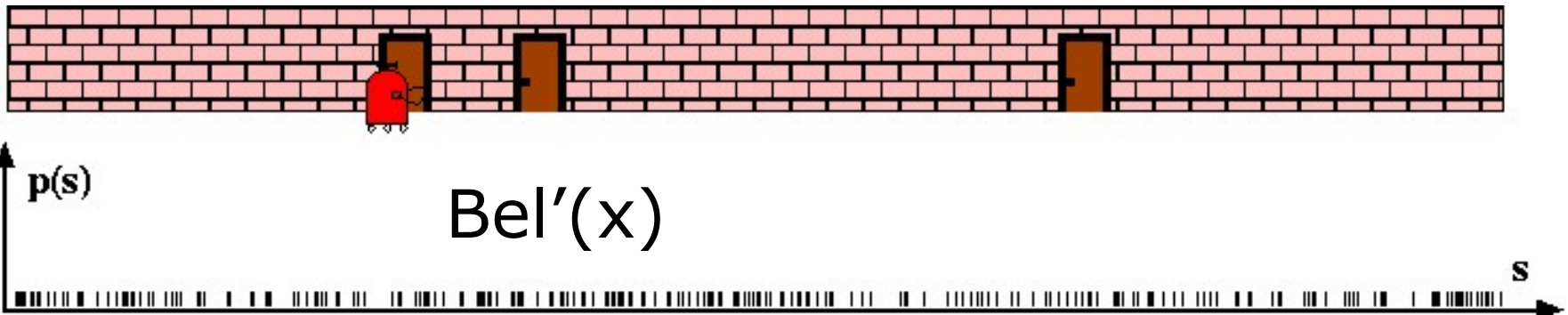
Particle Filters



La posa iniziale sono particelle prese in modo random

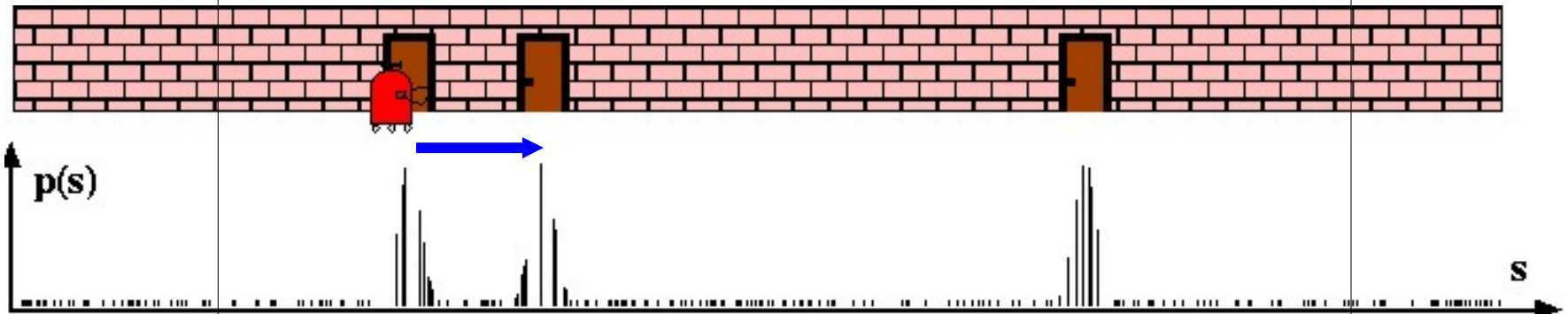
Informazione Sensori: Importance Sampling

$$\begin{aligned} Bel(x) &\leftarrow \alpha p(z|x) Bel^-(x) \\ w &\leftarrow \frac{\alpha p(z|x) Bel^-(x)}{Bel^-(x)} = \alpha p(z|x) \end{aligned}$$

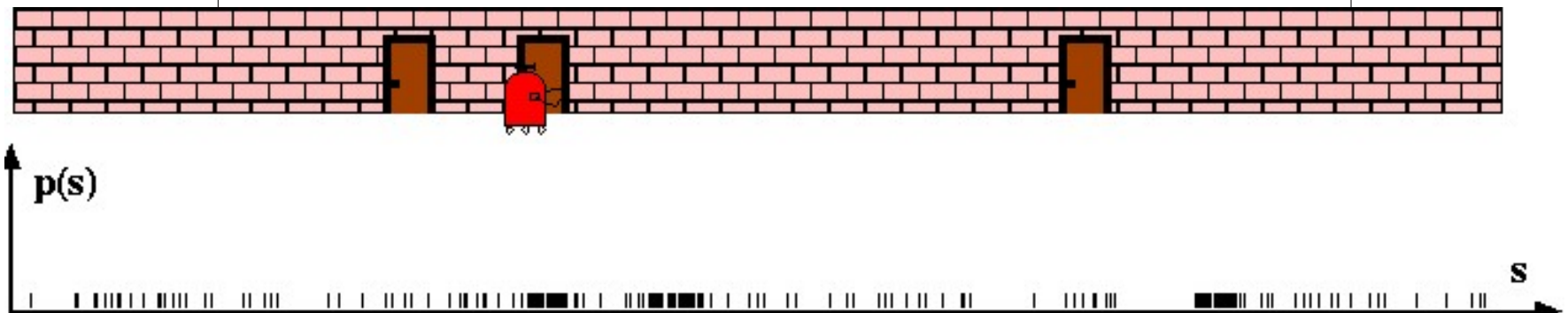


Robot Motion

$$Bel^-(x) \leftarrow \int p(x|u, x') Bel(x') dx'$$

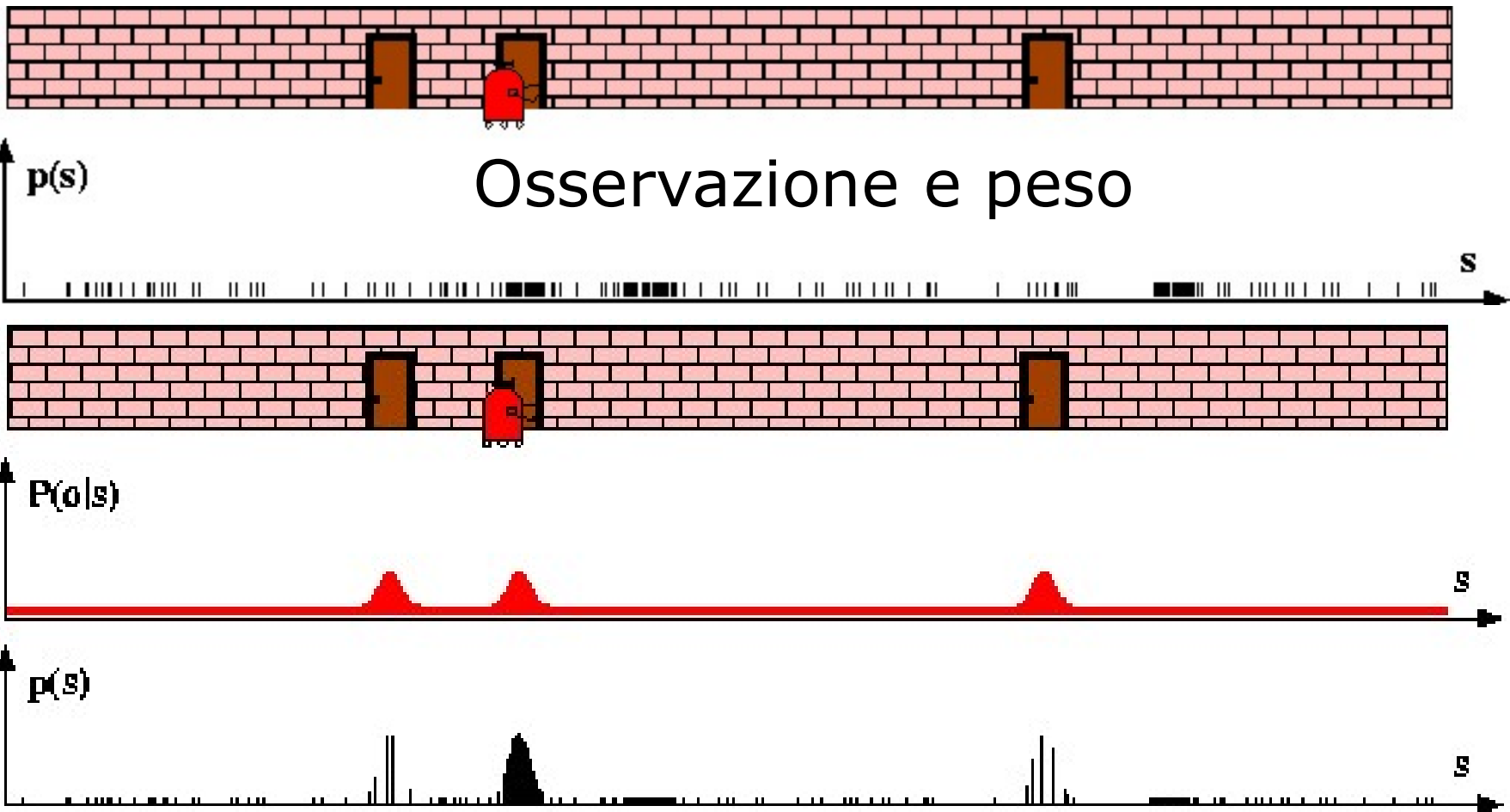


Ricampionamento e spostamento $Bel'(x)$



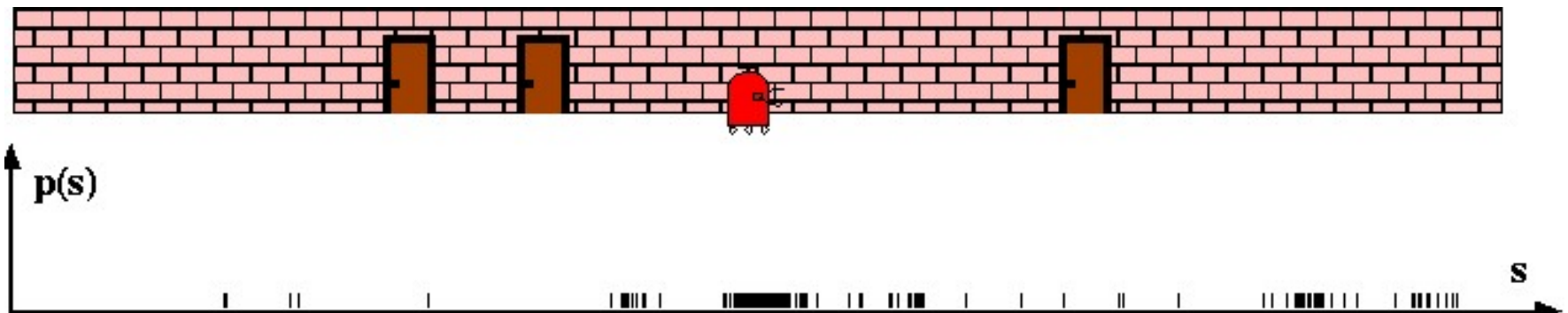
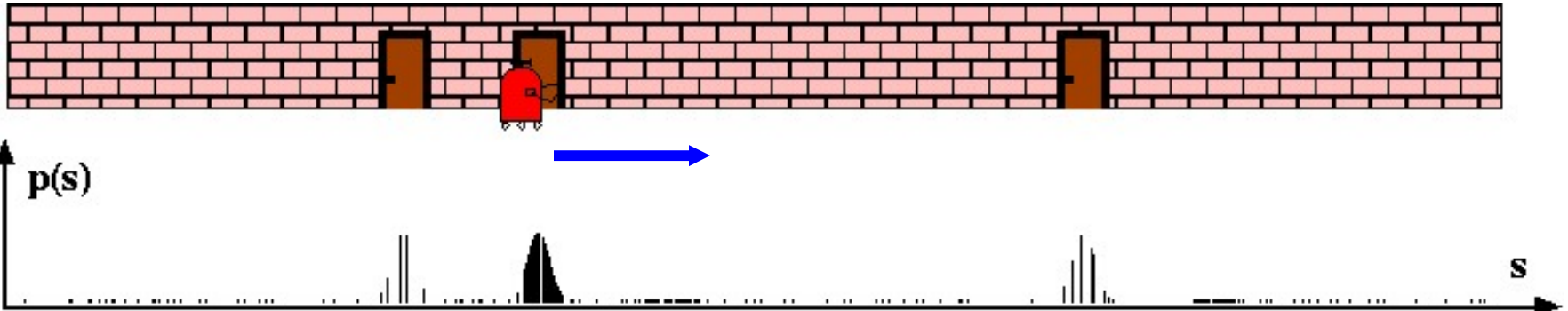
Informazione Sensori: Importance Sampling

$$\begin{aligned} Bel(x) &\leftarrow \alpha p(z|x) Bel^-(x) \\ w &\leftarrow \frac{\alpha p(z|x) Bel^-(x)}{Bel^-(x)} = \alpha p(z|x) \end{aligned}$$



Robot Motion

$$Bel^-(x) \leftarrow \int p(x|u, x') Bel(x') dx'$$

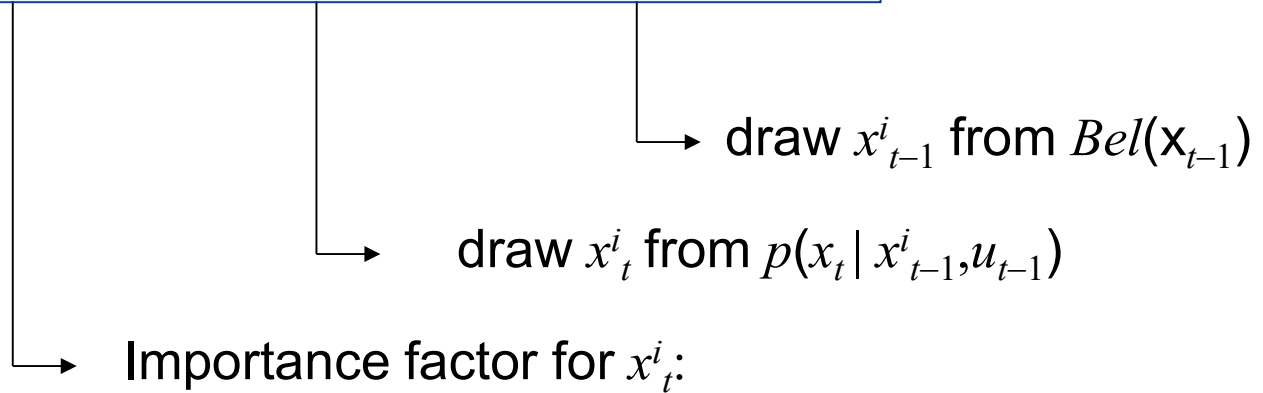


Algoritmo del Particle Filter

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

Algoritmo Particle Filter

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



Sampling da Bel'
 Peso da P(z|x)

$$w_t^i = \frac{\text{target distribution}}{\text{proposal distribution}}$$

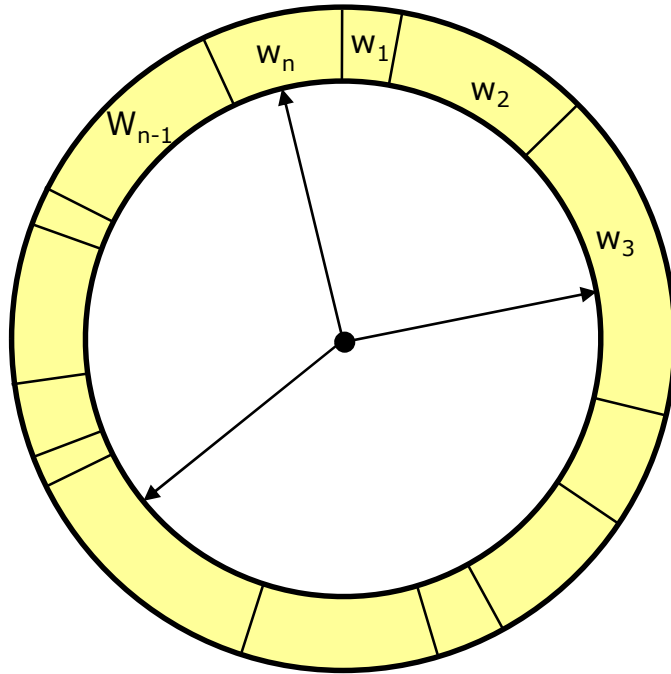
$$= \frac{\eta p(z_t | x_t) p(x_t | x_{t-1}, u_{t-1}) Bel(x_{t-1})}{p(x_t | x_{t-1}, u_{t-1}) Bel(x_{t-1})}$$

