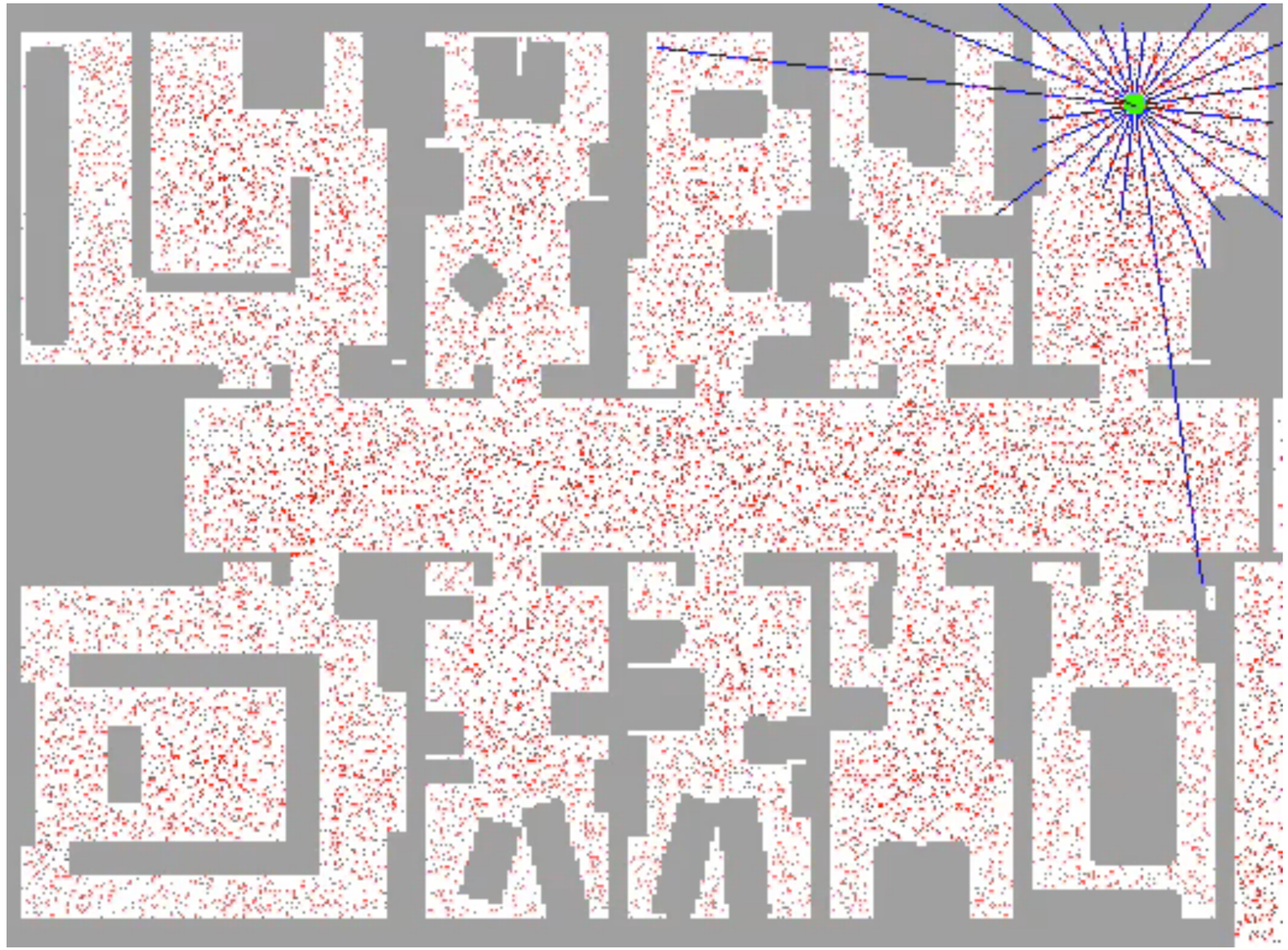
Lecture 20
Mobile Robotics - IV - Particle Filter

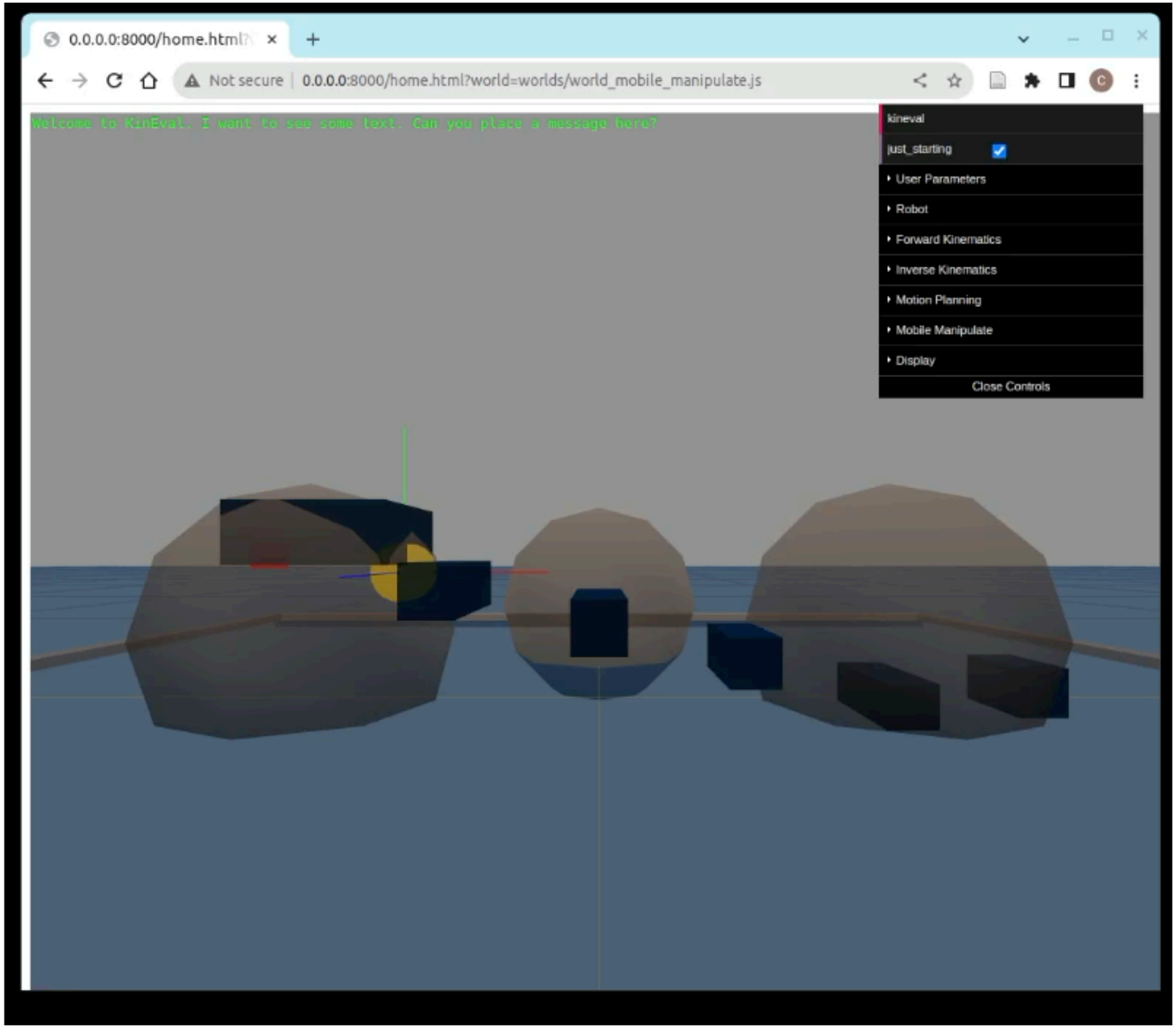


Course logistics

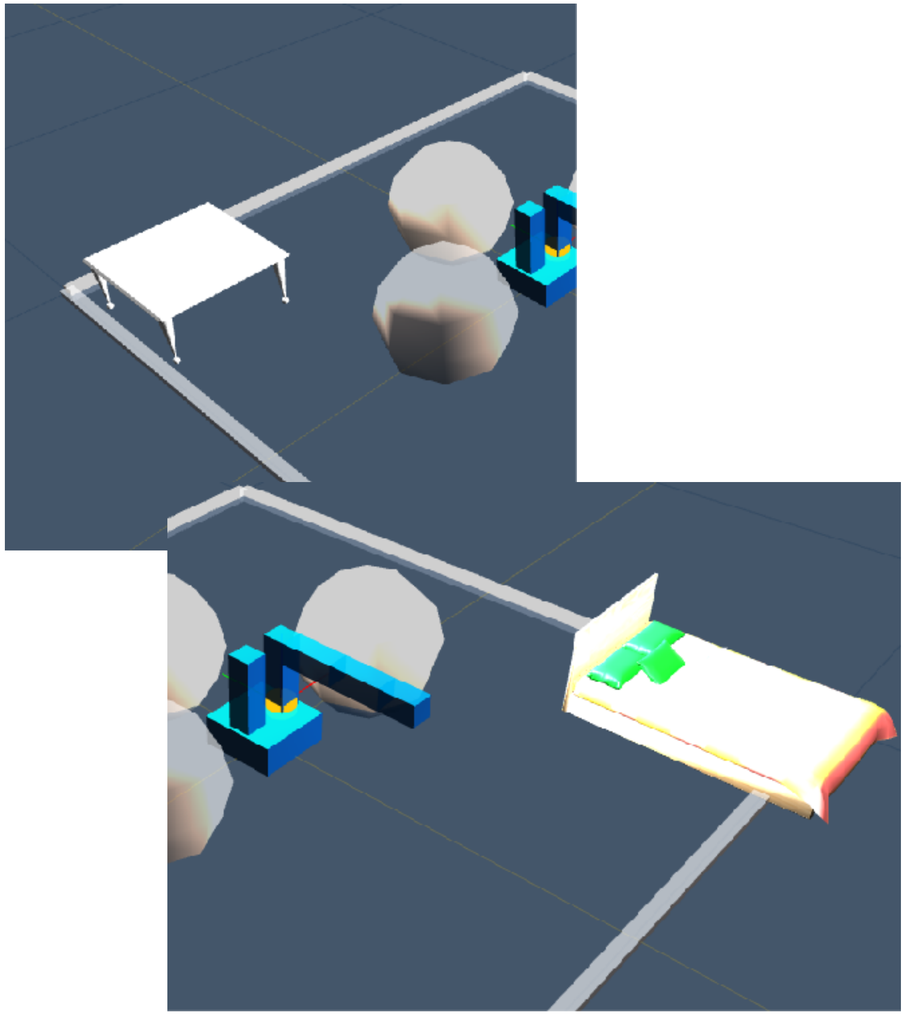
- **Quiz 18-19 was posted yesterday and was due before today's lecture**
- Project 4 is posted on 10/30 and will be due **11/15 (today)**
- Project 5 will be posted today
  - Start early!

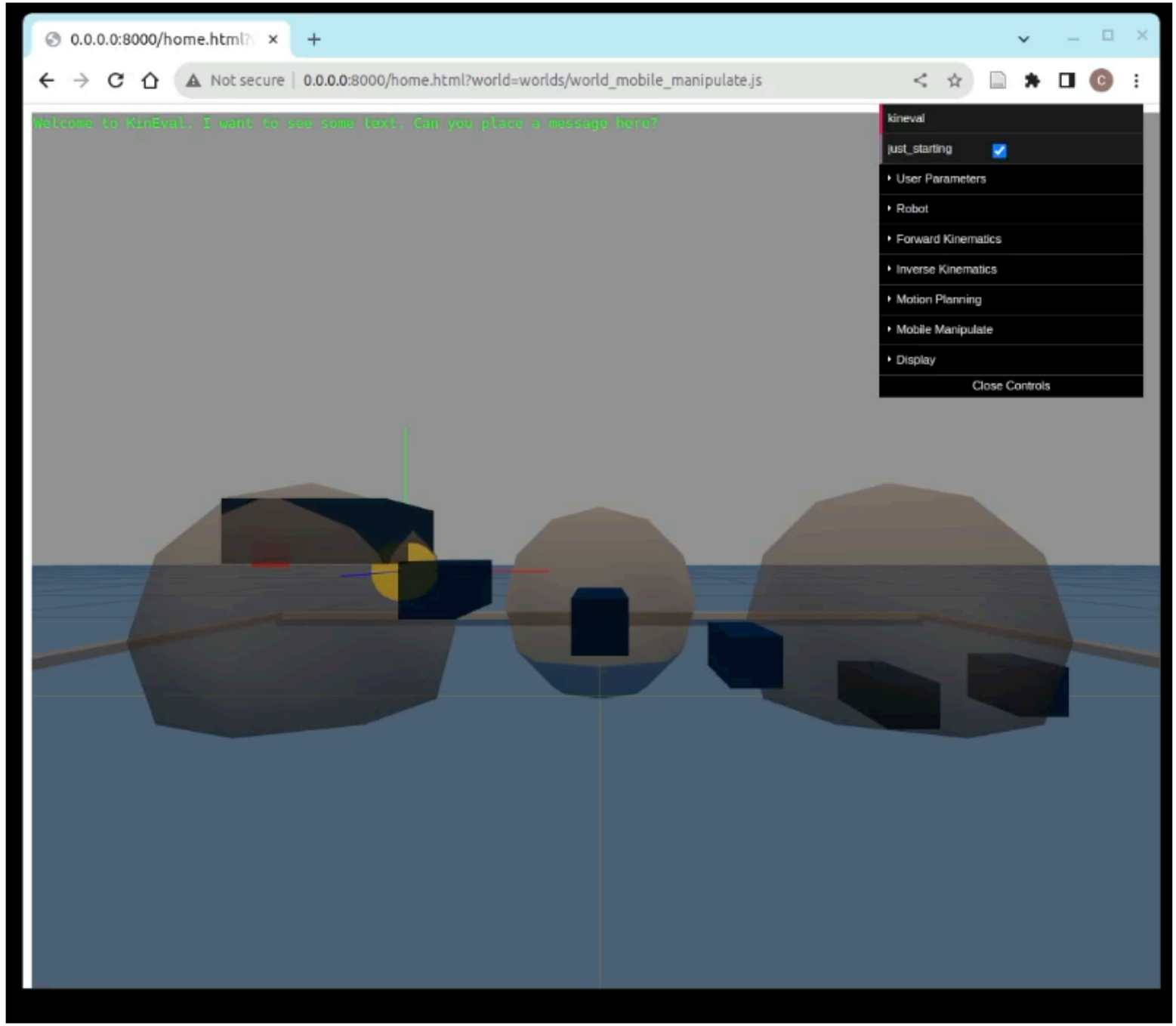
Project-5

- RRT-Connect
- FSM
- IK

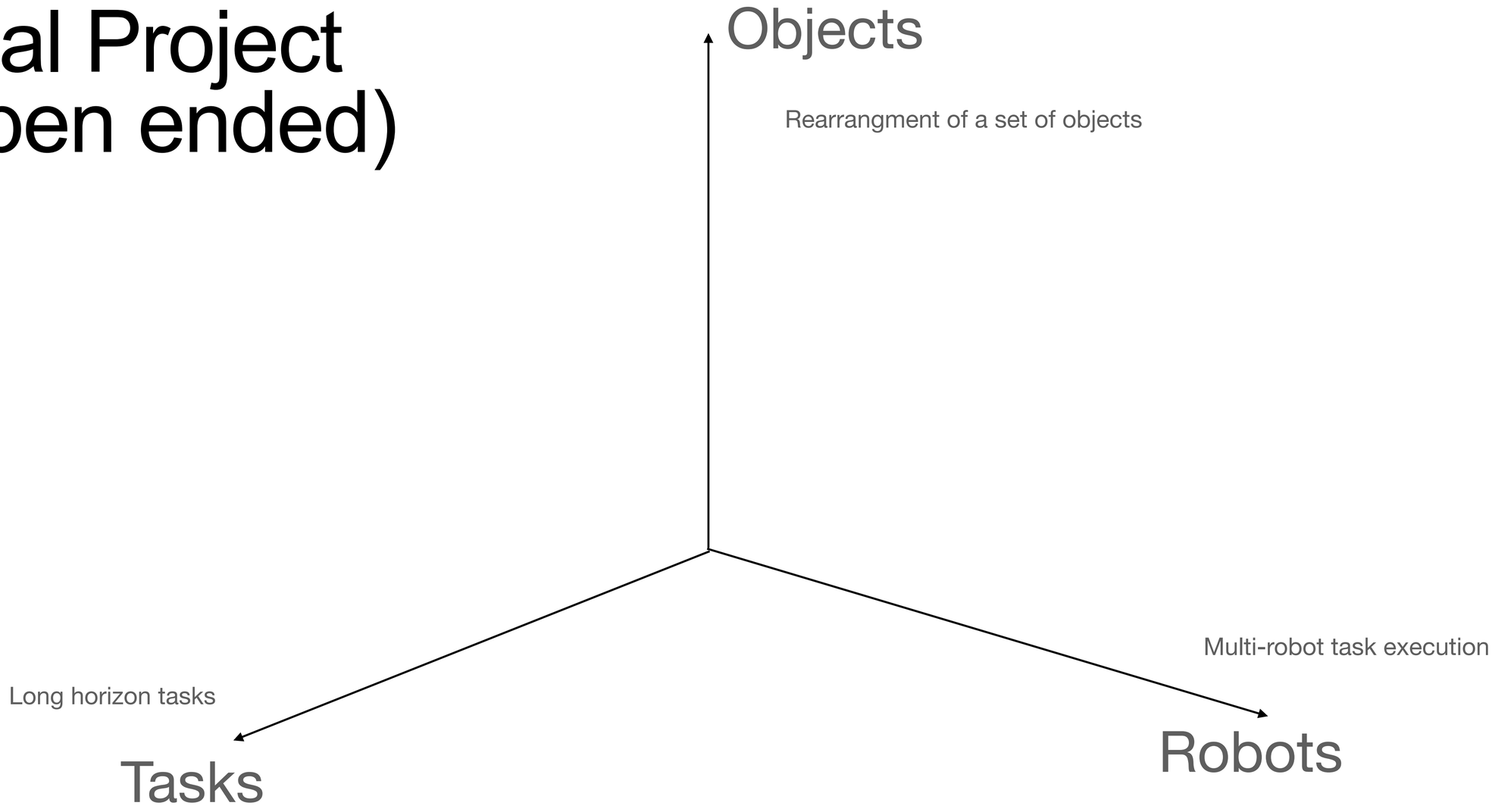


Final Project (Open ended)





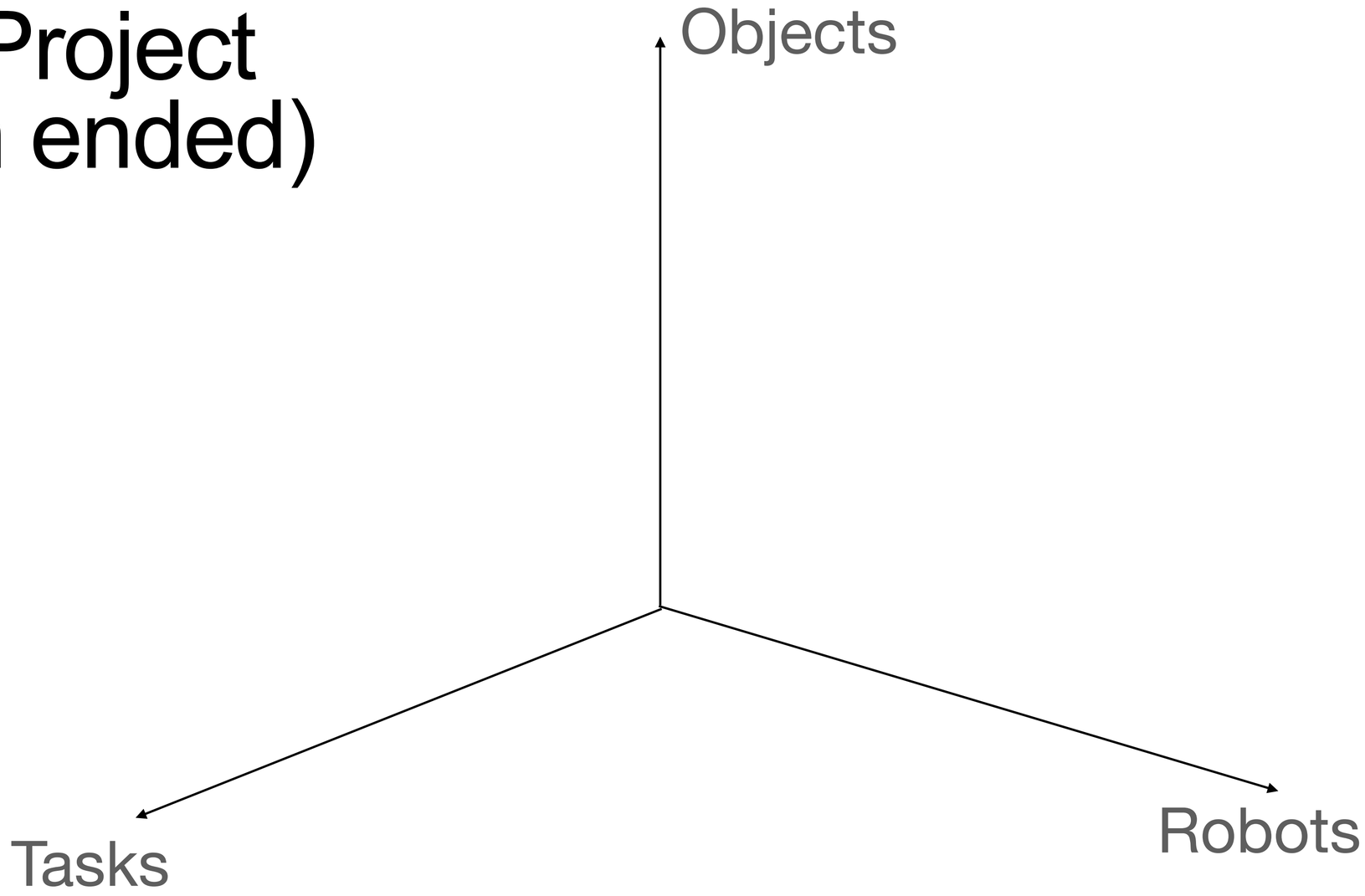
Final Project (Open ended)



Final Project (Open ended)
For inspiration!

Yang, Zhutian, Caelan Reed Garrett, Tomás Lozano-Pérez, Leslie Kaelbling, and Dieter Fox. "Sequence-based plan feasibility prediction for efficient task and motion planning." *arXiv preprint arXiv:2211.01576* (2022).

Final Project (Open ended)



Continuing previous Lecture
KF and EKF

Discrete Kalman Filter

Estimates the state x of a discrete-time controlled process that is governed by the linear stochastic difference equation

$$x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t$$

with a measurement

$$z_t = C_t x_t + \delta_t$$

Components of a Kalman Filter

A_t Matrix (nxn) that describes how the state evolves from t-1 to t without controls or noise.

B_t Matrix (nxl) that describes how the control u_t changes the state from t-1 to t.

C_t Matrix (kxn) that describes how to map the state x_t to an observation z_t.

ε_t δ_t Random variables representing the process and measurement noise that are assumed to be independent and normally distributed with covariance R_t and Q_t respectively.

Kalman Filter Algorithm

1. Algorithm **Kalman_filter**$(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t)$:

2. Prediction:
3. $\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t$
4. $\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t$

5. Correction:
6. $K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$
7. $\mu_t = \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t)$
8. $\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t$

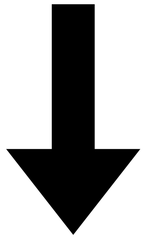
9. Return μ_t, Σ_t

Slide borrowed from Dieter Fox

Non-linear Dynamic Systems

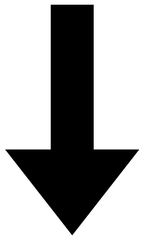
- Most realistic problems involve nonlinear functions

$$x_t = A_t x_{t-1} + B_t u_t + \epsilon_t$$



$$x_t = g(u_t, x_{t-1}) + \epsilon_t$$

$$z_t = C_t x_t + \delta_t$$



$$z_t = h(x_t) + \delta_t$$

Linearization

$$x_t = g(u_t, x_{t-1}) + \varepsilon_t$$

$$g(u_t, x_{t-1}) \approx g(u_t, \mu_{t-1}) + \underbrace{g'(u_t, \mu_{t-1})}_{=: G_t} (x_{t-1} - \mu_{t-1})$$

$$= g(u_t, \mu_{t-1}) + G_t (x_{t-1} - \mu_{t-1})$$

$$z_t = h(x_t) + \delta_t$$

$$h(x_t) \approx h(\bar{\mu}_t) + \underbrace{\frac{\partial h(\bar{\mu}_t)}{\partial x_t}}_{=: H_t} (x_t - \bar{\mu}_t)$$

Slide borrowed from Dieter Fox

EKF Algorithm

1. **Extended_Kalman_filter**$(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t)$:

2. Prediction:

3. $\bar{\mu}_t = g(u_t, \mu_{t-1})$ $\longleftarrow$ $\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t$

4. $\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + R_t$ $\longleftarrow$ $\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t$

5. Correction:

6. $K_t = \bar{\Sigma}_t H_t^T (H_t \bar{\Sigma}_t H_t^T + Q_t)^{-1}$ $\longleftarrow$ $K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$

7. $\mu_t = \bar{\mu}_t + K_t (z_t - h(\bar{\mu}_t))$ $\longleftarrow$ $\mu_t = \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t)$

8. $\Sigma_t = (I - K_t H_t) \bar{\Sigma}_t$ $\longleftarrow$ $\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t$

9. Return μ_t, Σ_t

$$H_t = \frac{\partial h(\bar{\mu}_t)}{\partial x_t} \qquad G_t = \frac{\partial g(u_t, \mu_{t-1})}{\partial x_{t-1}}$$

Slide borrowed from Dieter Fox

Localization

> "Using sensory information to locate the robot in its environment is the most fundamental problem to providing a mobile robot with autonomous capabilities." [Cox '91]

- **Given**
  - Map of the environment.
  - Sequence of sensor measurements.
- **Wanted**
  - Estimate of the robot's position.
- **Problem classes**
  - Position tracking
  - Global localization
  - Kidnapped robot problem (recovery)

Slide borrowed from Dieter Fox

EKF Summary

- Highly efficient: Polynomial in measurement dimensionality k and state dimensionality n:

$$O(k^{2.376} + n^2)$$

- Not optimal!
- Can diverge if nonlinearities are large!
- Works surprisingly well even when all assumptions are violated!