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

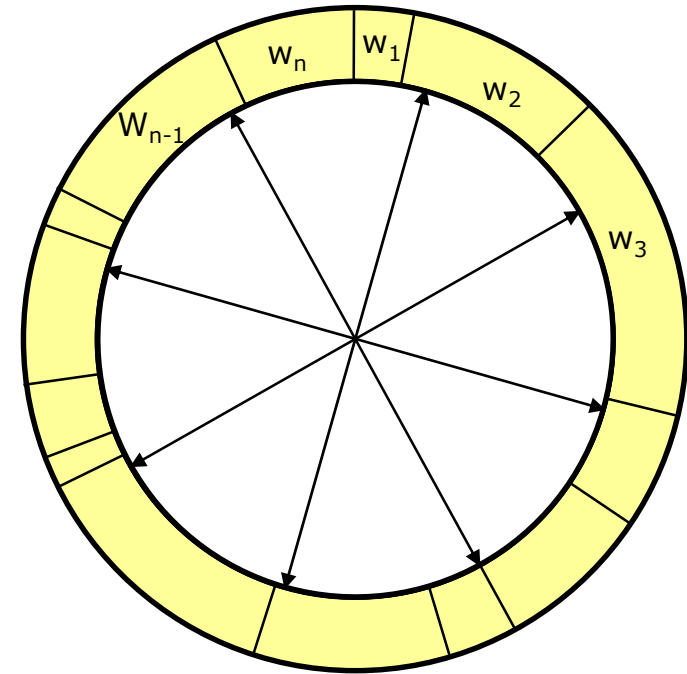
Resampling

- **Dato**: Insieme S di campioni pesati.
- **Desiderata** : Campione Random, dove la prob di estrarre x_i è proporzionale a w_i .
- Fatto n volte con rimpiazzamento per generare il nuovo insieme di campioni S' .

Resampling



- Roulette wheel
- Binary search, $n \log n$



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

Resampling

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

2. $S' = \emptyset, c_1 = w^1$

3. **For** $i = 2 \dots n$ *Generate cdf*

4. $c_i = c_{i-1} + w^i$

5. $u_1 \sim U]0, n^{-1}]$, $i = 1$ *Initialize threshold*

6. **For** $j = 1 \dots n$ *Draw samples ...*

7. **While** ($u_j > c_i$) *Skip until next threshold reached*

8. $i = i + 1$

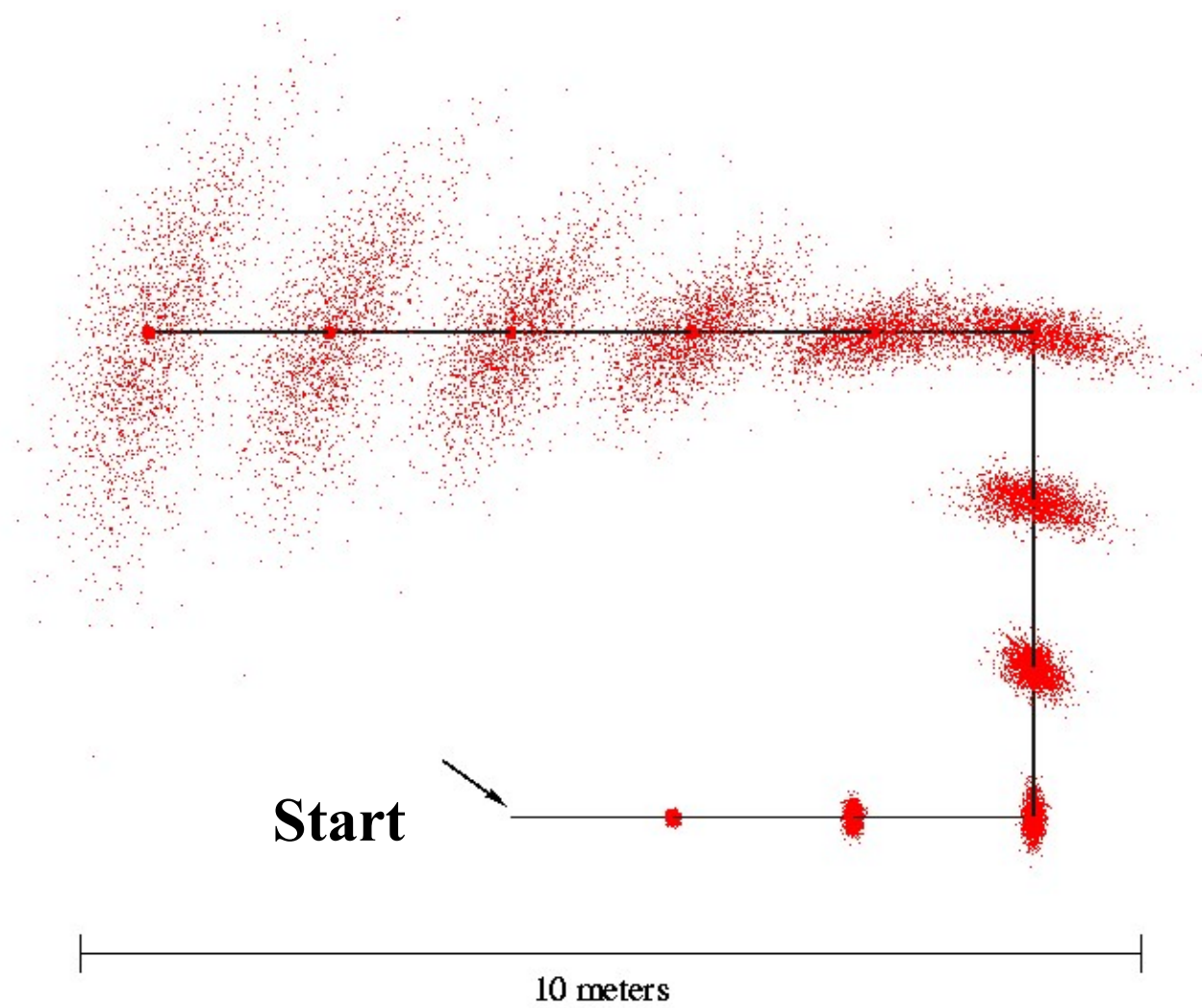
9. $S' = S' \cup \{ \langle x^i, n^{-1} \rangle \}$ *Insert*

10. $u_{j+1} = u_j + n^{-1}$ *Increment threshold*

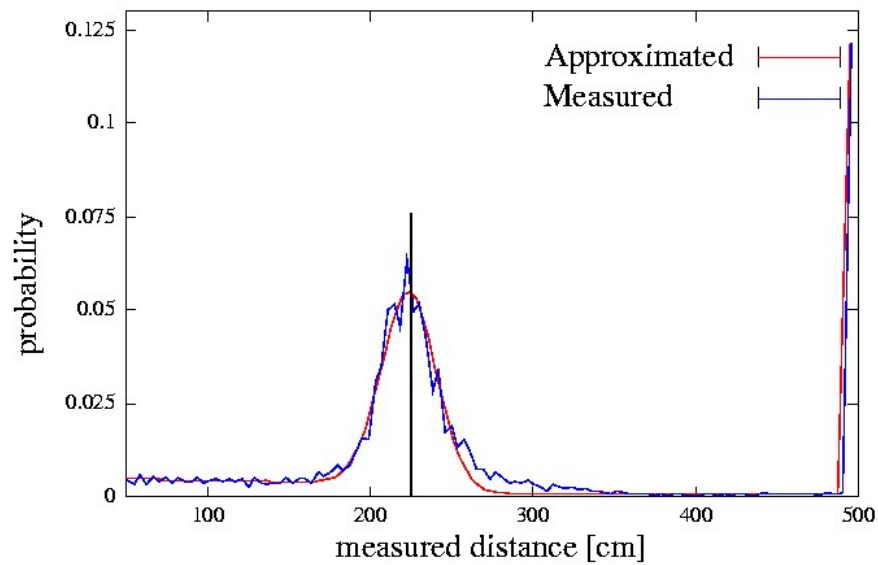
11. **Return** S'