Particle Filter

A Bayesian Filter Implementation

Motivation

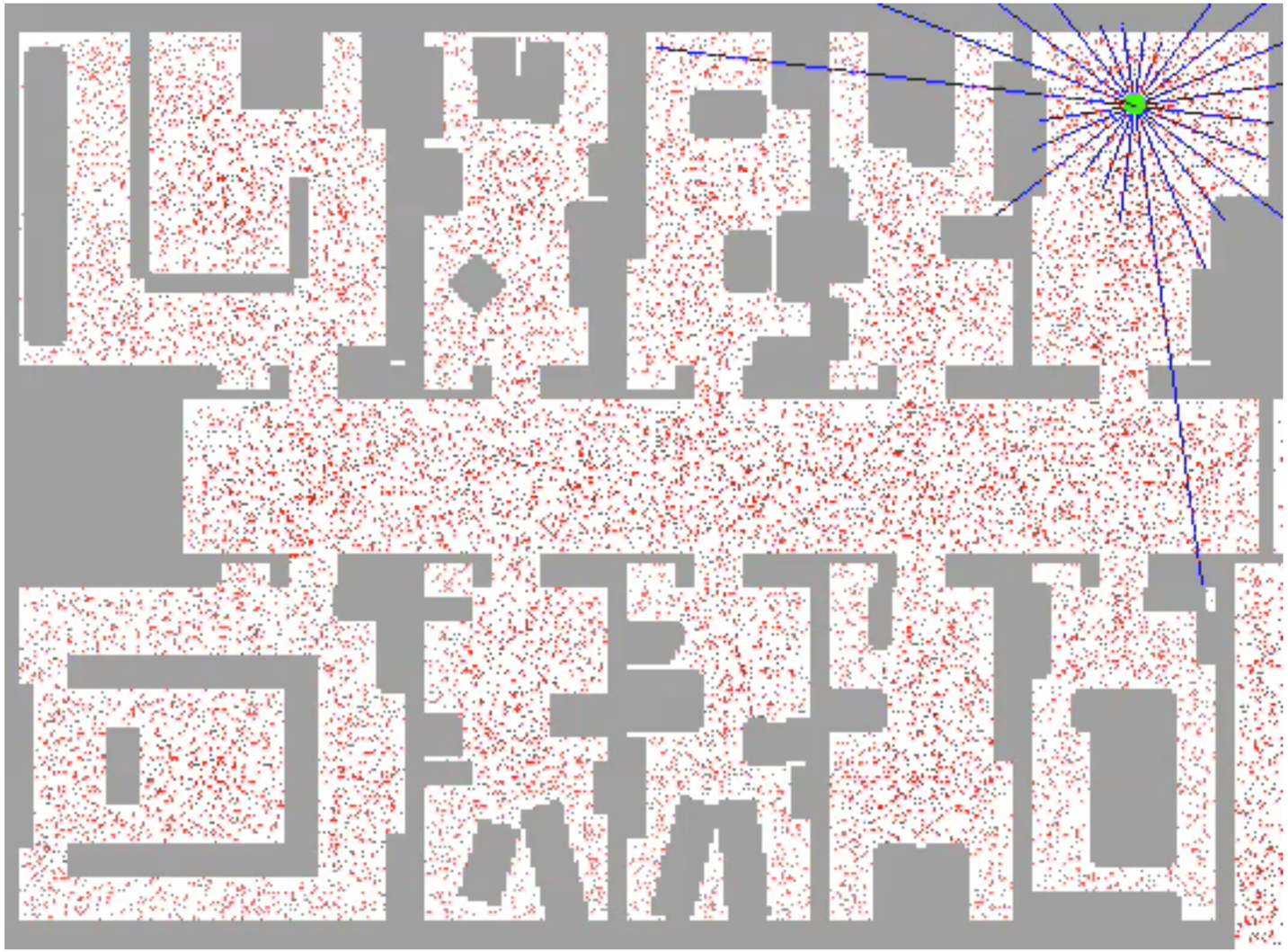
- So far, we discussed the
  - Kalman filter: Gaussian, linearization problems, multi-modal beliefs

- Particle filters are a way to **efficiently** represent **non-Gaussian distributions**

- Basic principle
  - Set of state hypotheses (“particles”)
  - Survival-of-the-fittest

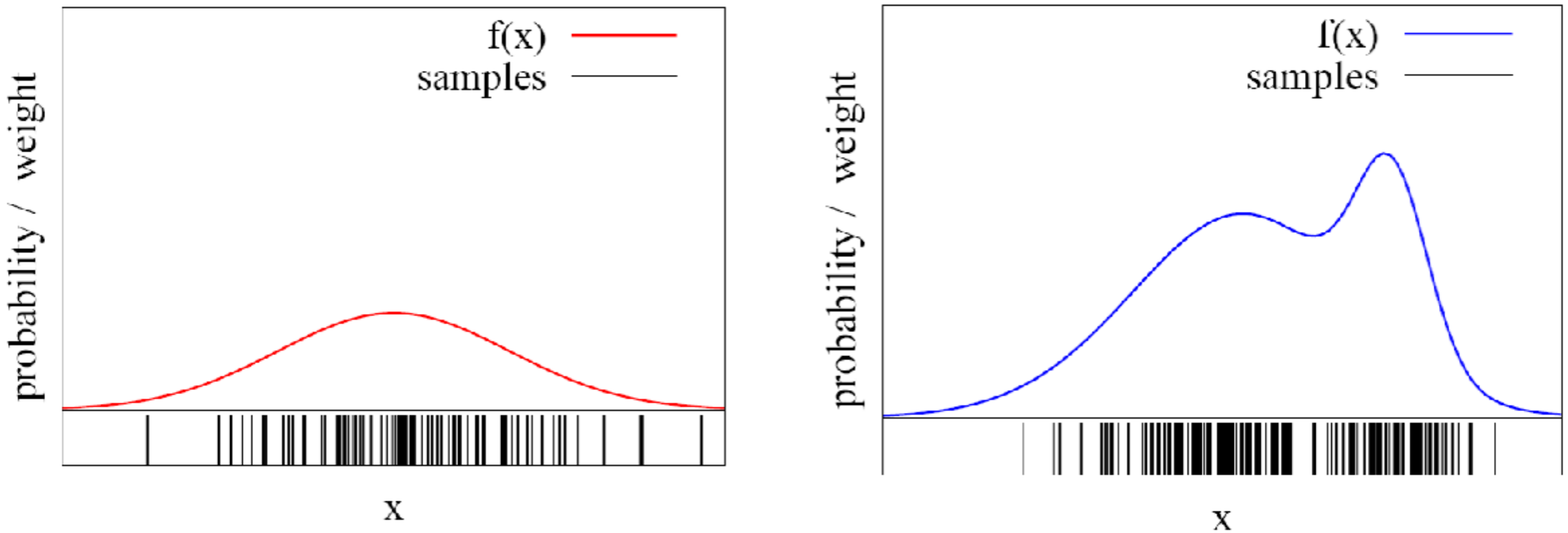
Slide borrowed from Dieter Fox

Sample-based Localization (sonar)



Density Approximation

- Particle sets can be used to approximate densities

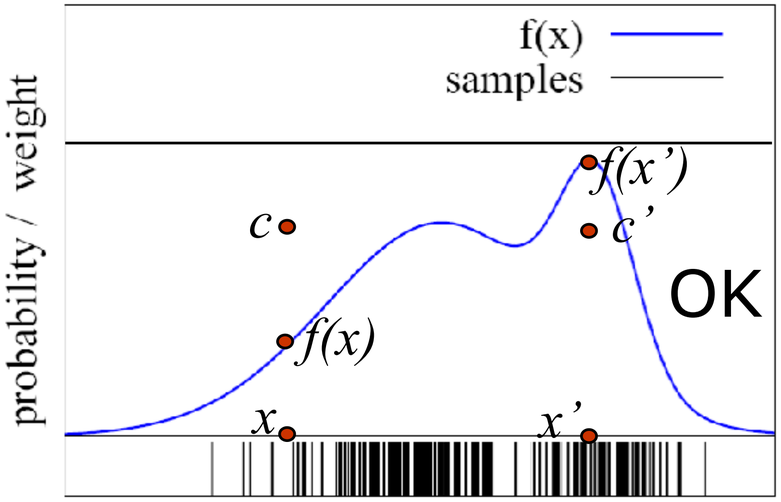


- The more particles fall into an interval, the higher the probability of that interval

- How to draw samples form a function/distribution?

Rejection Sampling

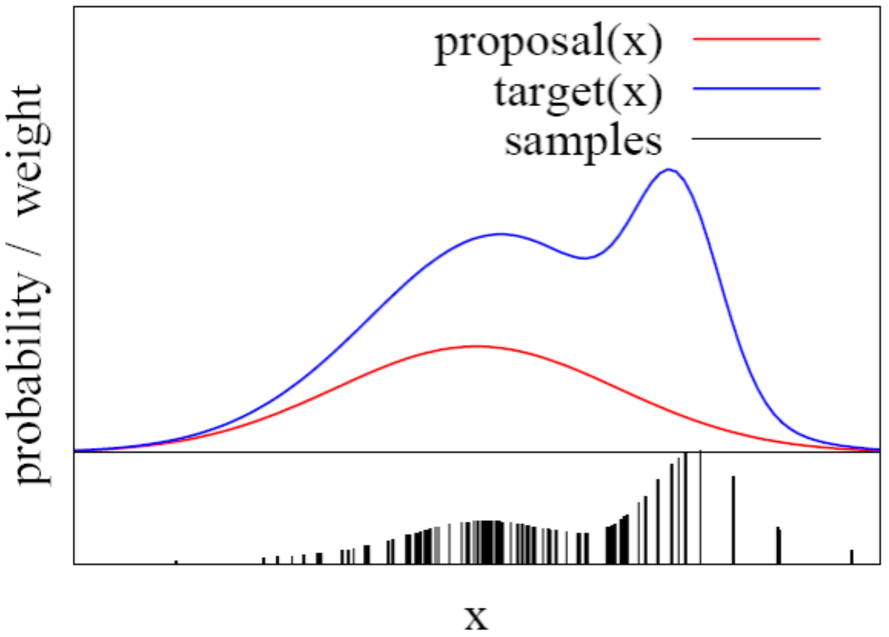
- Let us assume that $f(x)<=1$ for all x
- Sample x from a uniform distribution
- Sample c from [0,1]
- if $f(x) > c$ keep the sample
  otherwise reject the sampe



Slide borrowed from Dieter Fox

Importance Sampling Principle

- We can even use a different distribution g to generate samples from f
- By introducing an importance weight w, we can account for the "differences between g and f"
- $w = f / g$
- f is often called target
- g is often called proposal



Slide borrowed from Dieter Fox

Particle Filter for State estimation

- Non-parametric approach
- Recursive Bayes Filter
- Models the distribution by samples
- **Prediction:** draw from the proposal g
- **Correction:** weighting by the ratio of the target f and the proposal g

The more samples we use, the better is the estimate

Particle Filter Algorithm

1. Sample the particles using the proposal distribution.

$$x_t^{[j]} \sim \text{proposal}(x_t | \dots)$$

2. Compute the importance weights

$$w_t^{[j]} = \frac{\text{target}(x_t^{[j]})}{\text{proposal}(x_t^{[j]})}$$

3. Resampling: Draw samples i with propobability $w_t^{[i]}$ and repeat J times

Particle Filter Algorithm

Particle_filter$(\mathcal{X}_{t-1}, u_t, z_t)$**:**

1: $\bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset$

2: *for* $j = 1$ *to* J *do*

3: *sample* $x_t^{[j]} \sim \pi(x_t)$

4: $w_t^{[j]} = \frac{p(x_t^{[j]})}{\pi(x_t^{[j]})}$

5: $\bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \langle x_t^{[j]}, w_t^{[j]} \rangle$

6: *endfor*

7: *for* $j = 1$ *to* J *do*

8: *draw* $i \in 1, \ldots, J$ *with probability* $\propto w_t^{[i]}$

9: *add* $x_t^{[i]}$ *to* $\mathcal{X}_t$

10: *endfor*

11: *return* $\mathcal{X}_t$

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 $Bel(x_{t-1})$

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

Importance factor for x^i_t:

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

Particle Filter

Particle_filter$(\mathcal{X}_{t-1}, u_t, z_t)$**:**

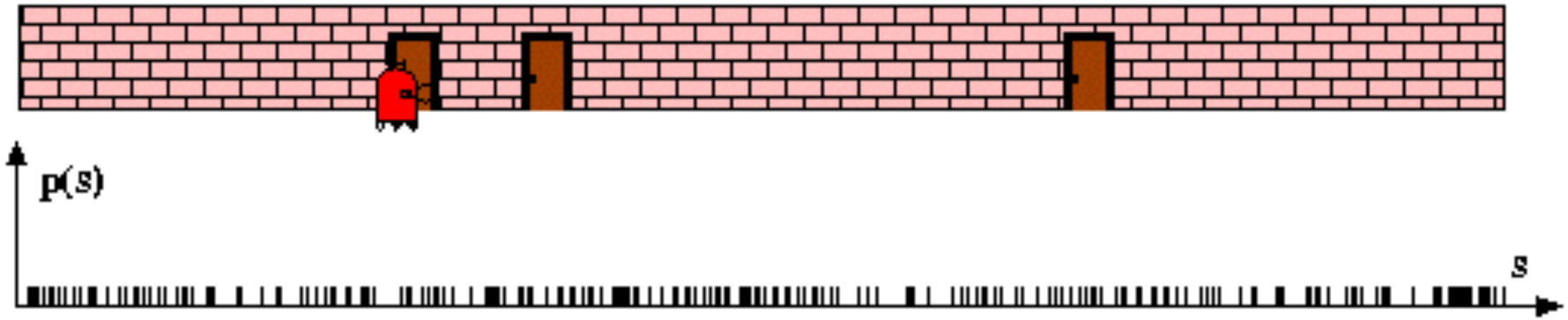
1: $\bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset$
2: *for* $j = 1$ *to* J *do*
3: *sample* $x_t^{[j]} \sim \pi(x_t)$
4: $w_t^{[j]} = \frac{p(x_t^{[j]})}{\pi(x_t^{[j]})}$
5: $\bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \langle x_t^{[j]}, w_t^{[j]} \rangle$
6: *endfor*
7: *for* $j = 1$ *to* J *do*
8: *draw* $i \in 1, \ldots, J$ *with probability* $\propto w_t^{[i]}$
9: *add* $x_t^{[i]}$ *to* $\mathcal{X}_t$
10: *endfor*
11: *return* $\mathcal{X}_t$

Particle Filter for Localization

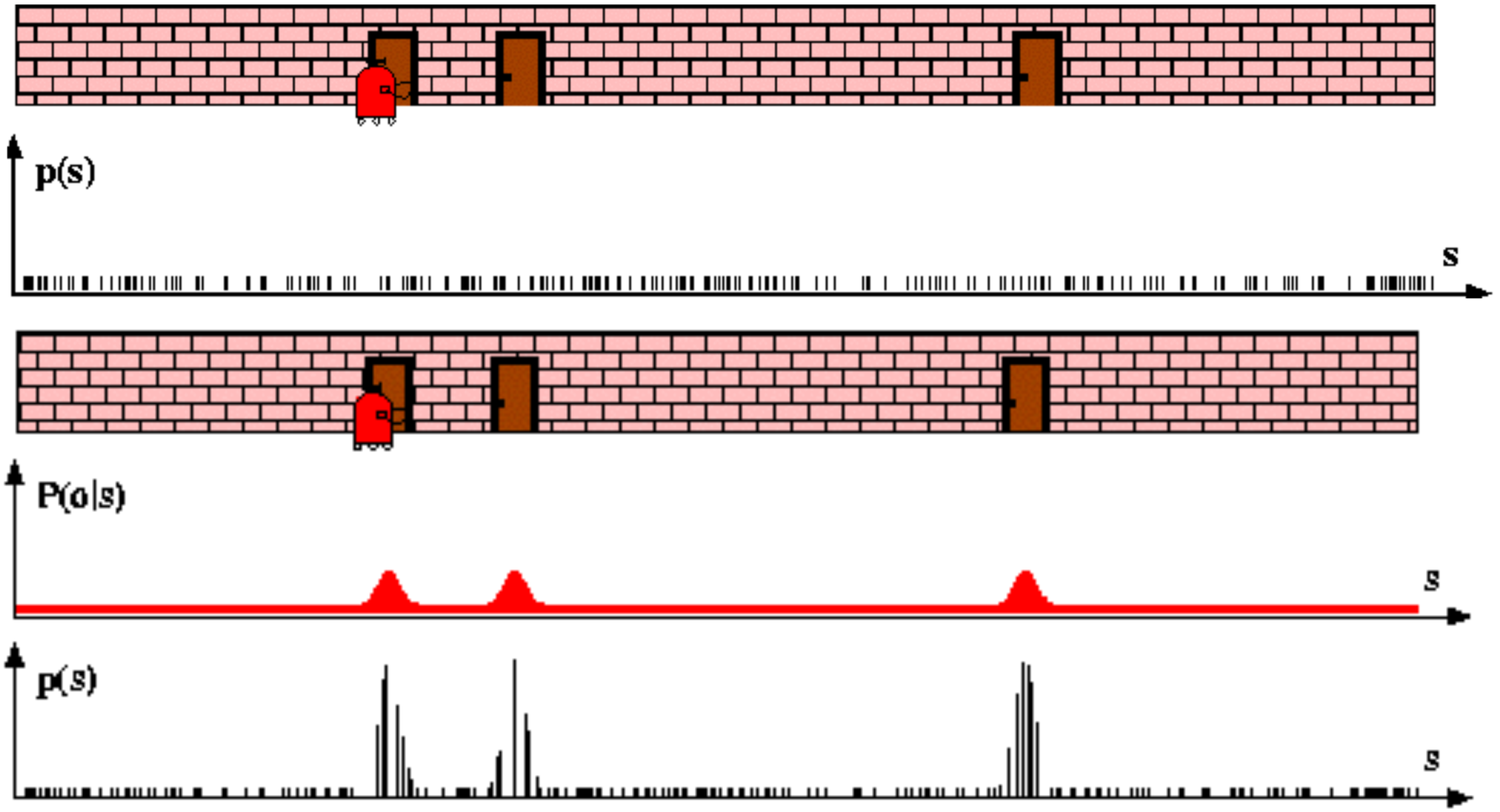
Particle_filter$(\mathcal{X}_{t-1}, u_t, z_t)$**:**

1: $\bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset$
2: *for* $j = 1$ *to* J *do*
3: *sample* $x_t^{[j]} \sim p(x_t \mid u_t, x_{t-1}^{[j]})$
4: $w_t^{[j]} = p(z_t \mid x_t^{[j]})$
5: $\bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \langle x_t^{[j]}, w_t^{[j]} \rangle$
6: *endfor*
7: *for* $j = 1$ *to* J *do*
8: *draw* $i \in 1, \ldots, J$ *with probability* $\propto w_t^{[i]}$
9: *add* $x_t^{[i]}$ *to* $\mathcal{X}_t$
10: *endfor*
11: *return* $\mathcal{X}_t$