Also called **stochastic universal sampling**

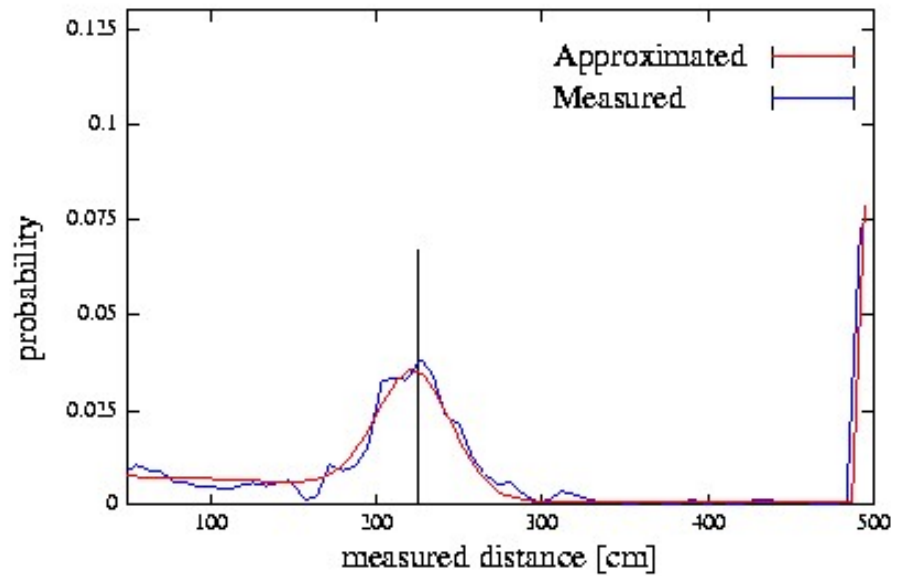
Modello di Moto



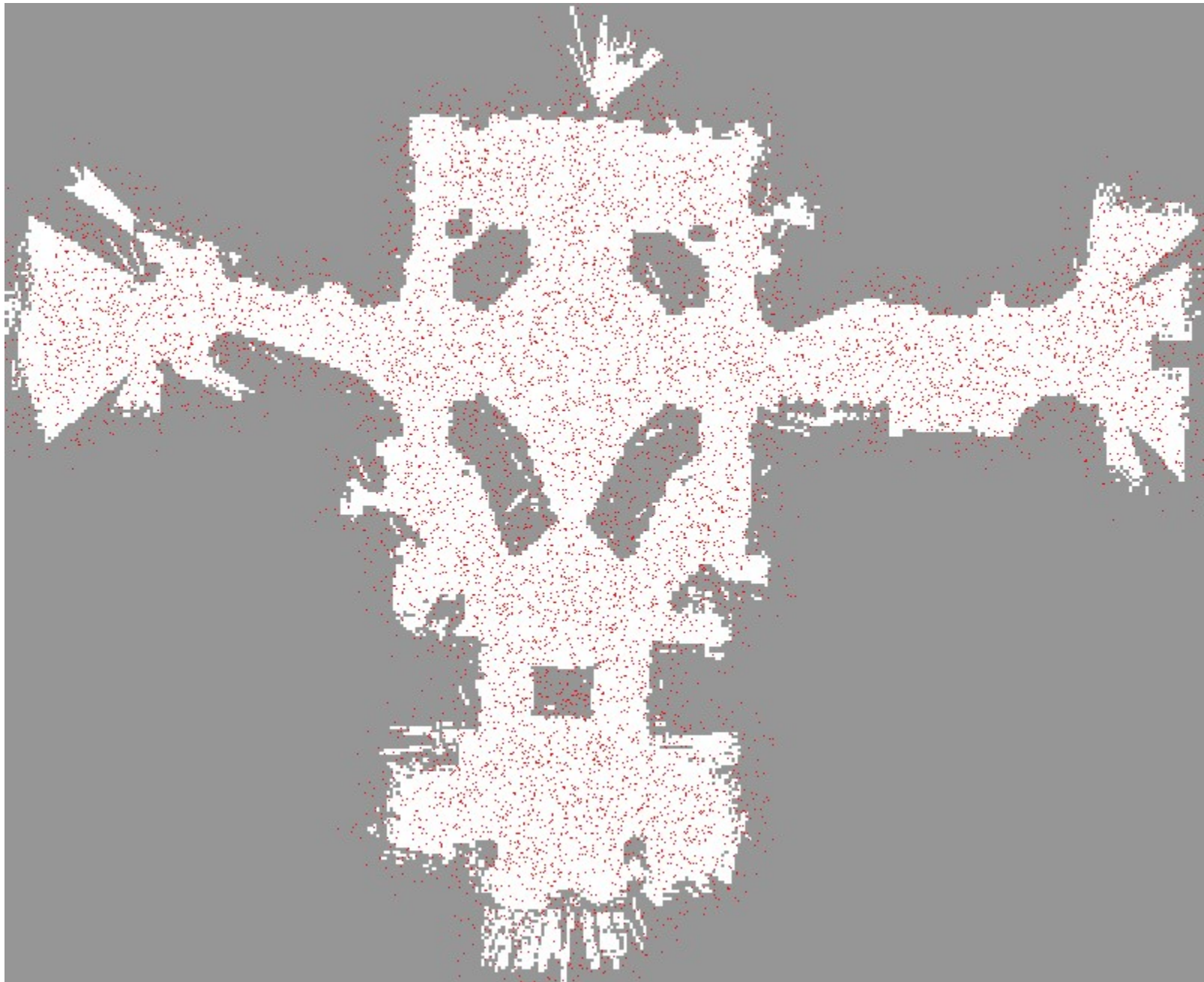
Proximity Sensor Model

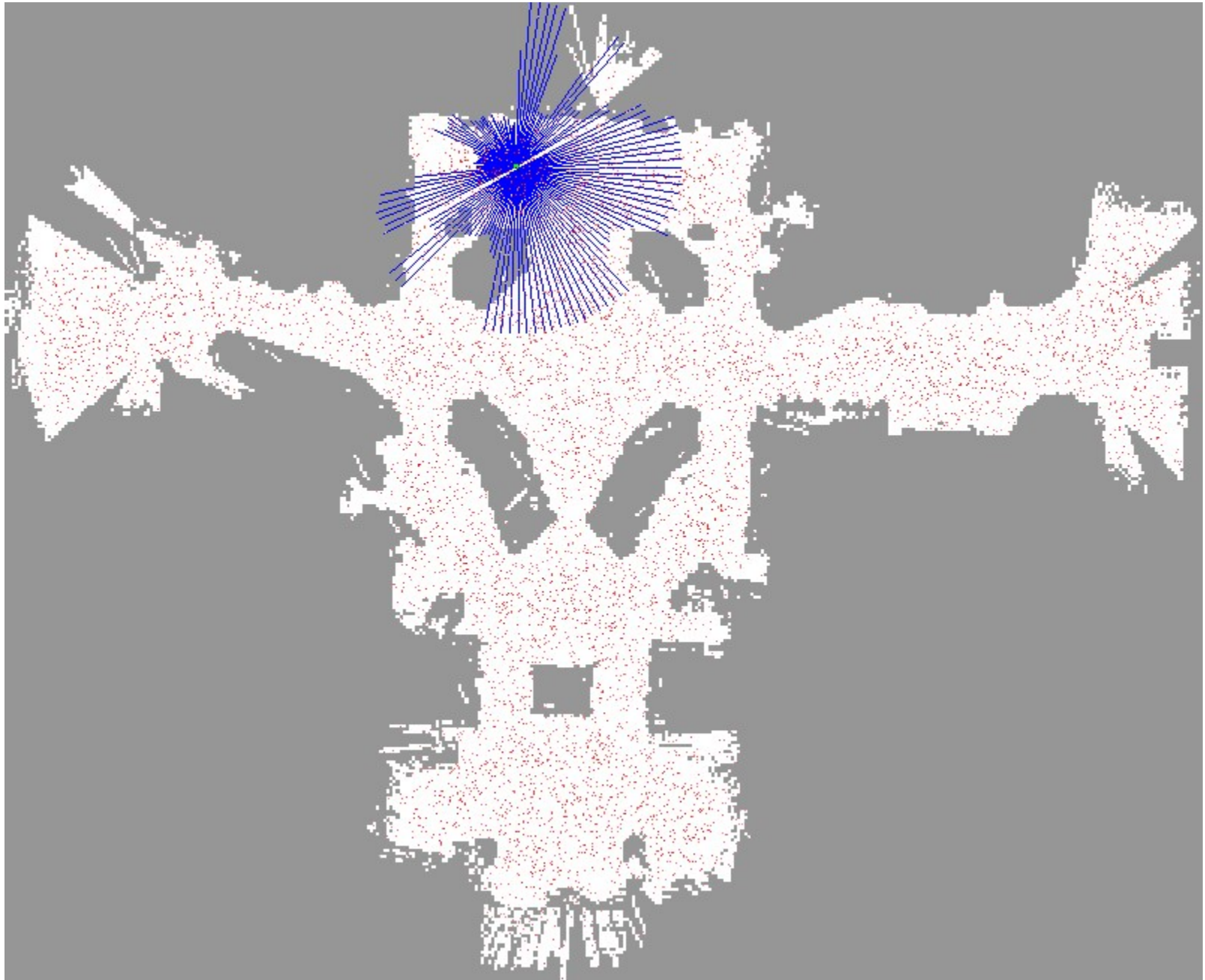


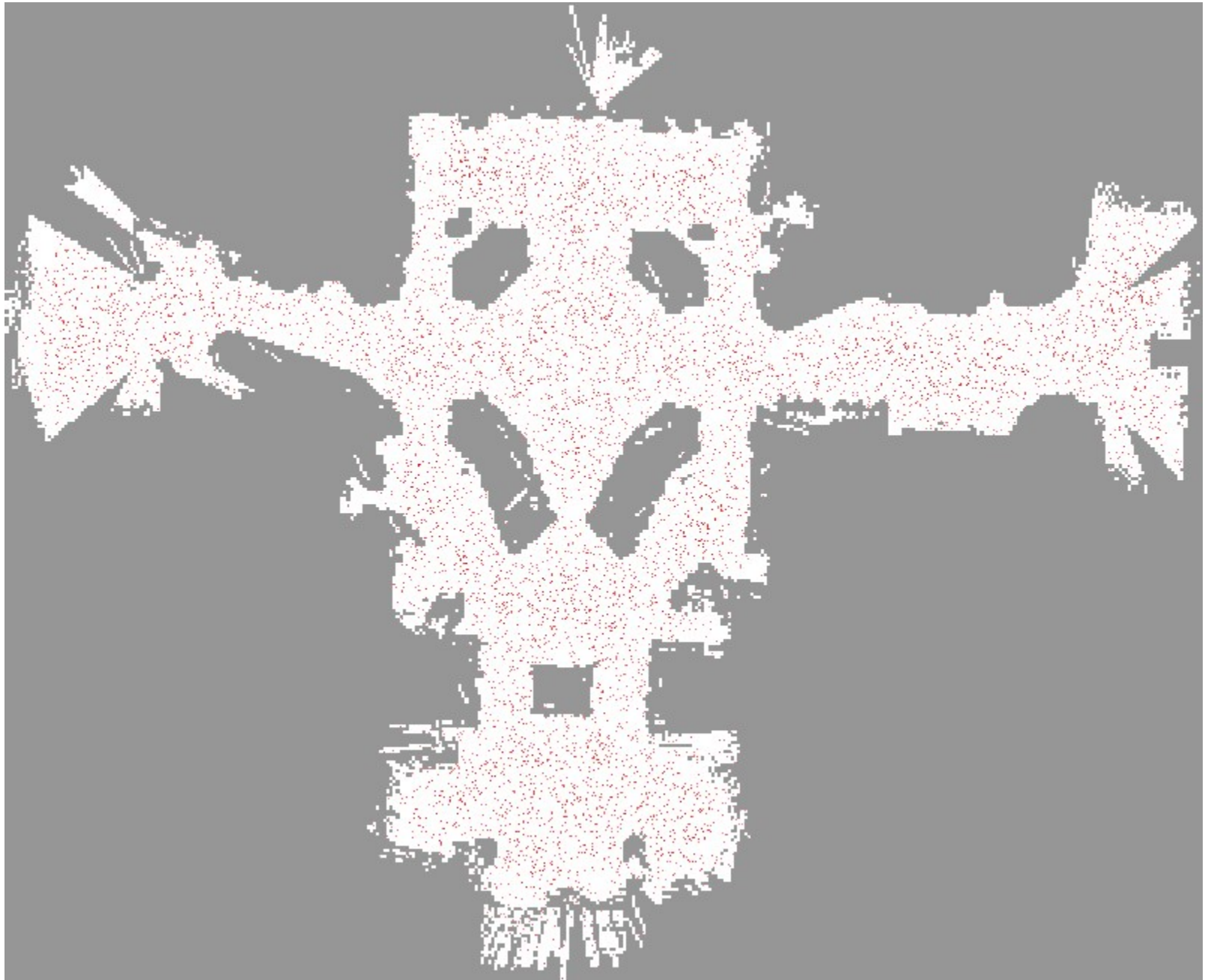
Laser sensor

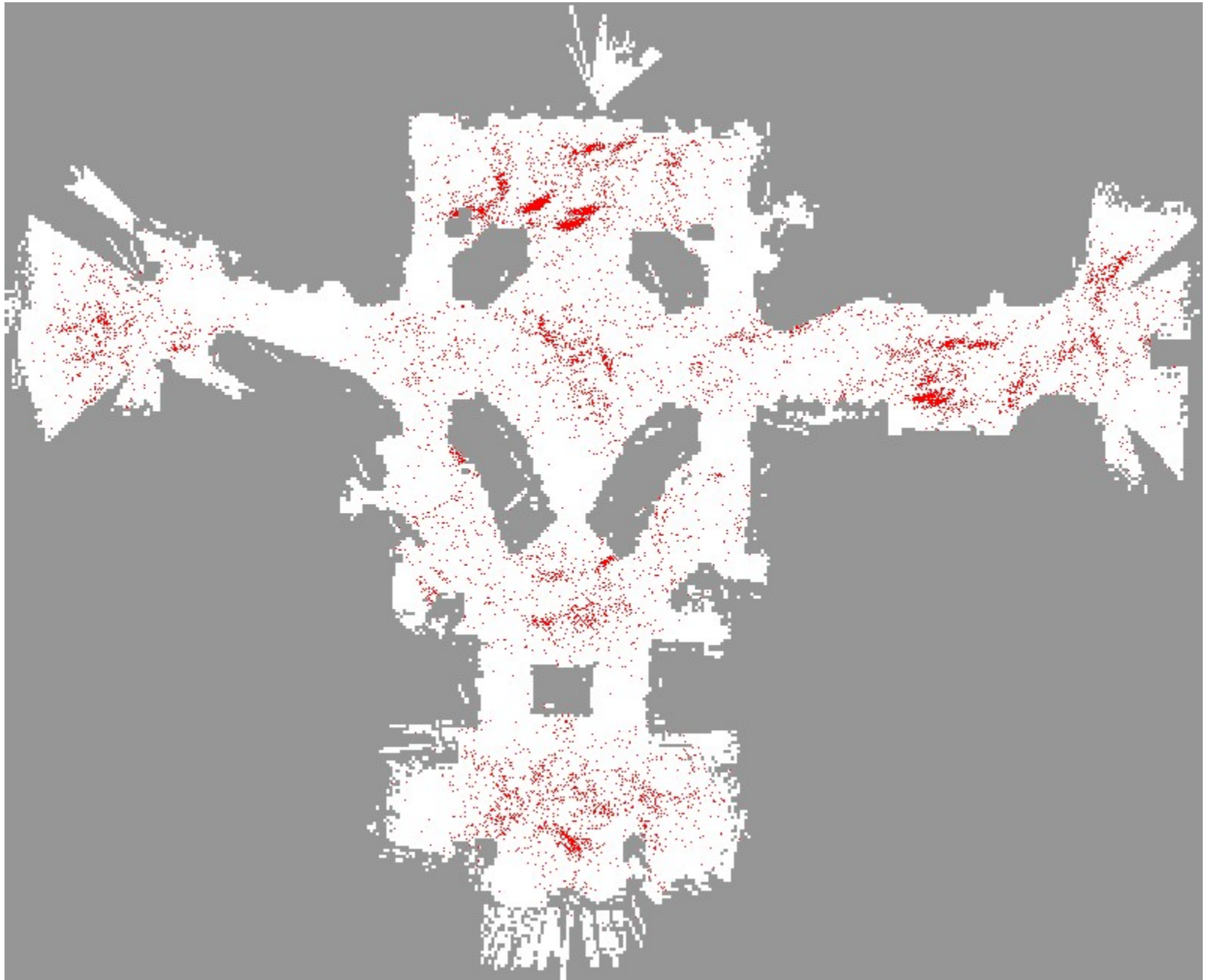