Particle Filters

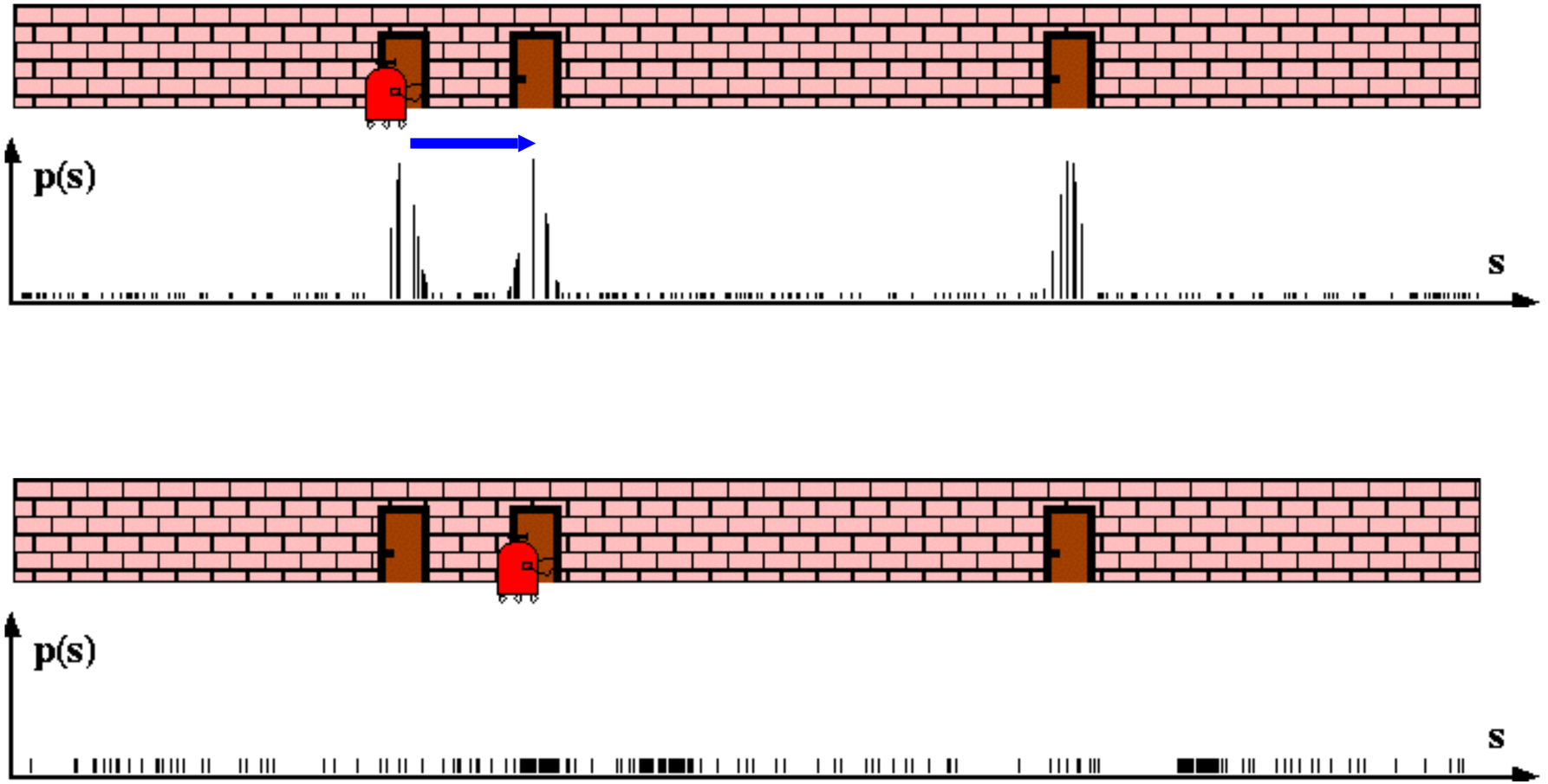


Sensor Information: Importance Sampling



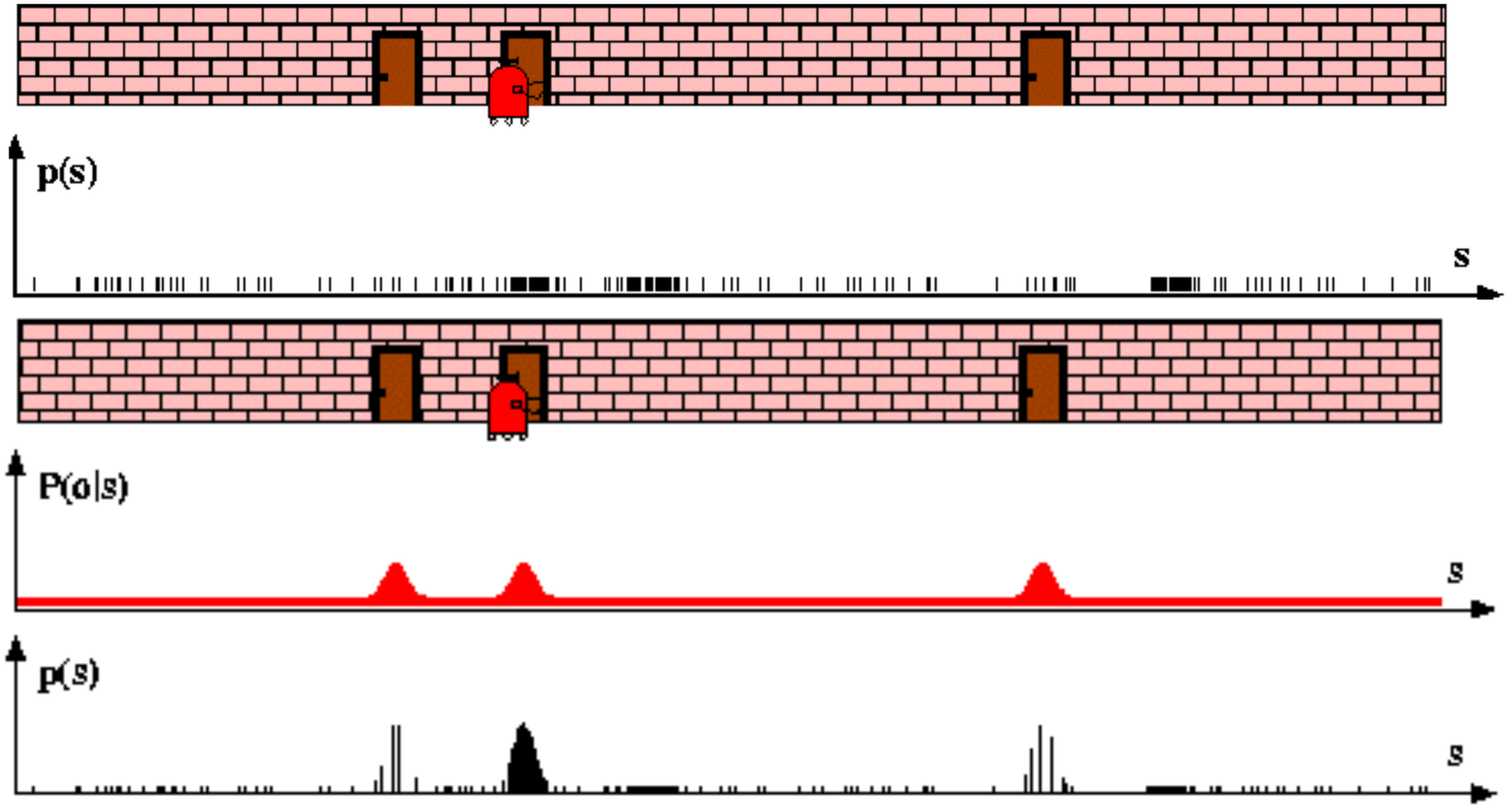
Slide borrowed from Dieter Fox

Robot Motion



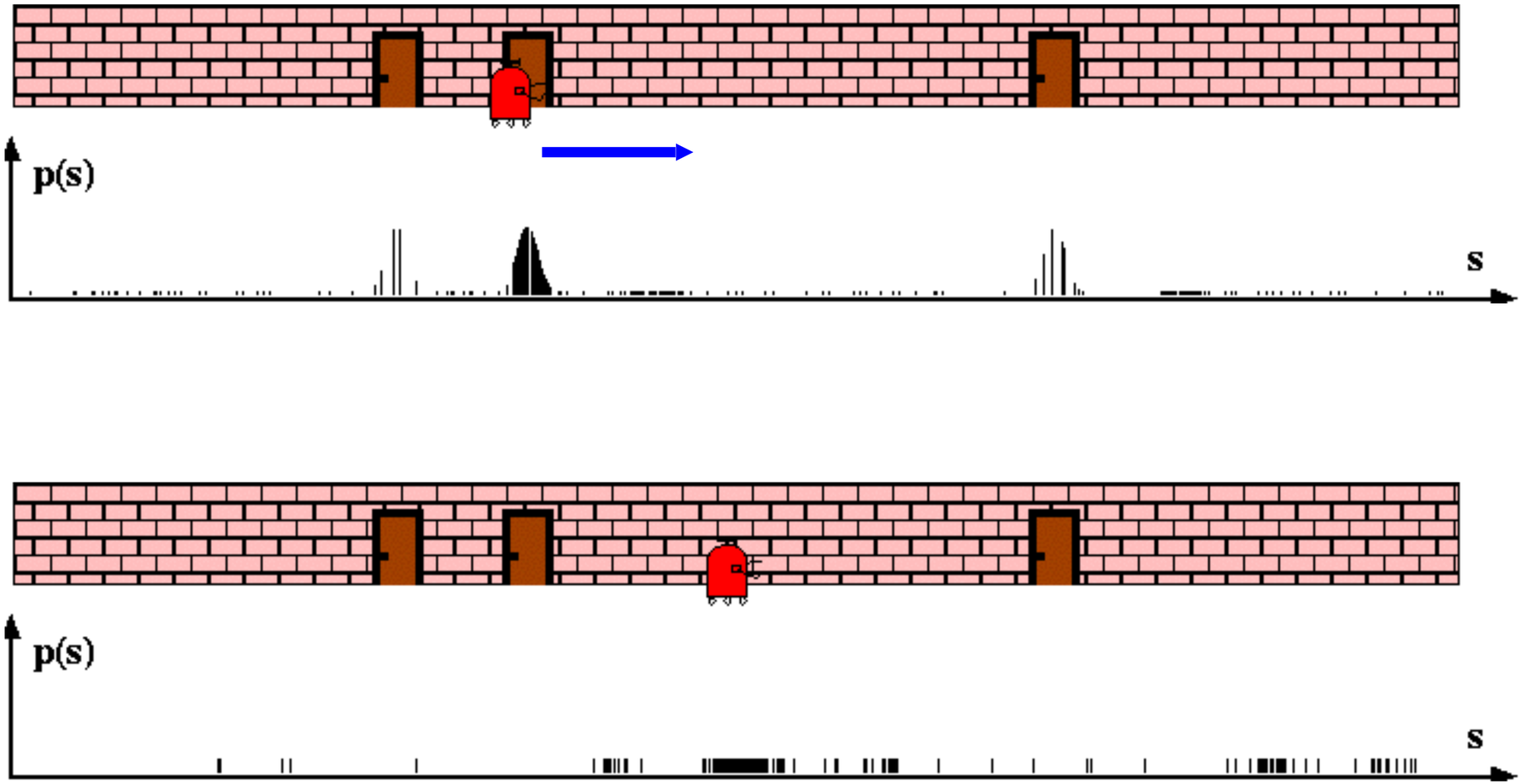
Slide borrowed from Dieter Fox

Sensor Information: Importance Sampling

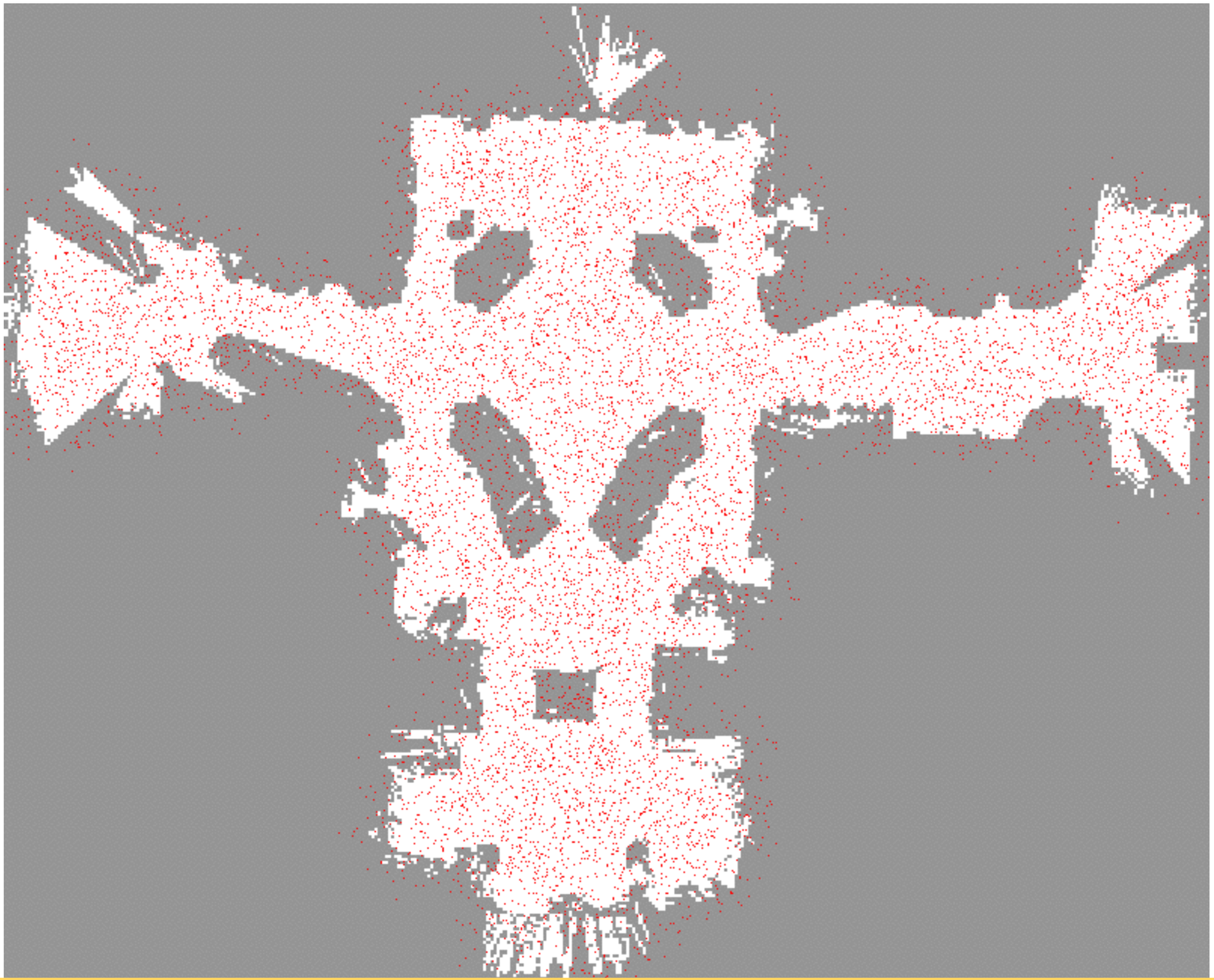


Slide borrowed from Dieter Fox

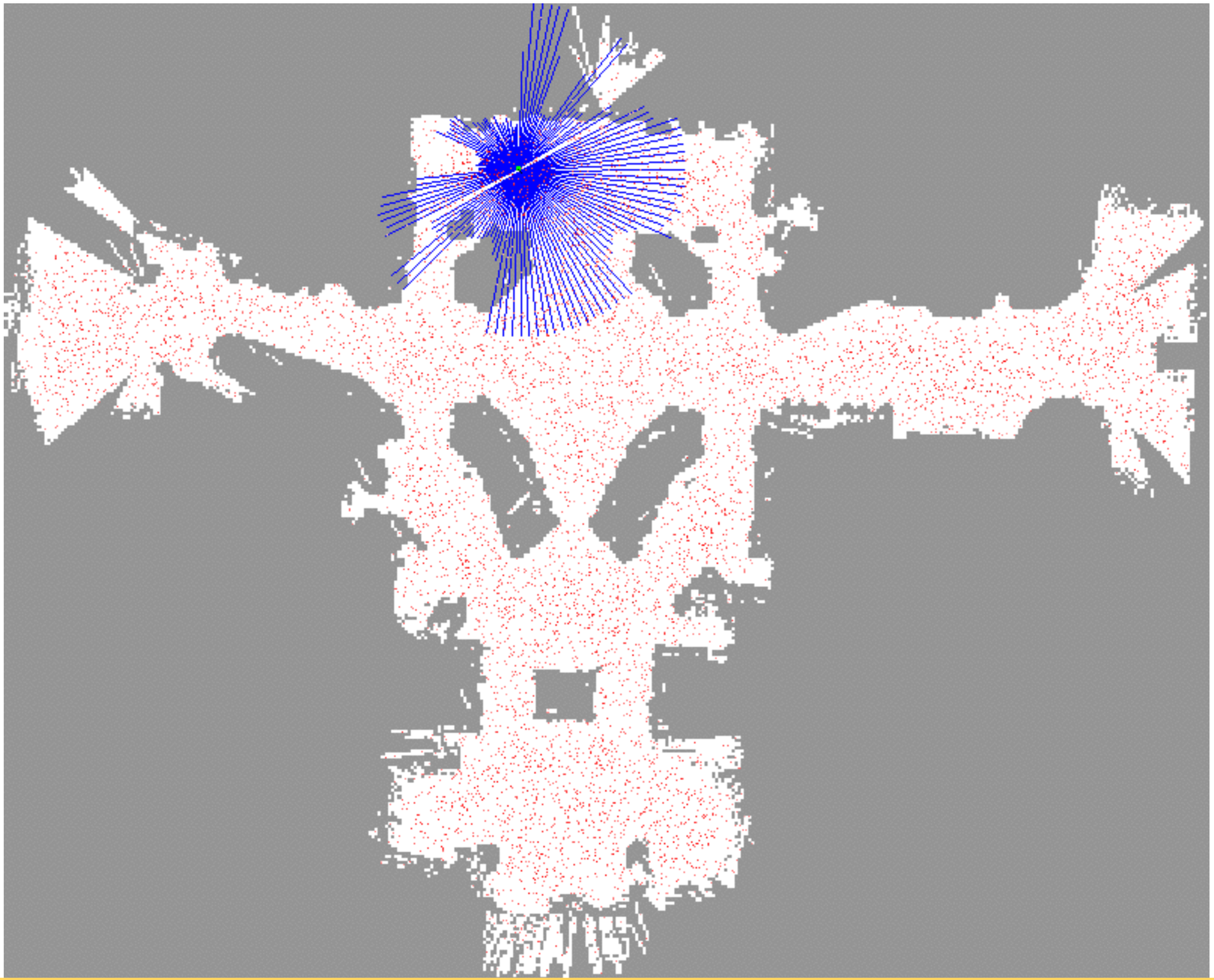
Robot Motion



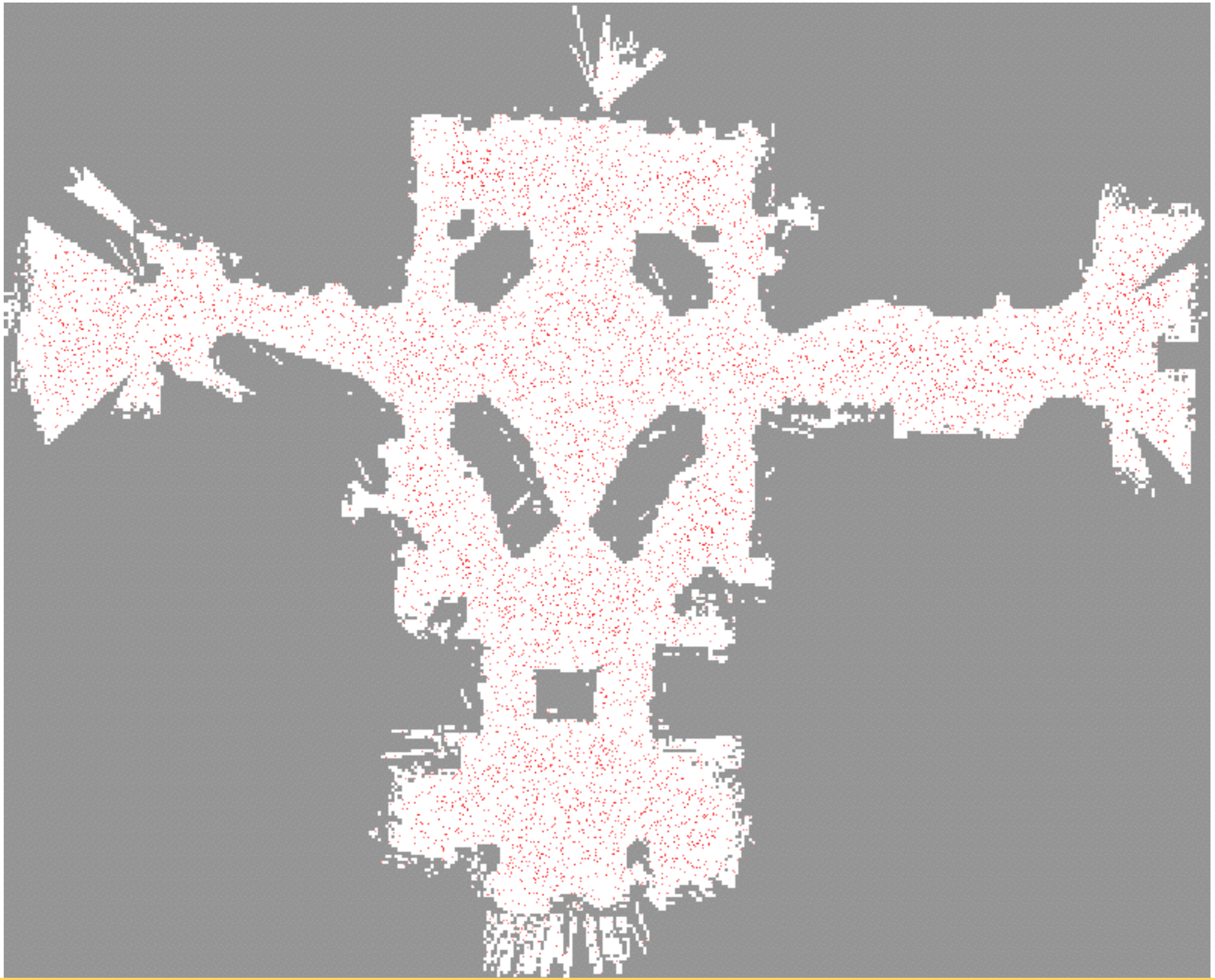
Slide borrowed from Dieter Fox



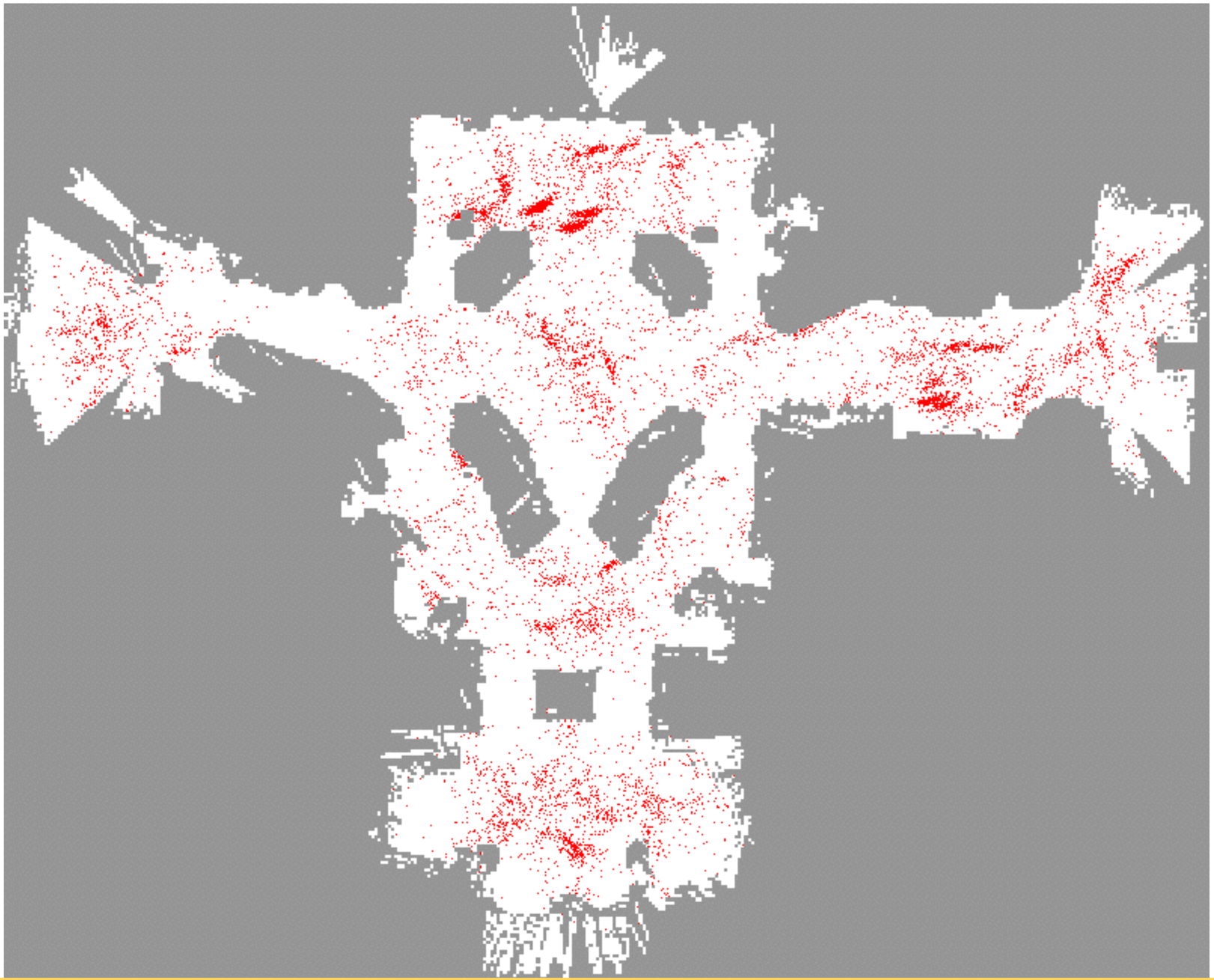
Slide borrowed from Dieter Fox



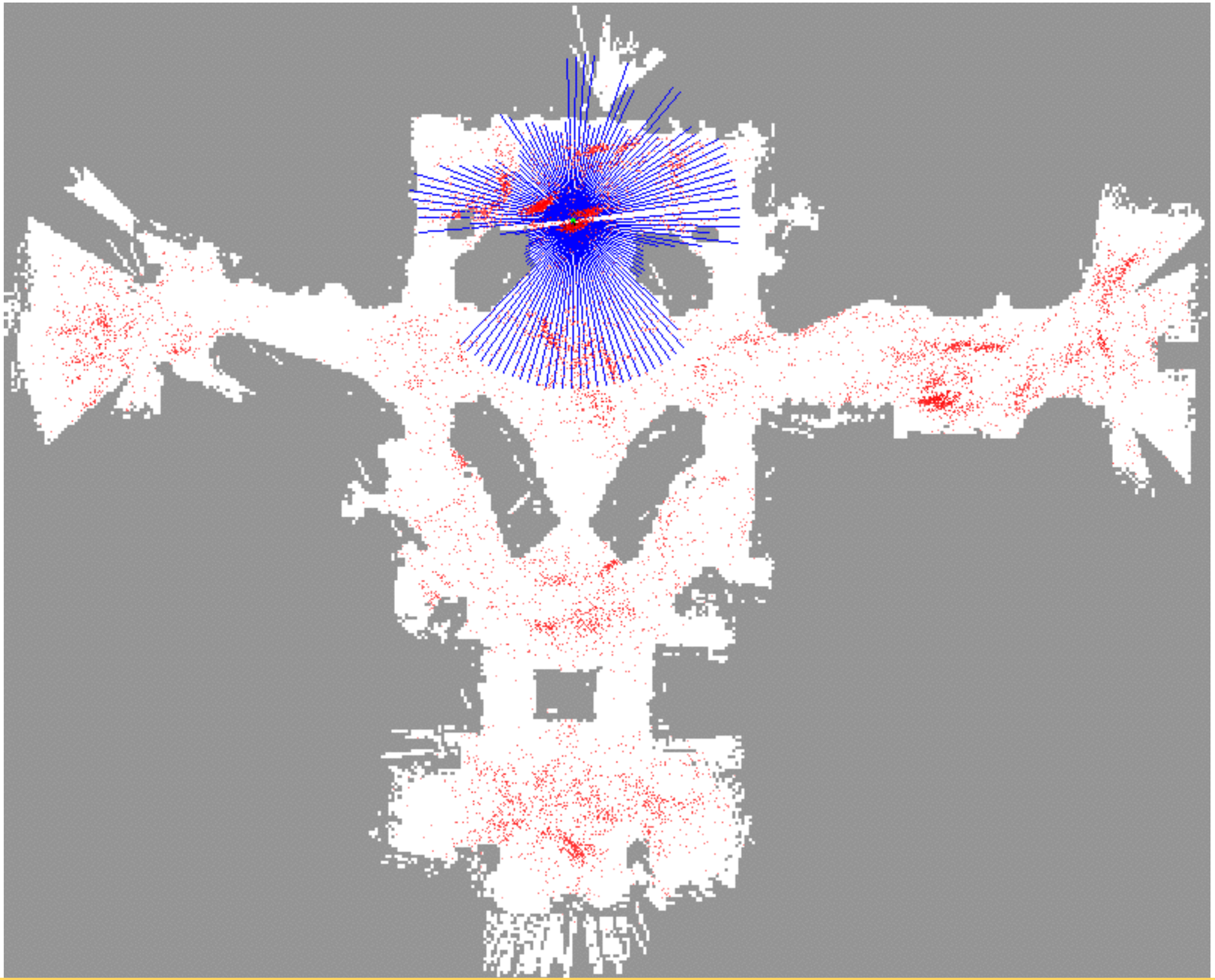
Slide borrowed from Dieter Fox



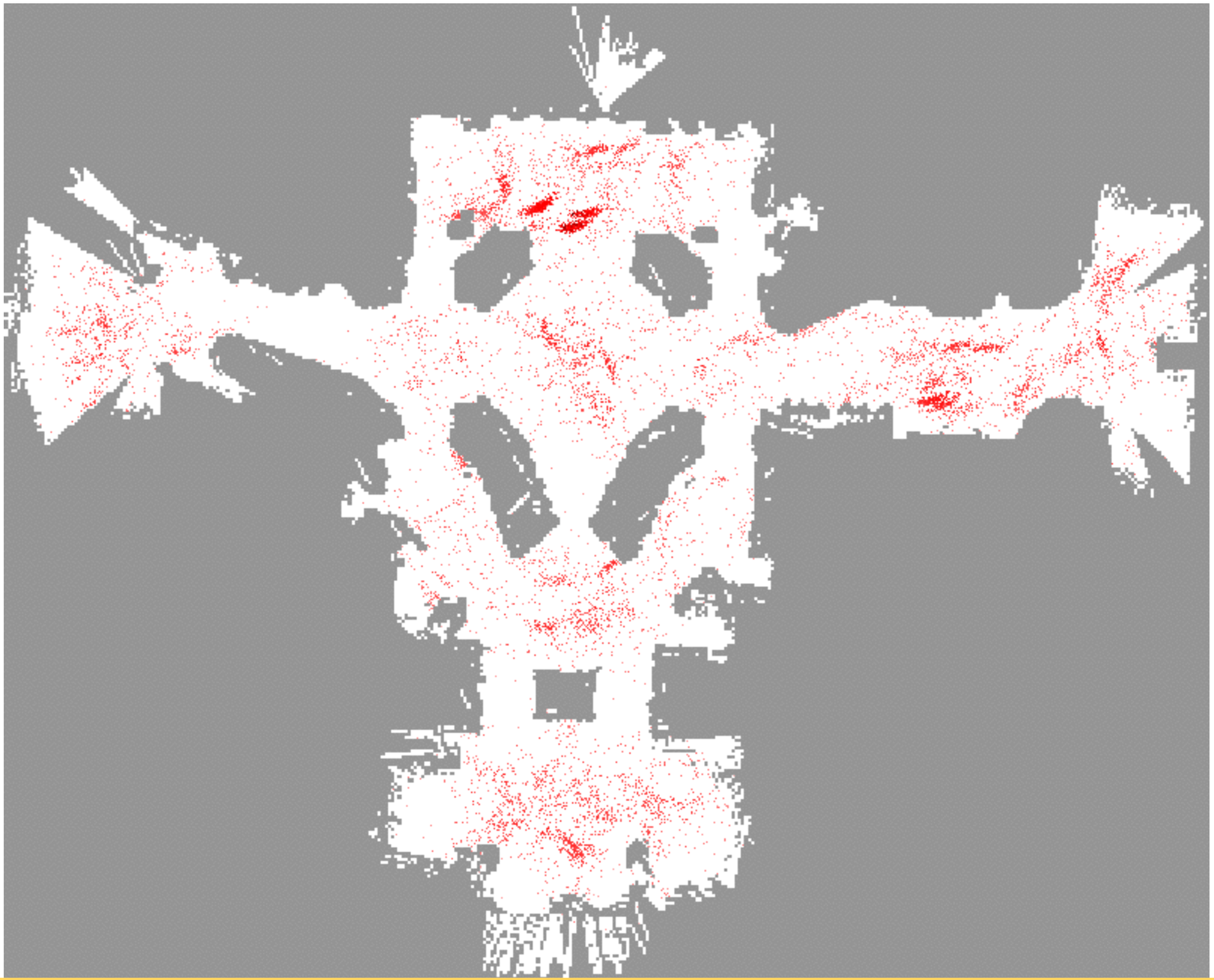
Slide borrowed from Dieter Fox