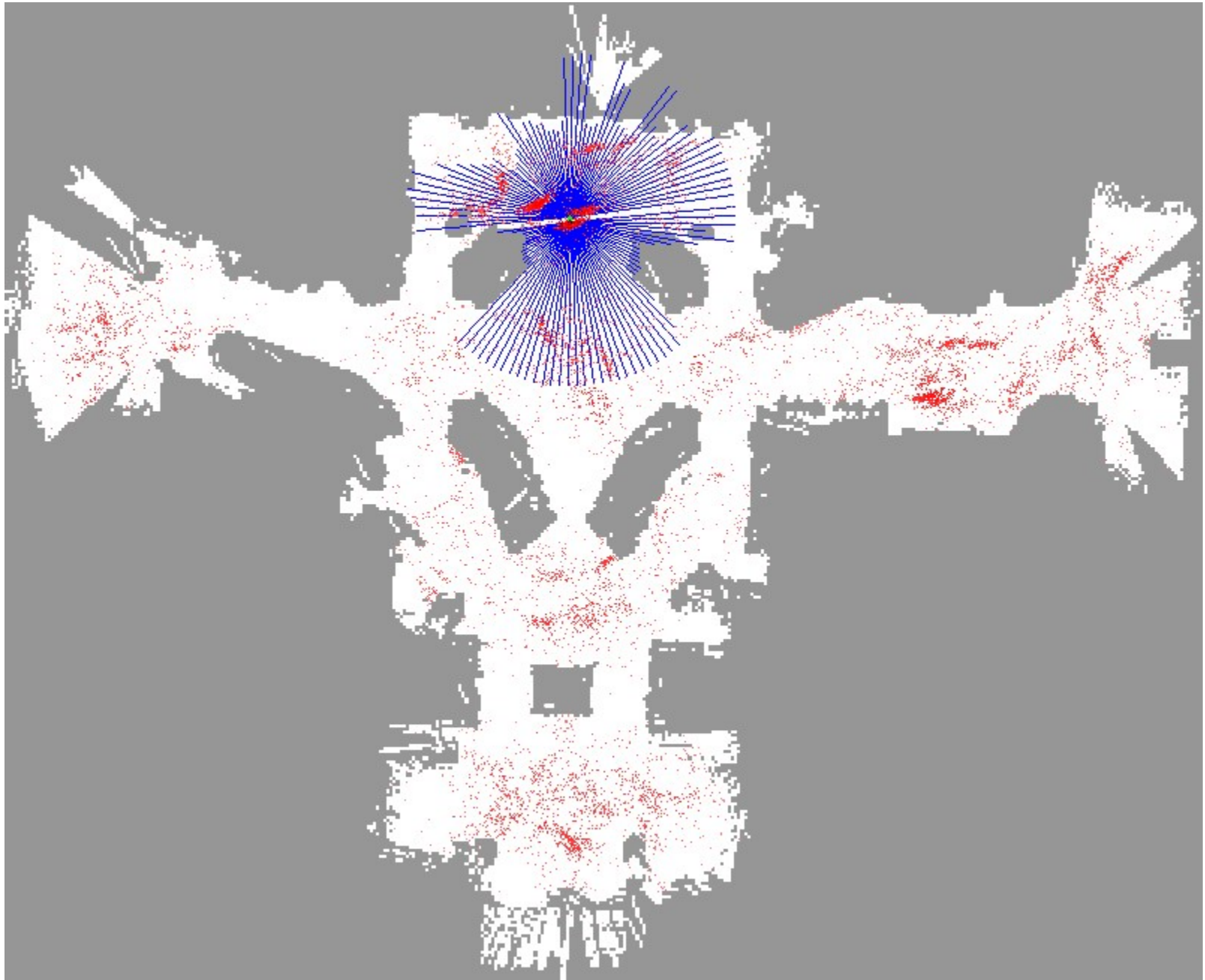
Sonar sensor

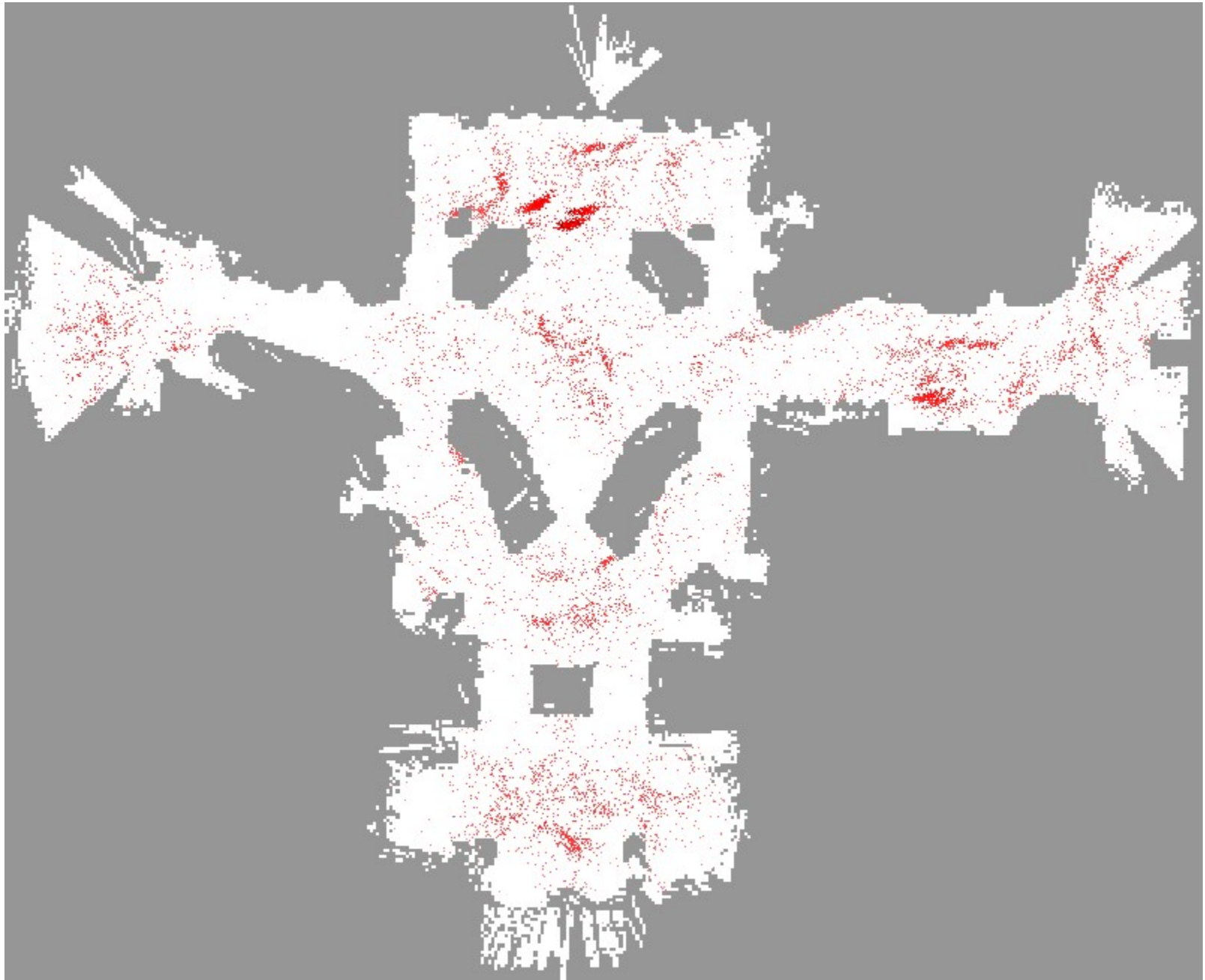


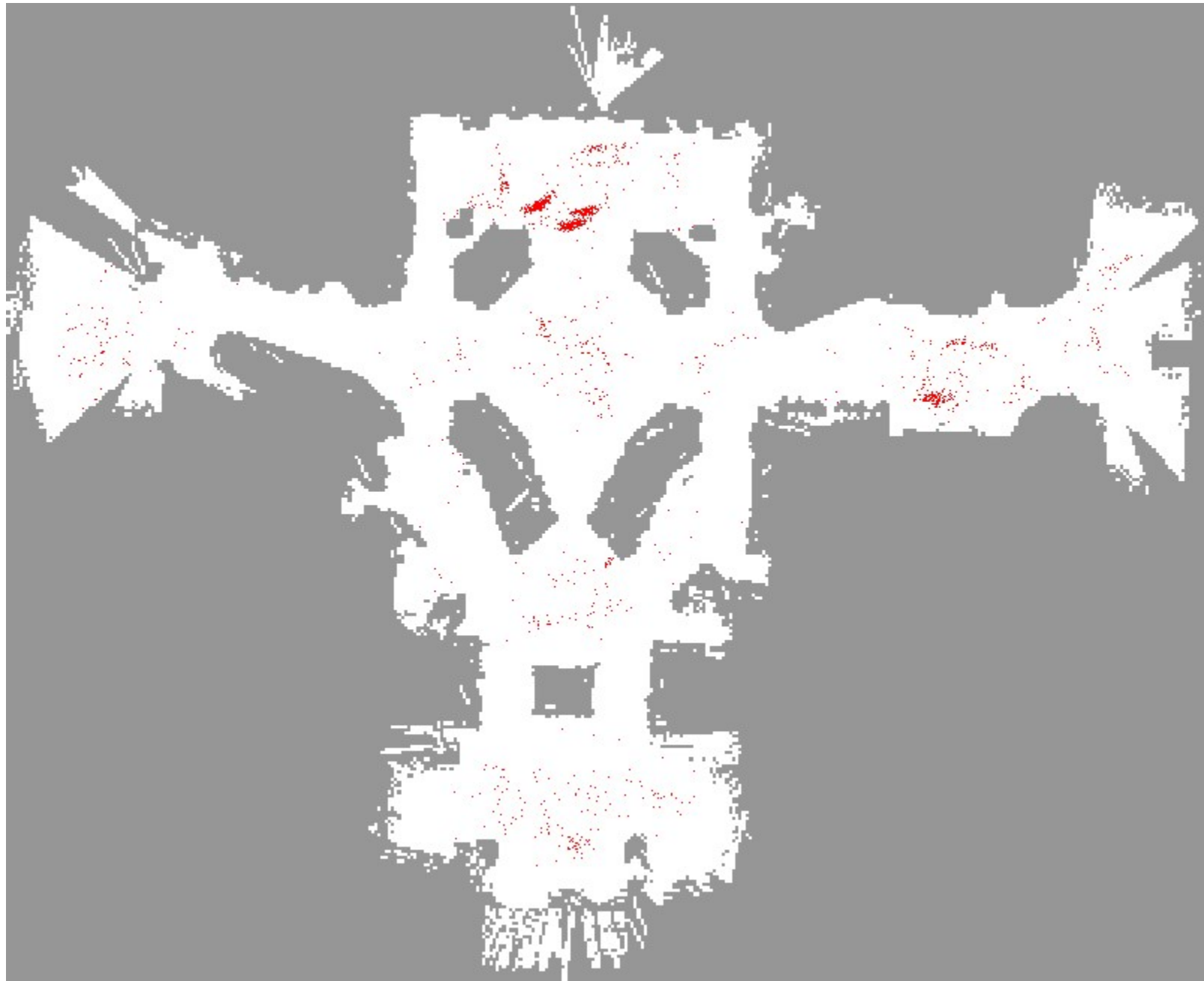


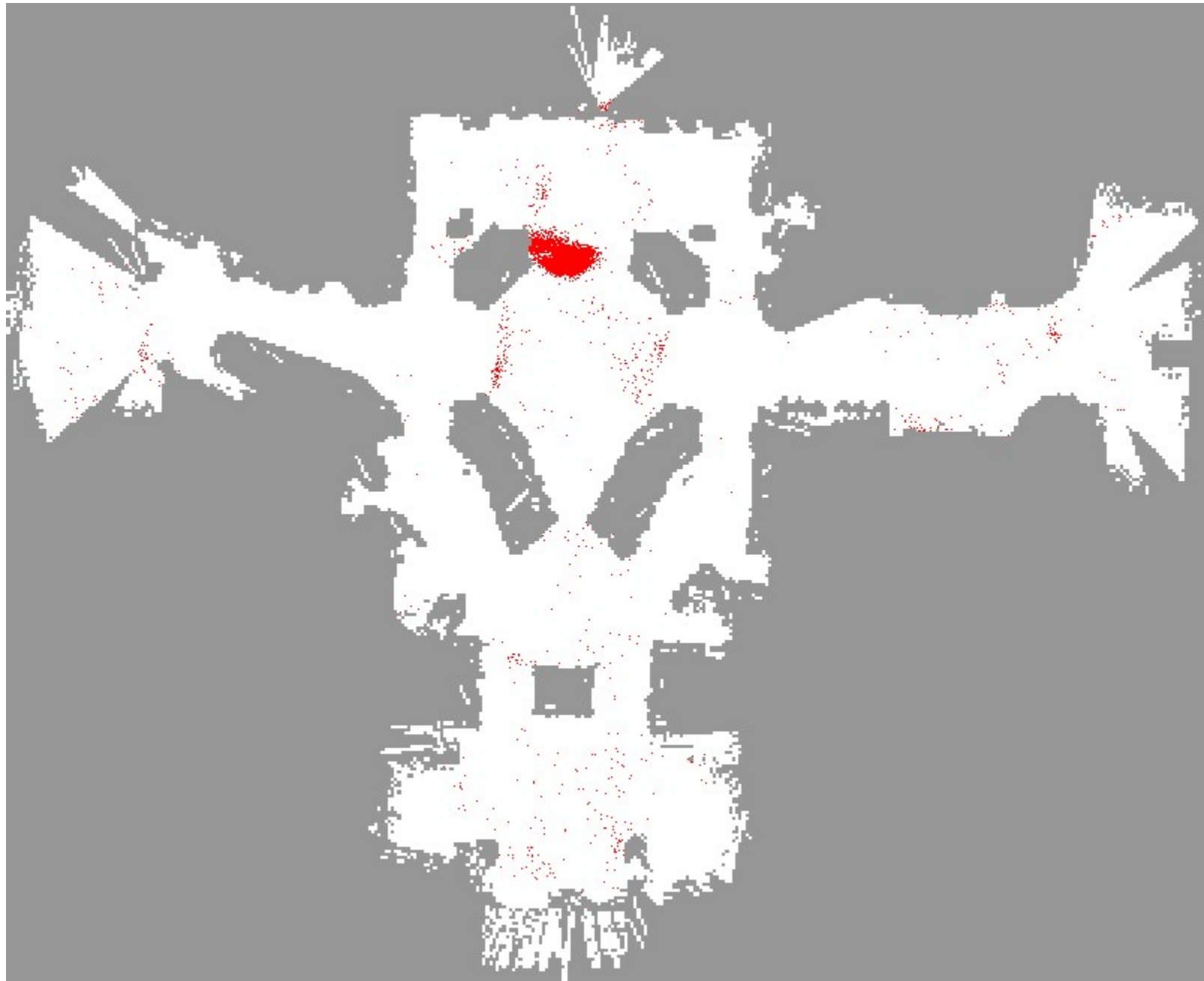


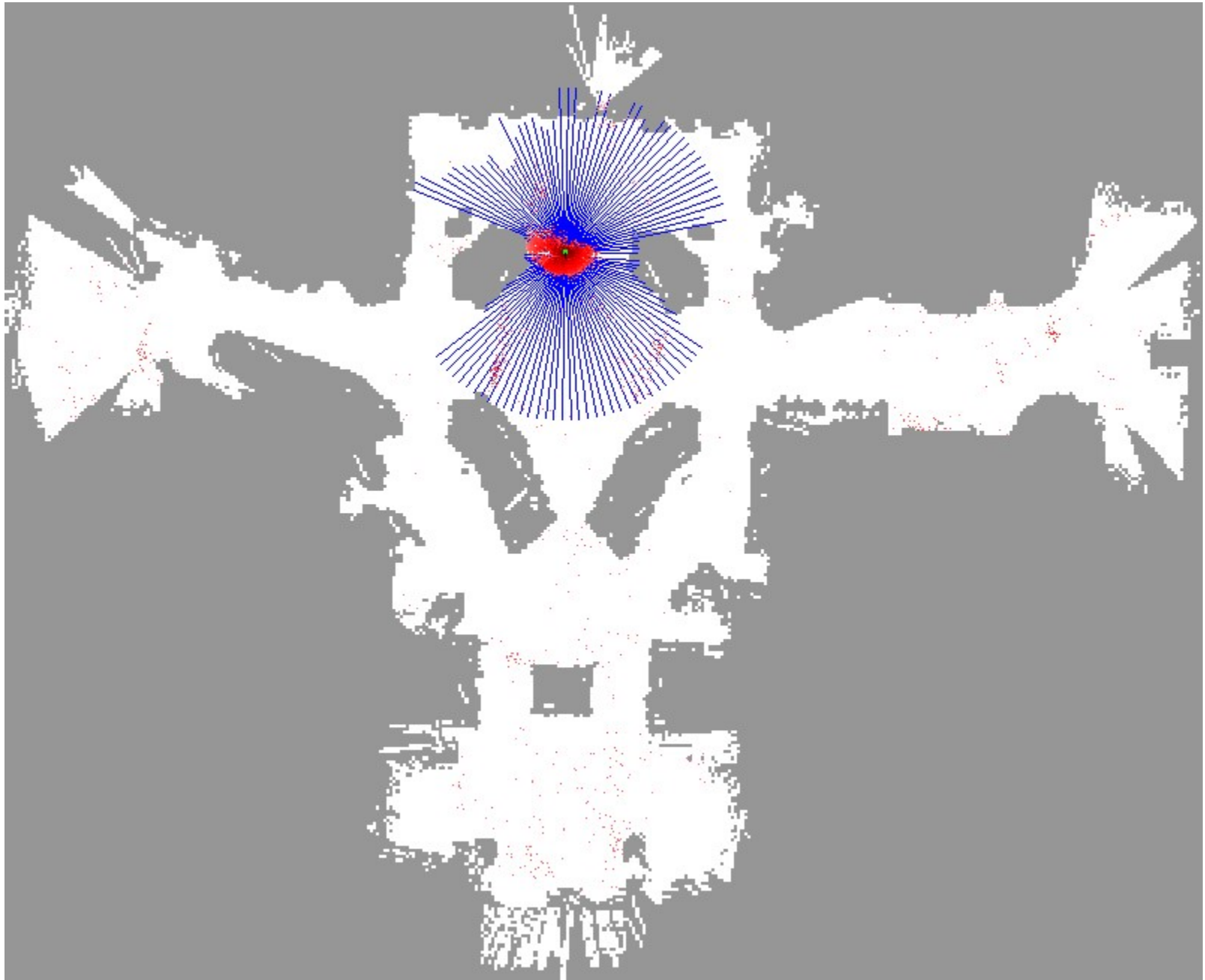


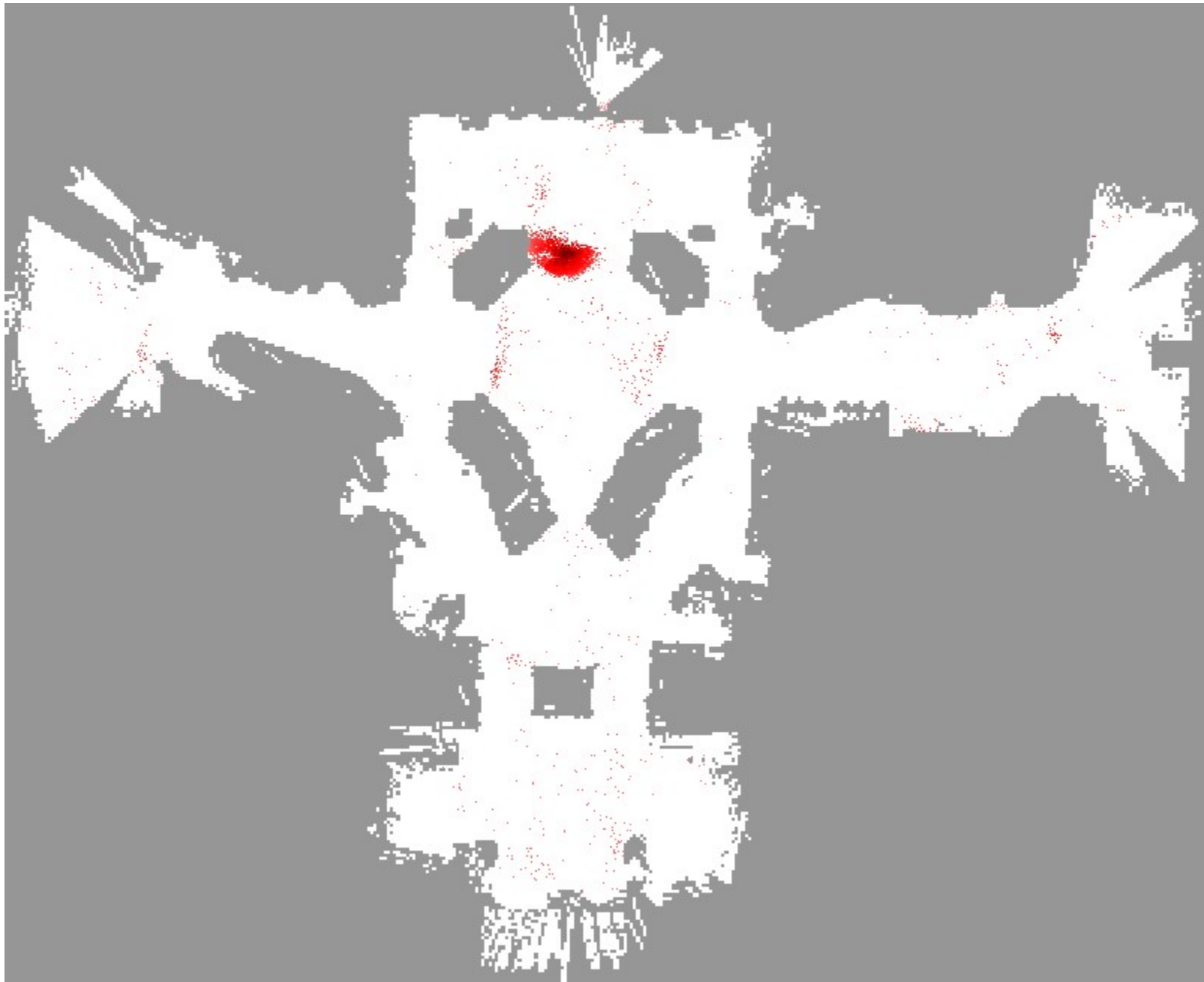