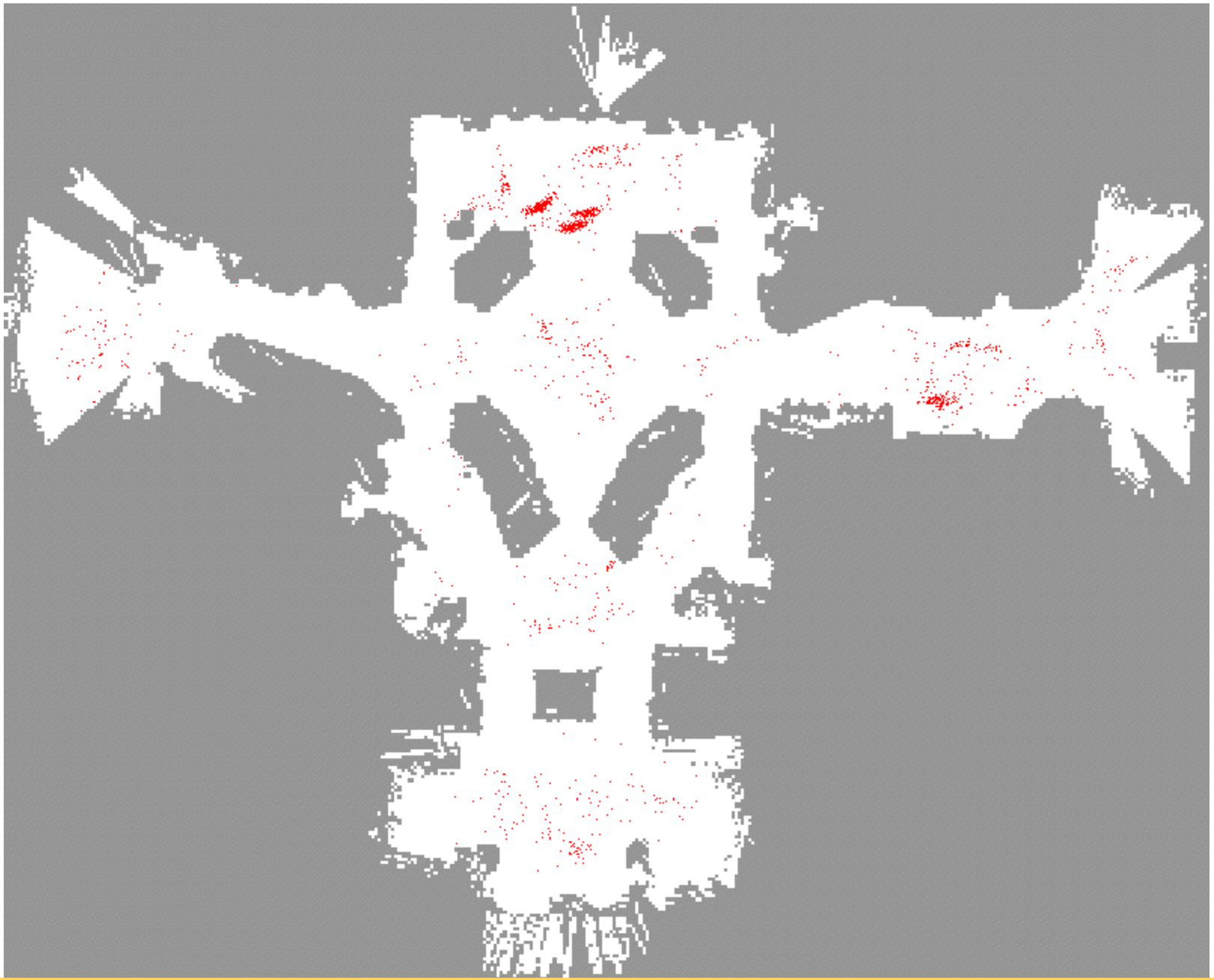
Slide borrowed from Dieter Fox



Slide borrowed from Dieter Fox

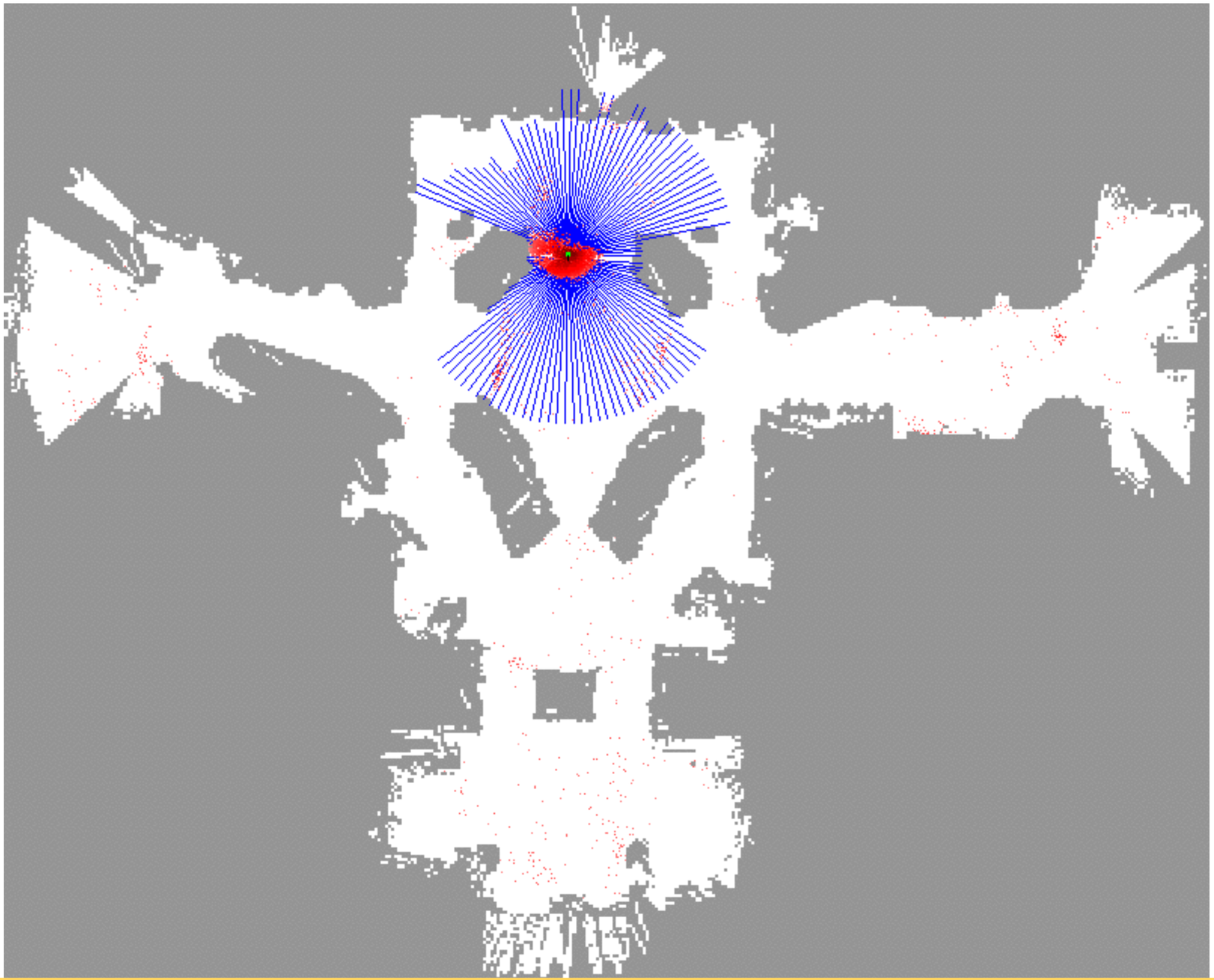


Slide borrowed from Dieter Fox



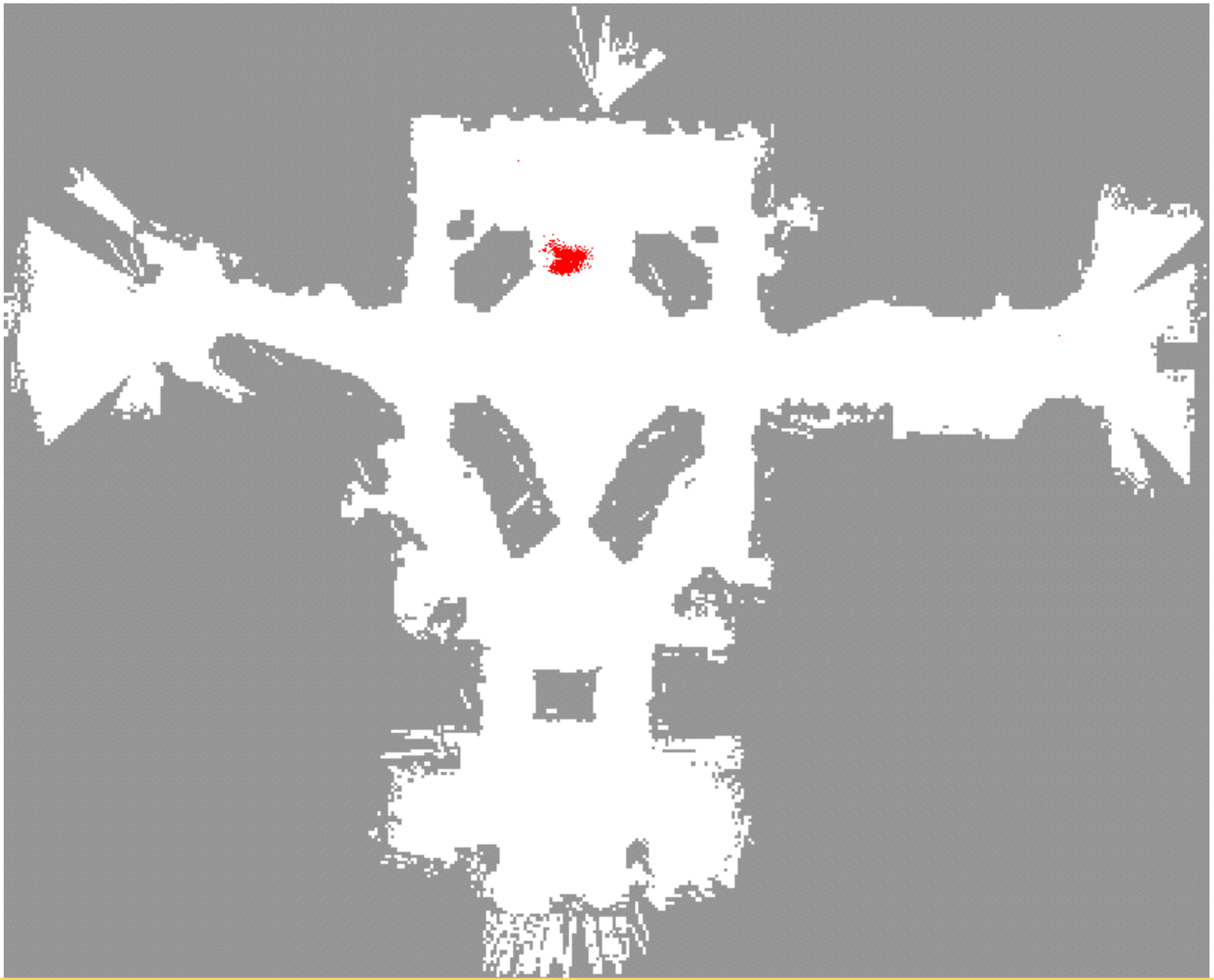
Slide borrowed from Dieter Fox

Slide borrowed from Dieter Fox

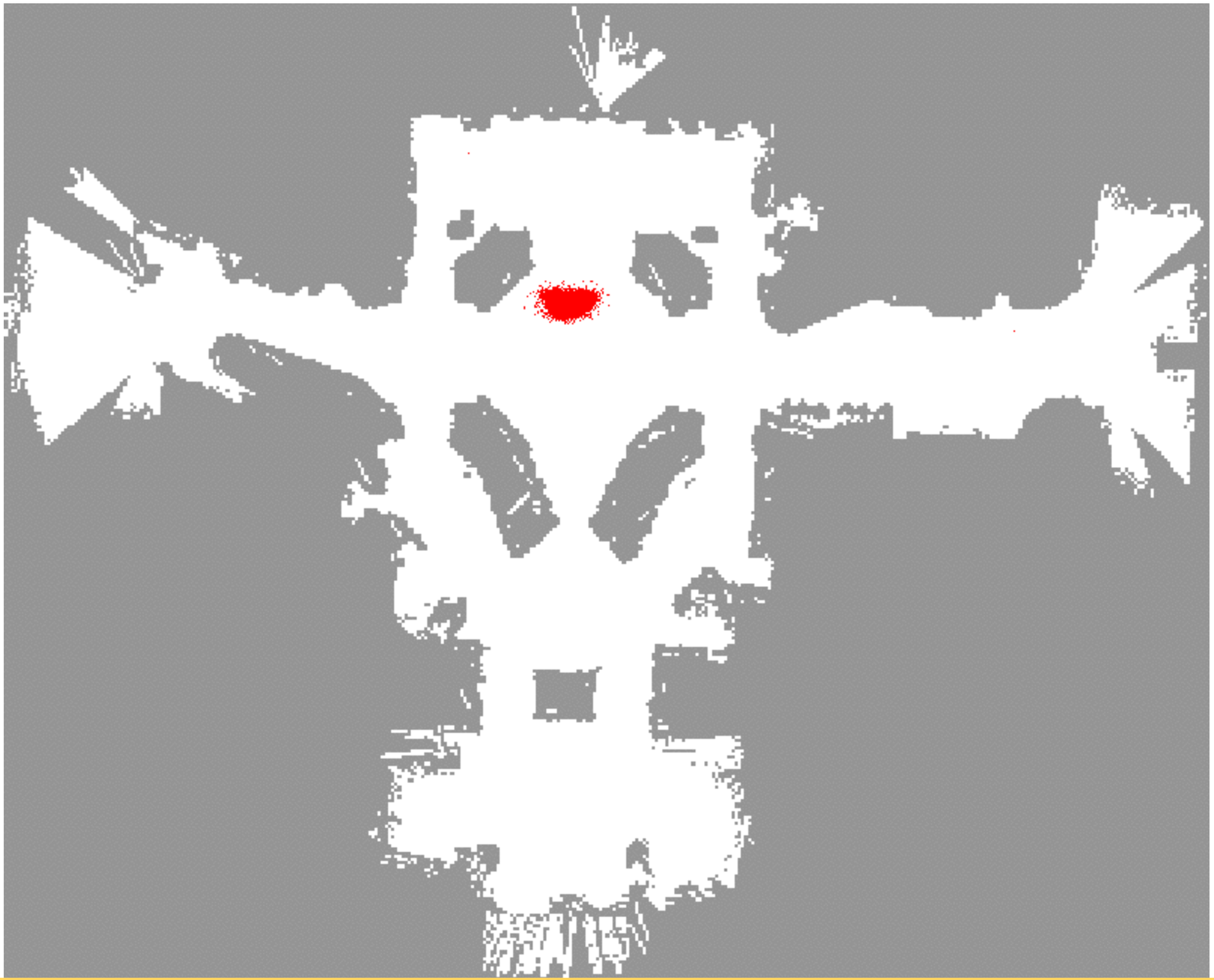


Slide borrowed from Dieter Fox