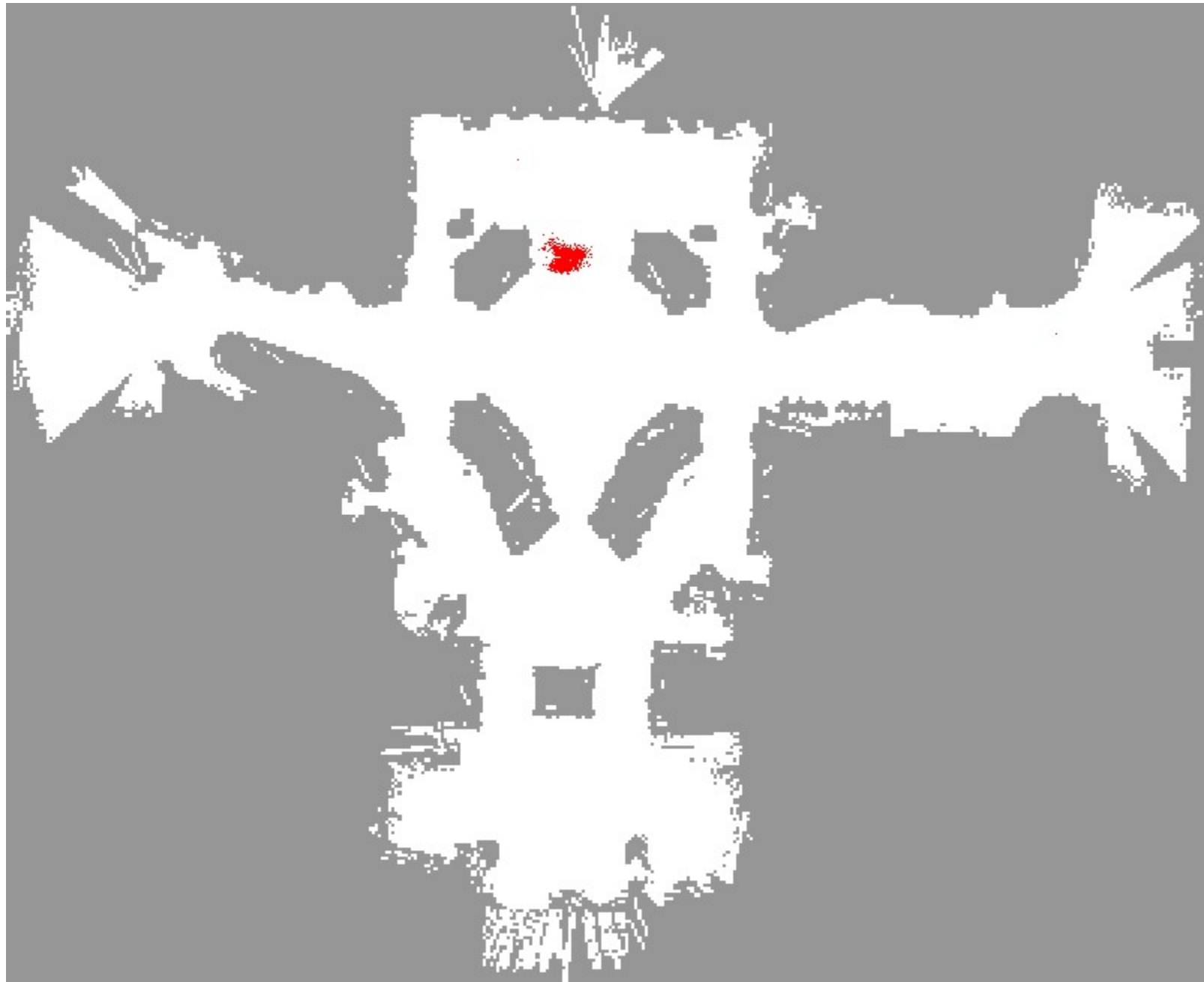


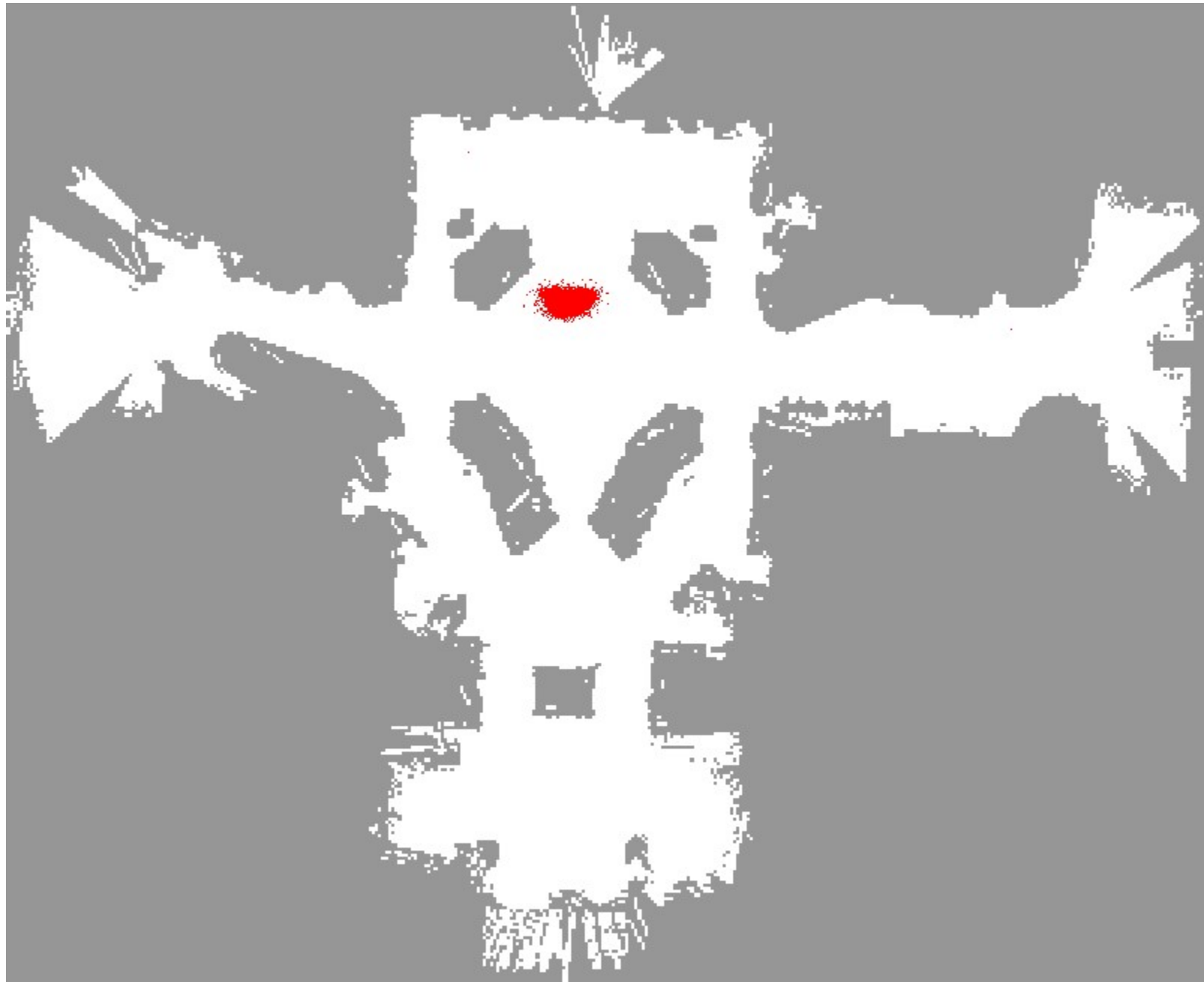


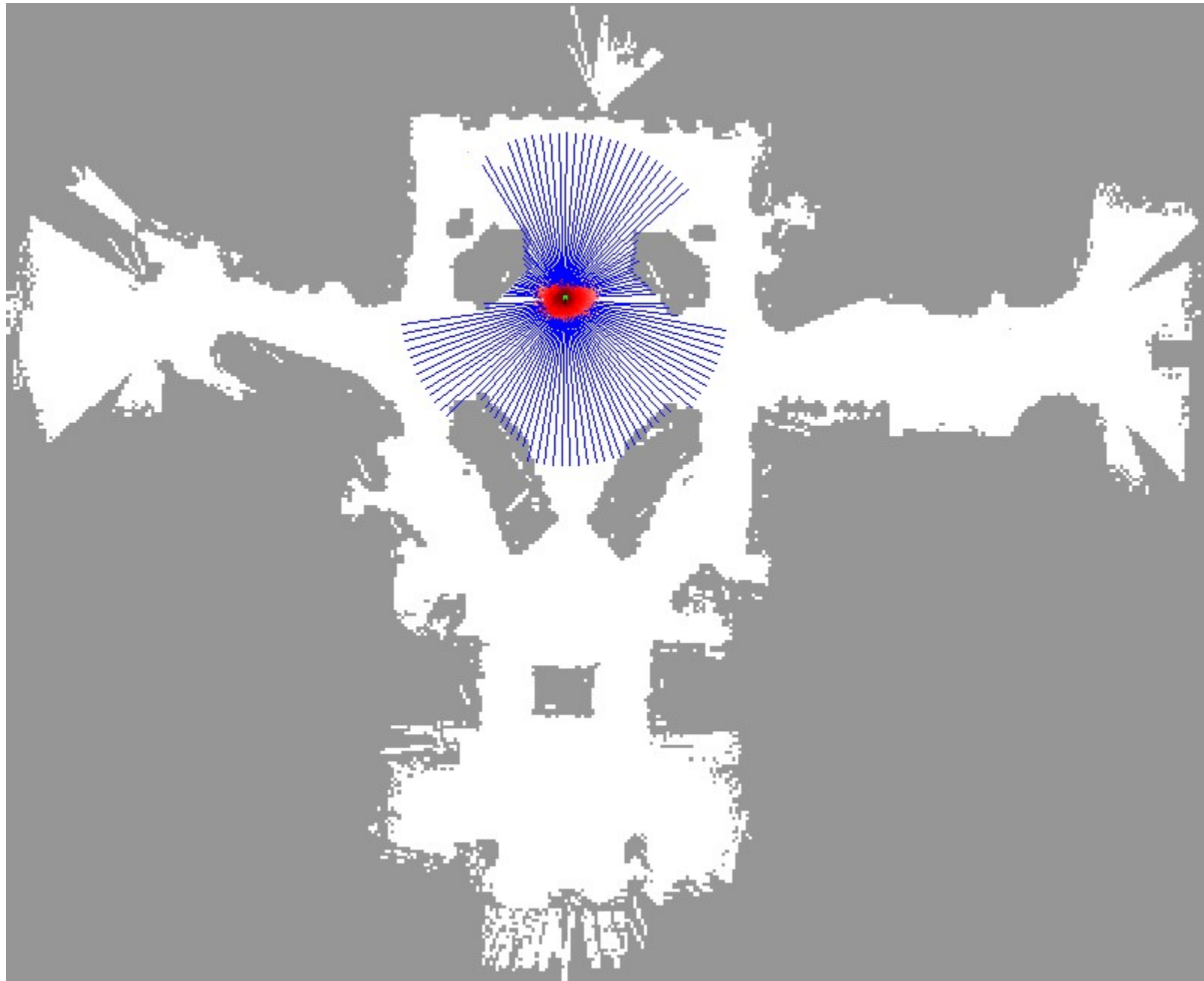


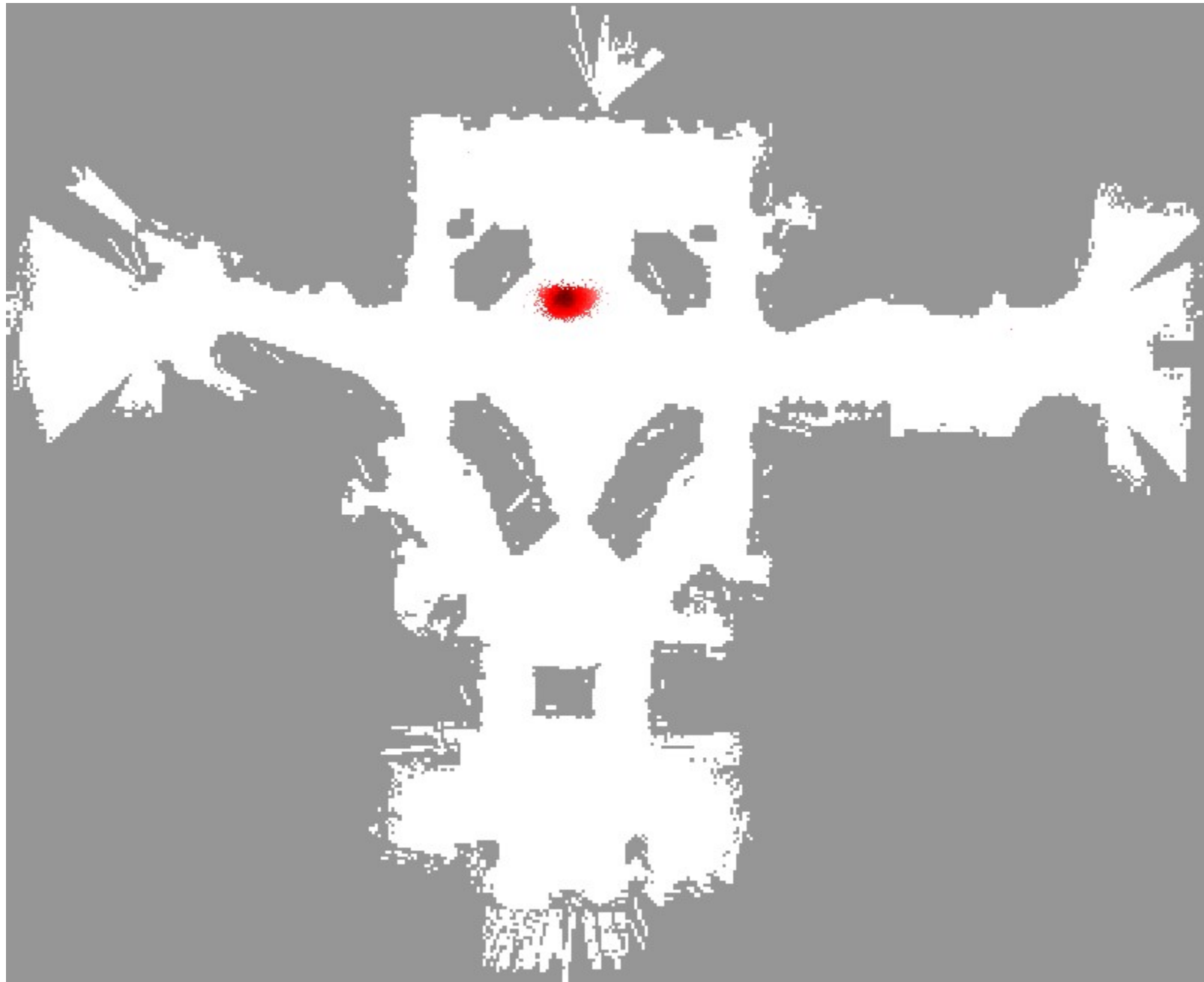


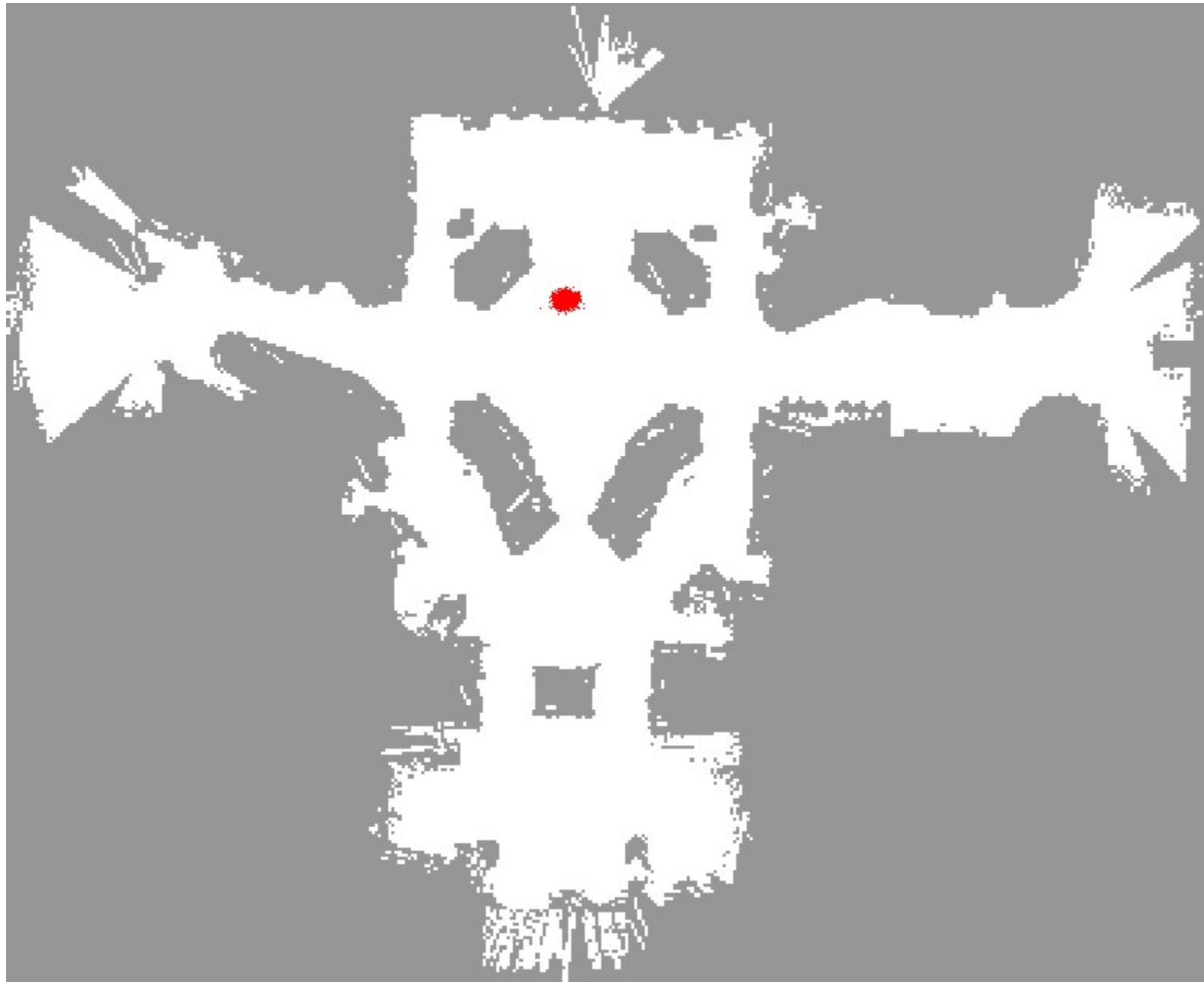


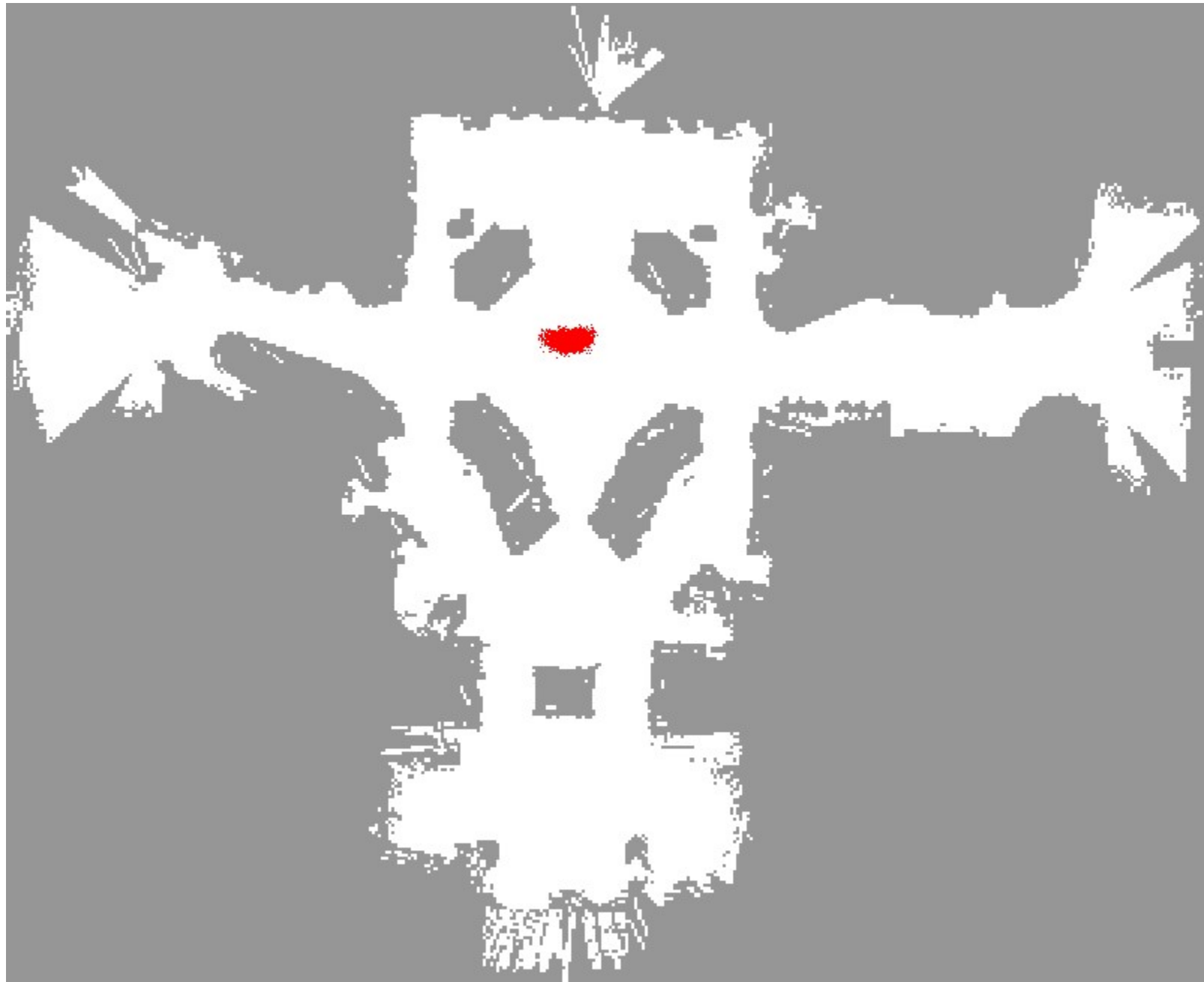


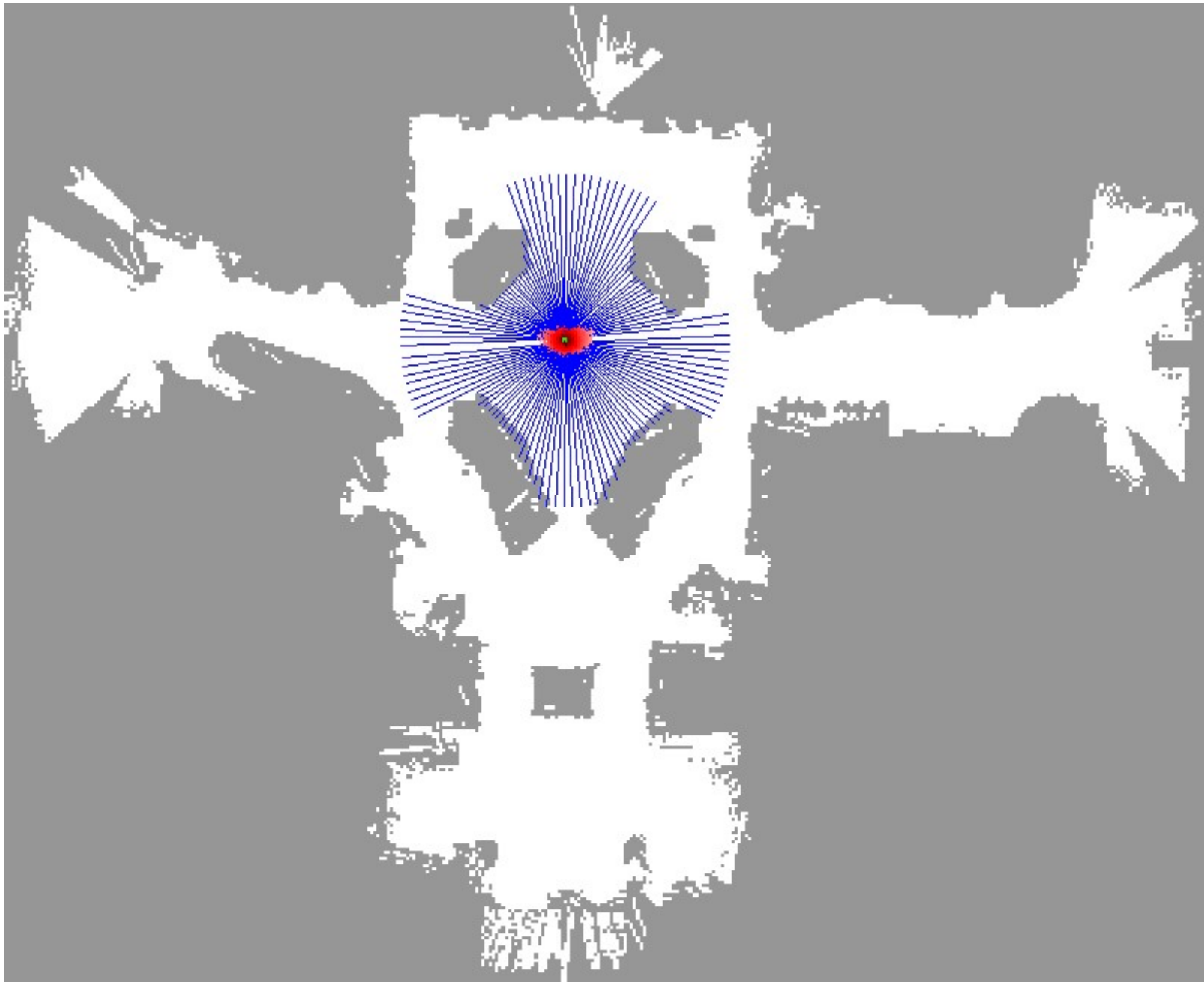


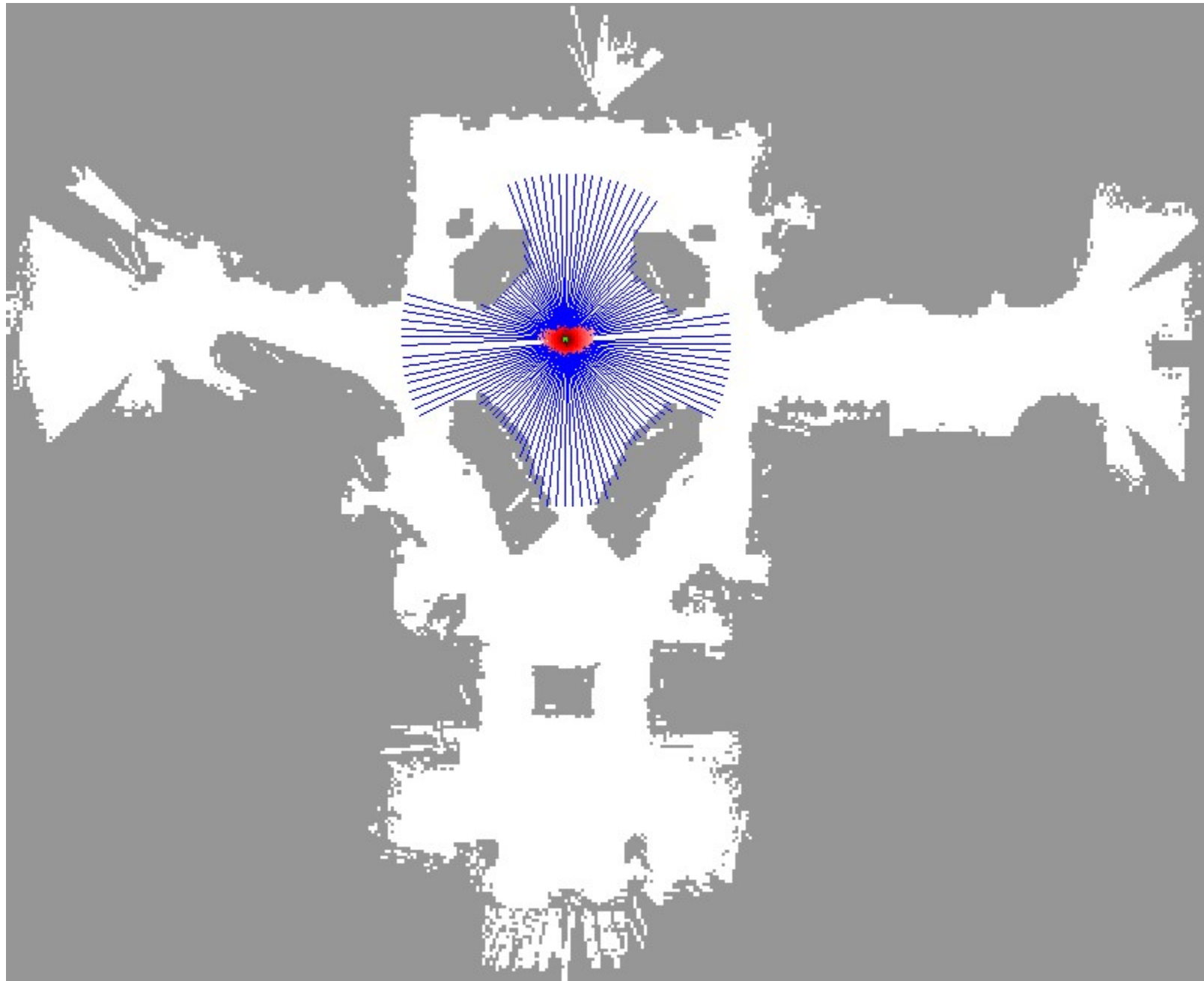




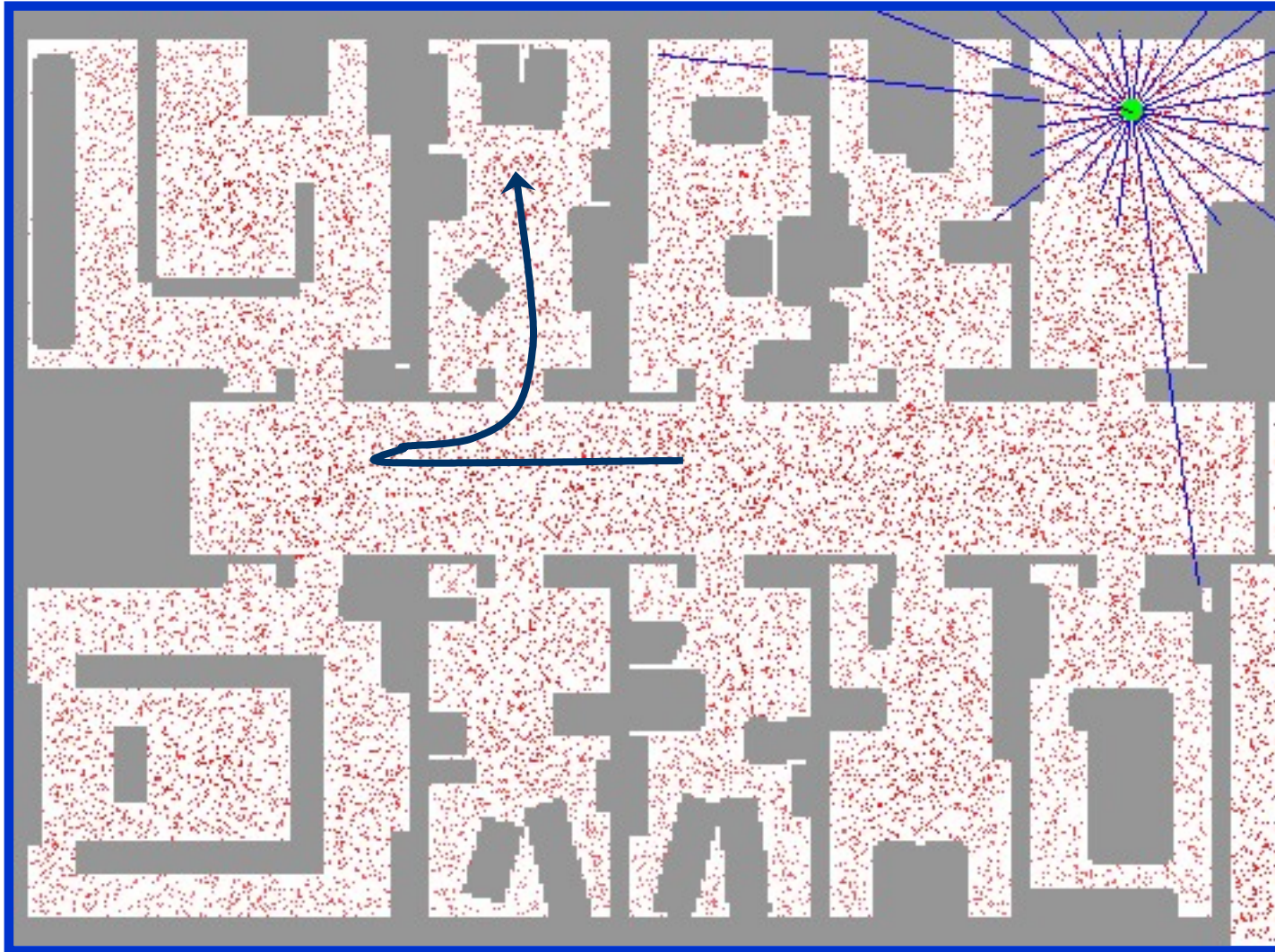




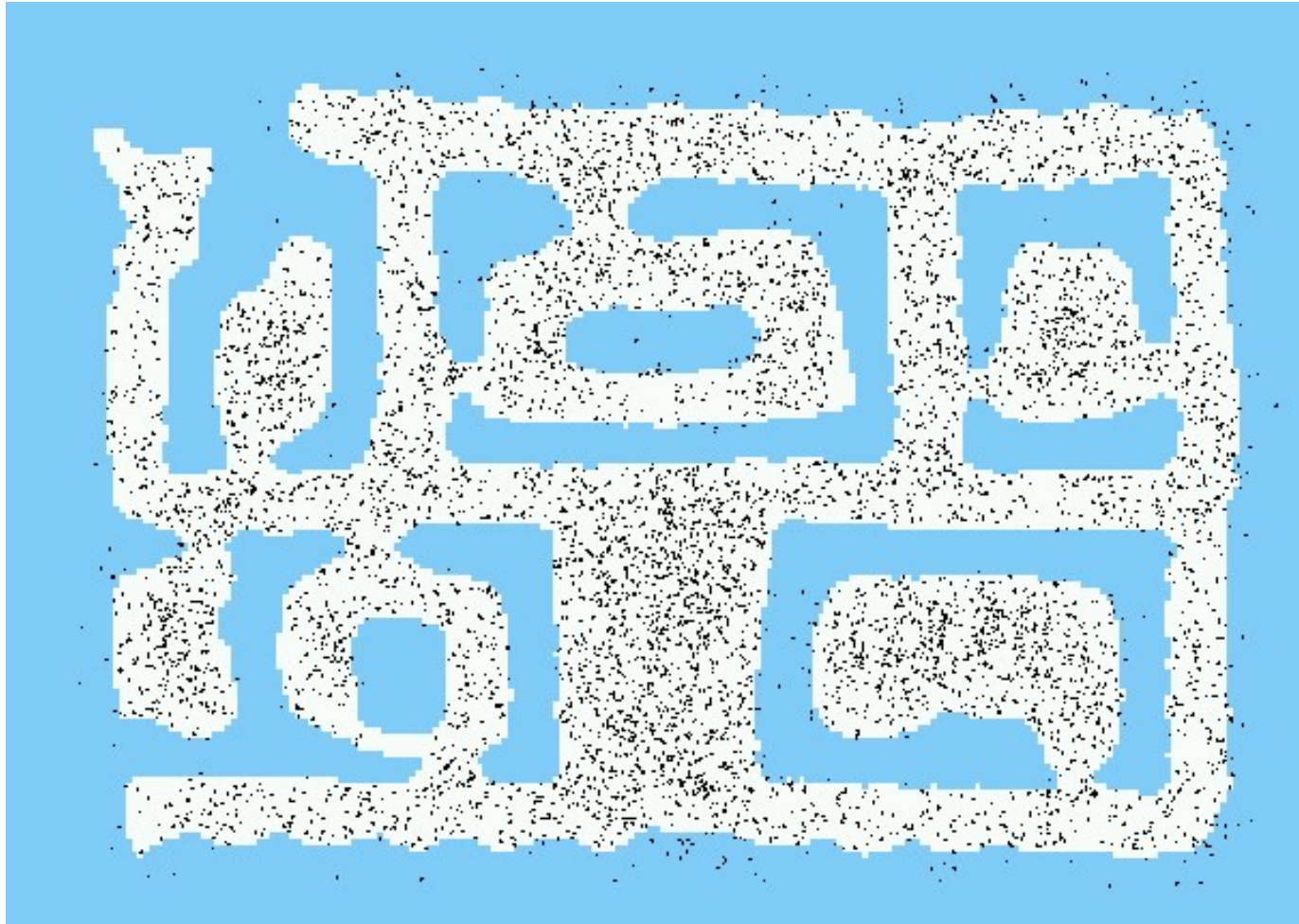




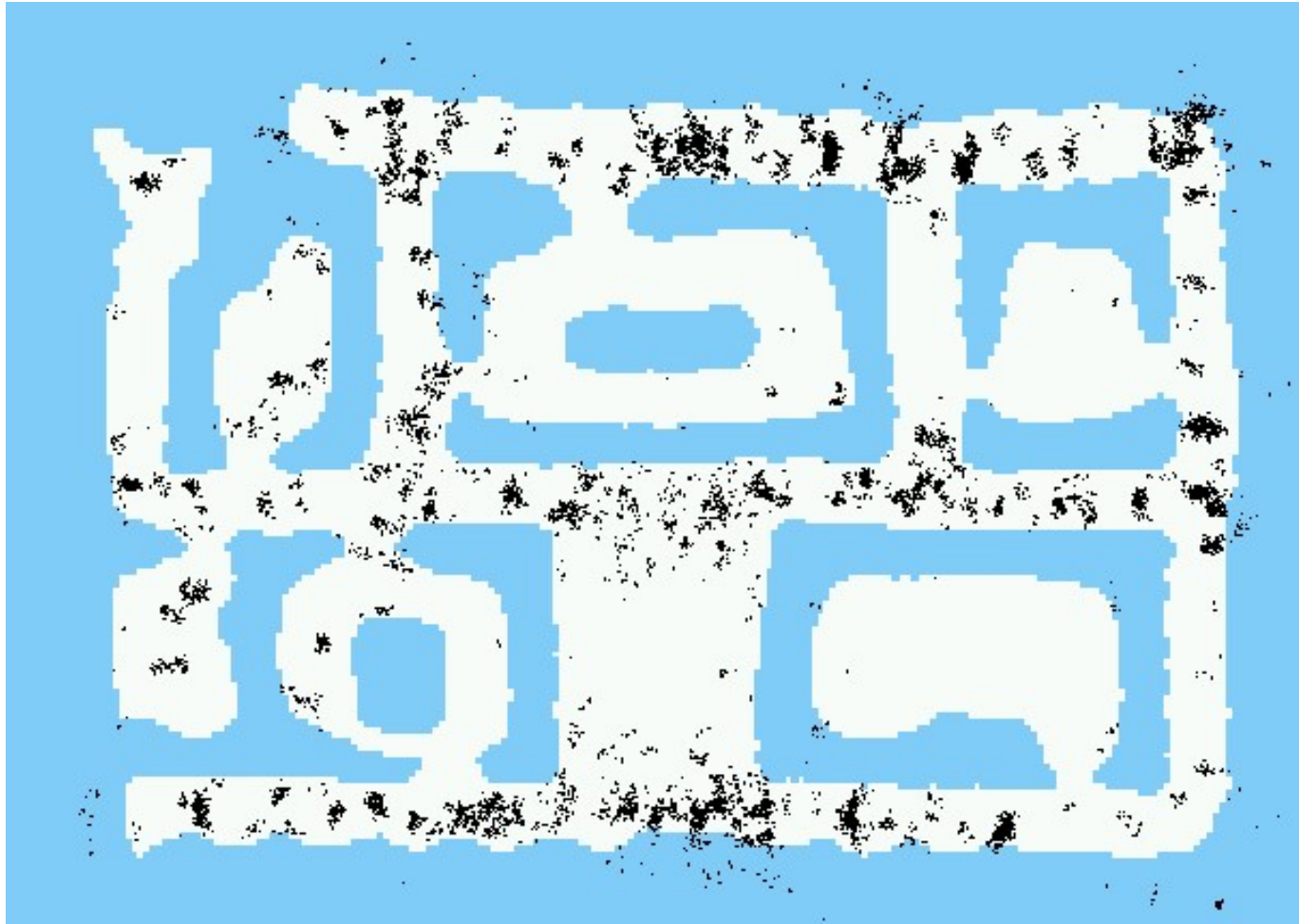
Localizzazione Sample based (sonar)



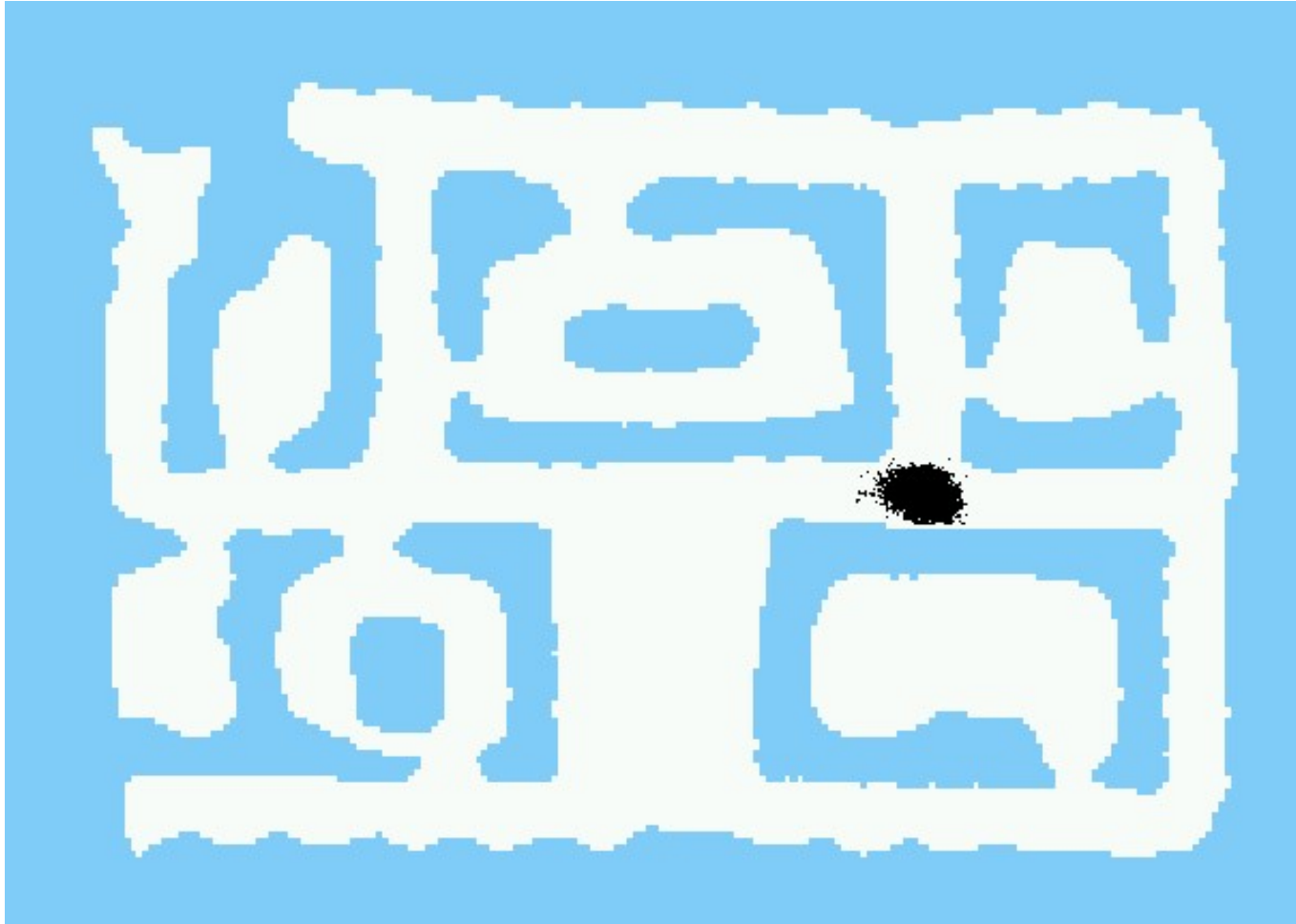
Initial Distribution



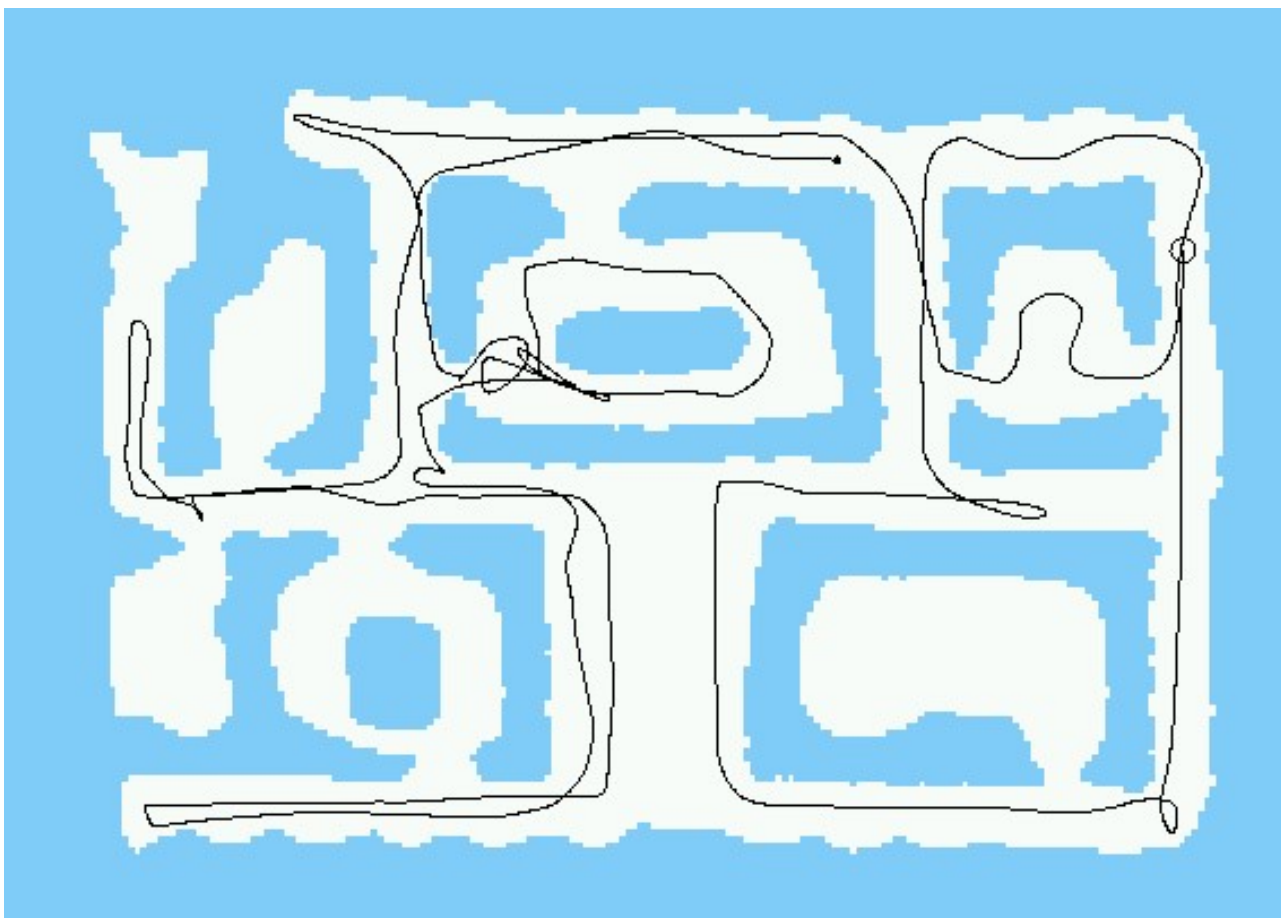
After Incorporating Ten Ultrasound Scans



After Incorporating 65 Ultrasound Scans



Percorso stimato



Limiti

- L'approccio funziona per
 - tracciare la posa del robot mobile
 - Localizzazione globale.
- Errori di localizzazione (i.e., kidnapped robot)?

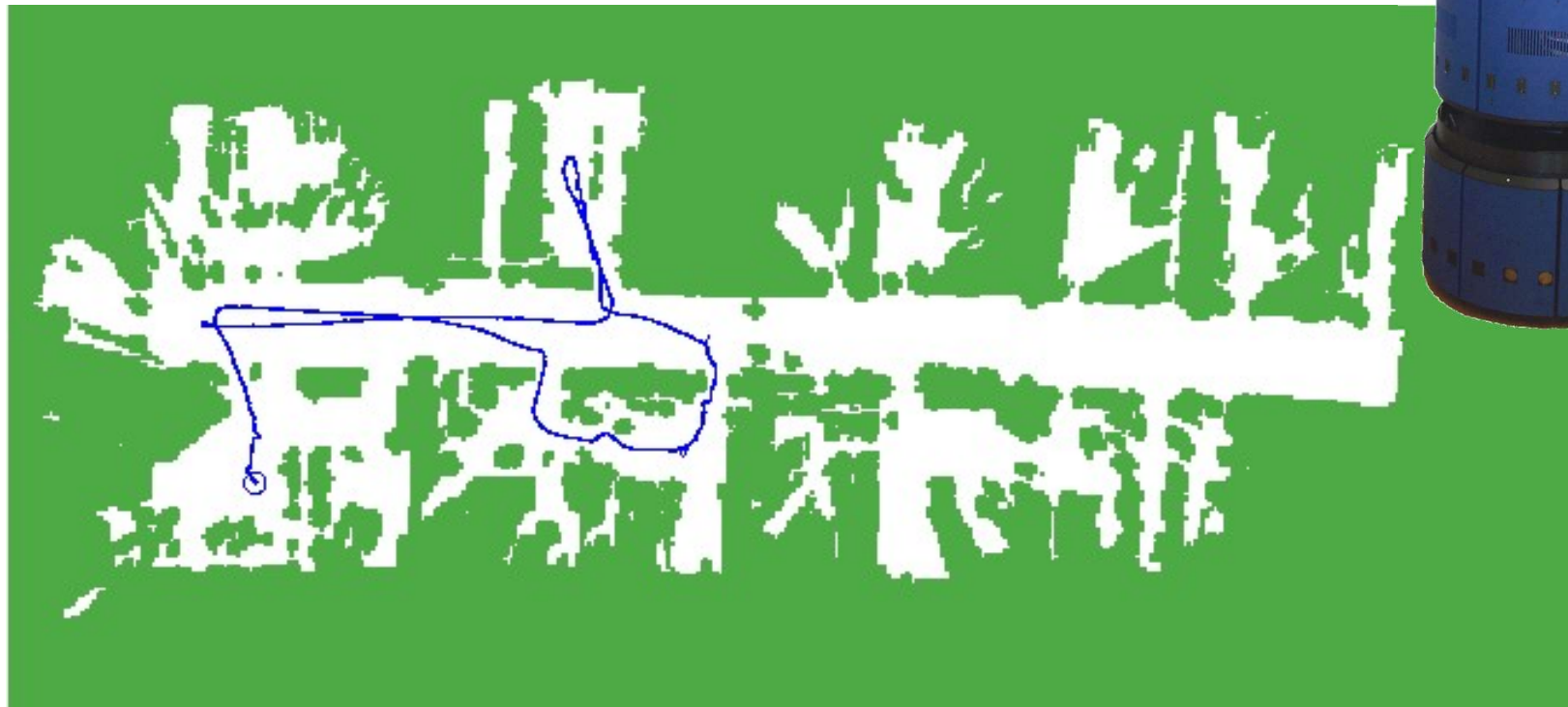
Approcci

- Inserire campioni random (robot può essere teletrasportato in ogni punto).
- Inserire campioni random proporzionali al likelihood medio delle particelle (il robot è teletrasportato con più probabilità quando il likelihood delle osservazioni decade).

Random Samples Localizzazione Vision-Based

936 Images, 4MB, .6secs/image

Trajectory of the robot:



Odometry Information



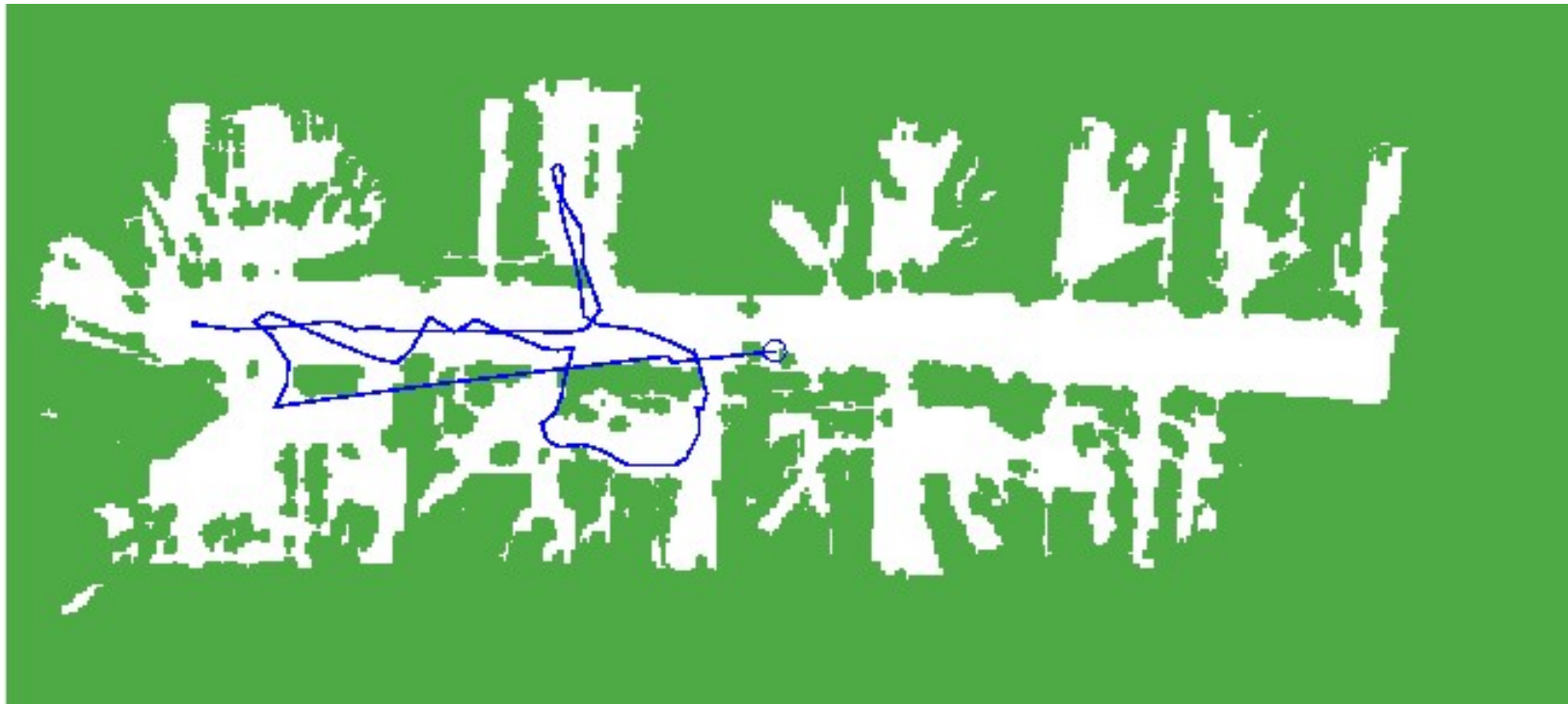
Traiettoria

Position tracking:

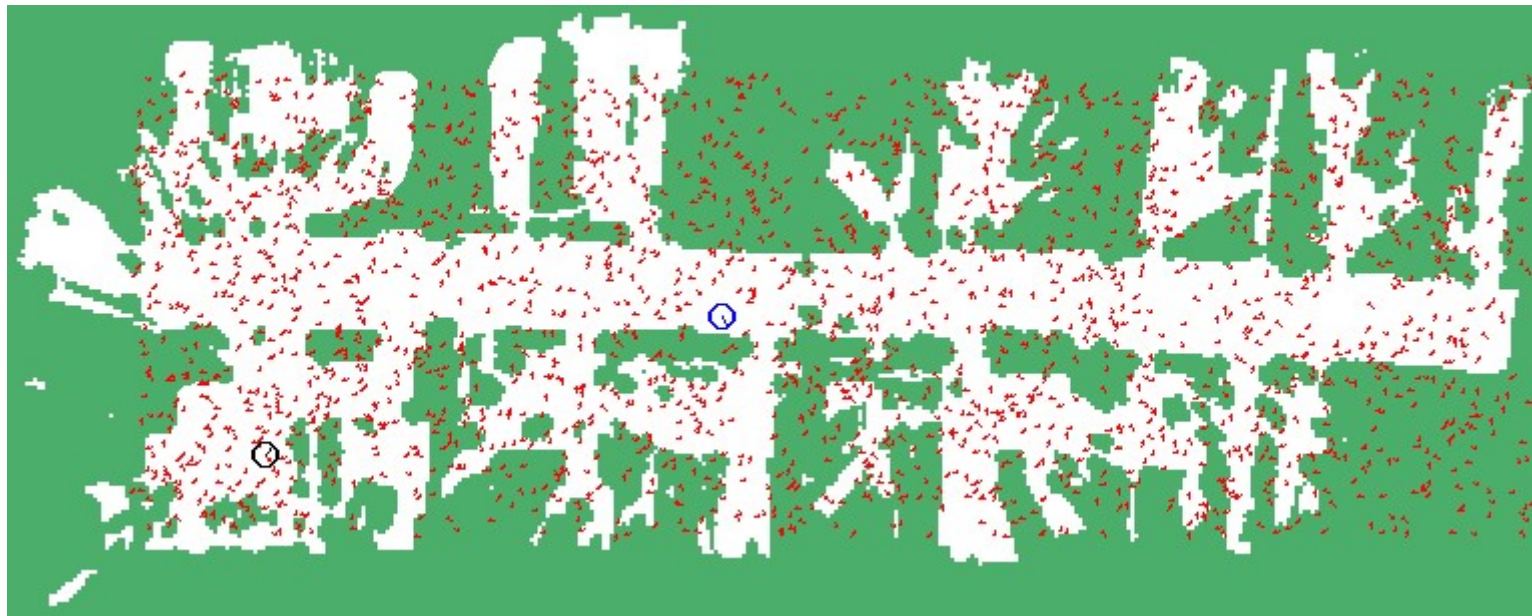


Resulting Trajectories

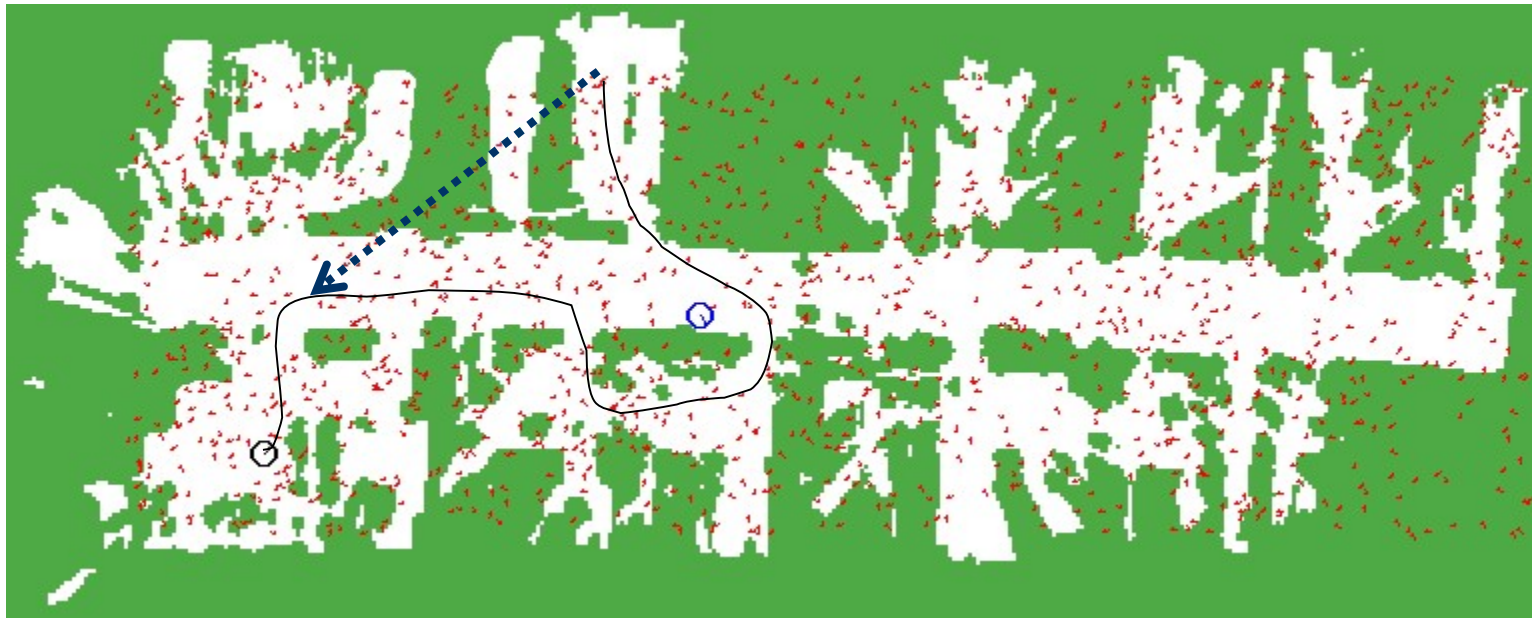
Global localization:



Global Localization



Kidnapping the Robot



Sommario

- Particle filter implementazione del recursive Bayesian filter
- Rappresentano il post con un insieme di campioni pesati.
- Per la localizzazione le particelle sono propagate secondo il motion model.
- Pesate secondo il likelihood delle osservazioni.
- Nel passo di re-sampling si estraggono nuove particelle proporzionali al likelihood dell'osservazione.