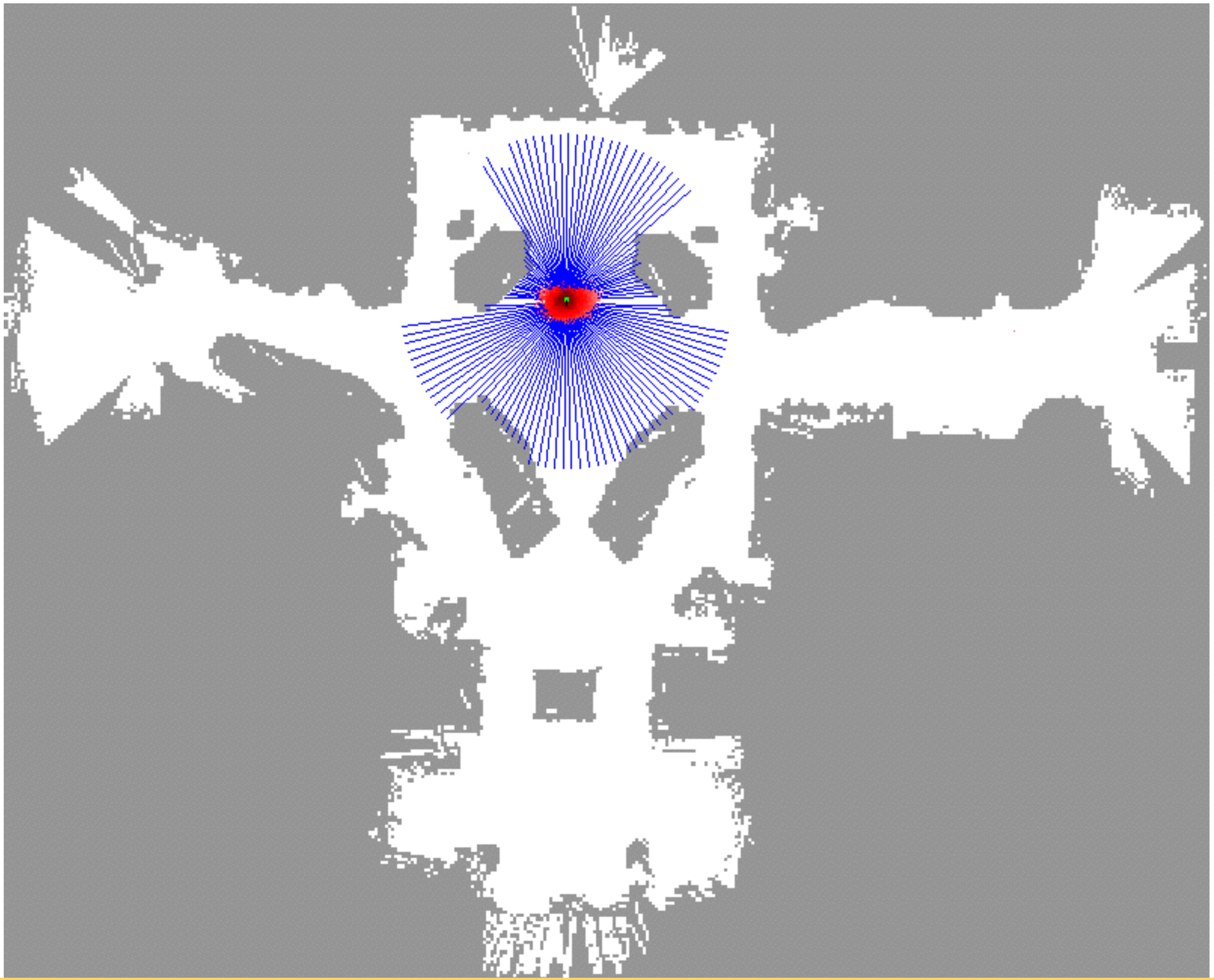
Slide borrowed from Dieter Fox



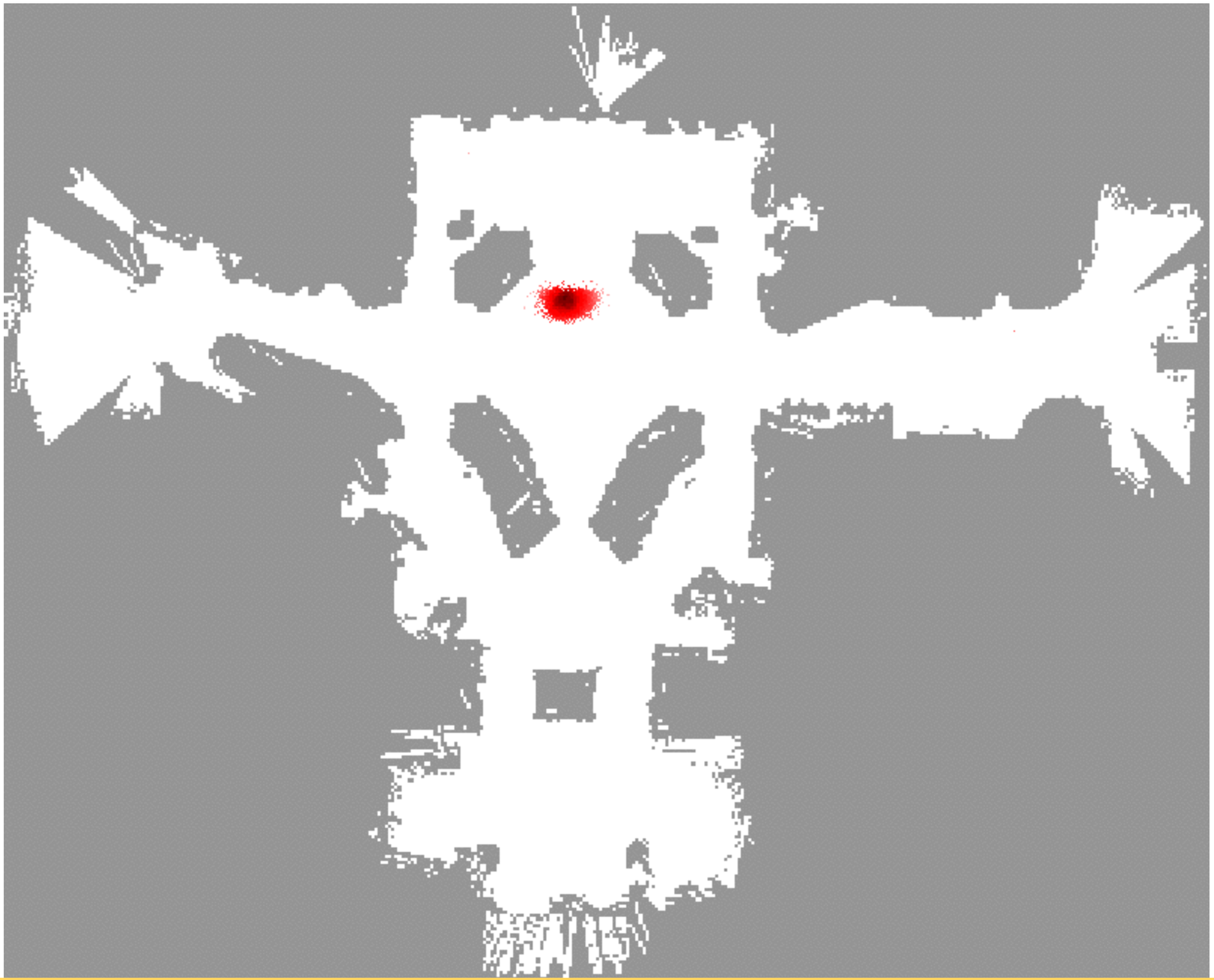
Slide borrowed from Dieter Fox



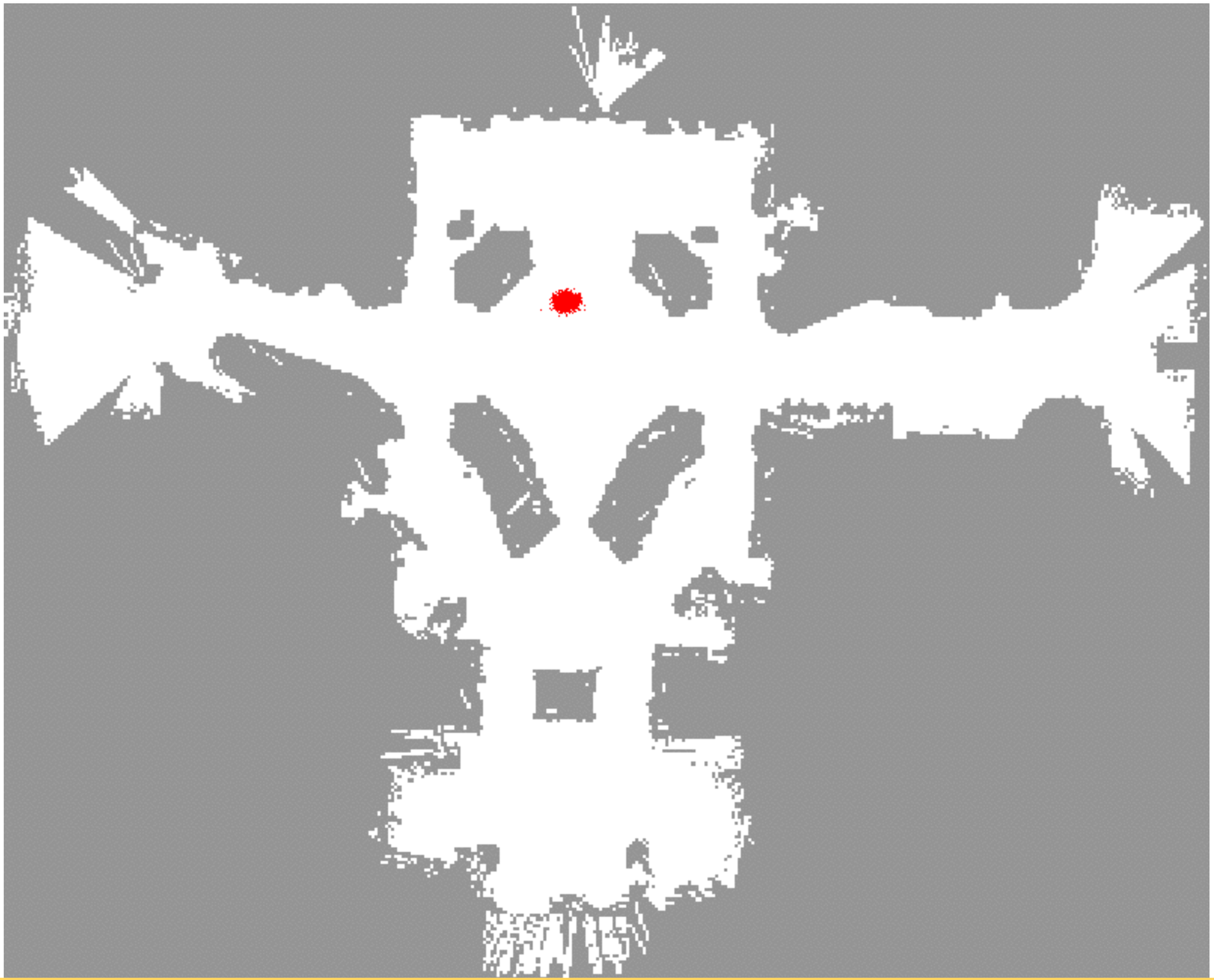
Slide borrowed from Dieter Fox



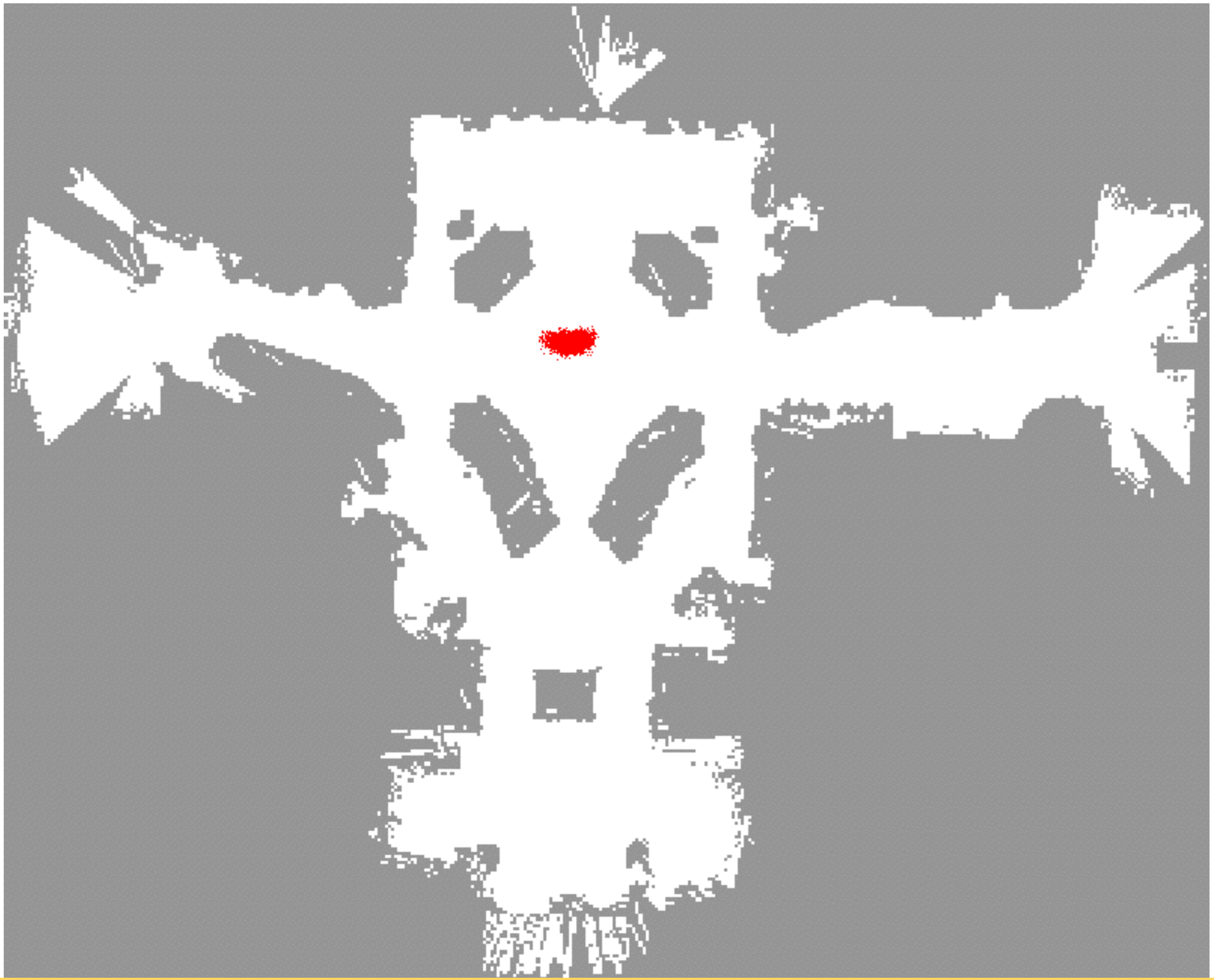
Slide borrowed from Dieter Fox



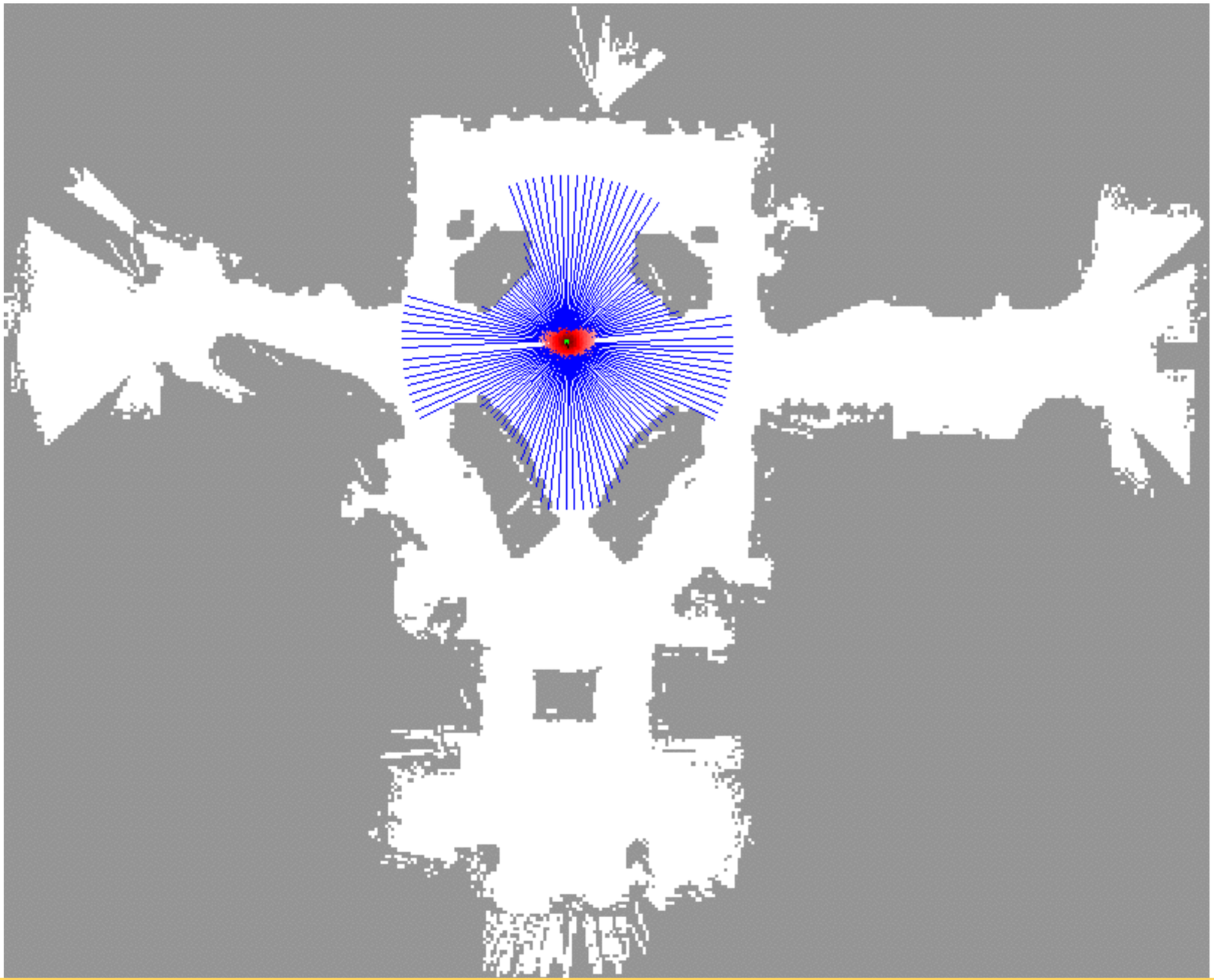
Slide borrowed from Dieter Fox



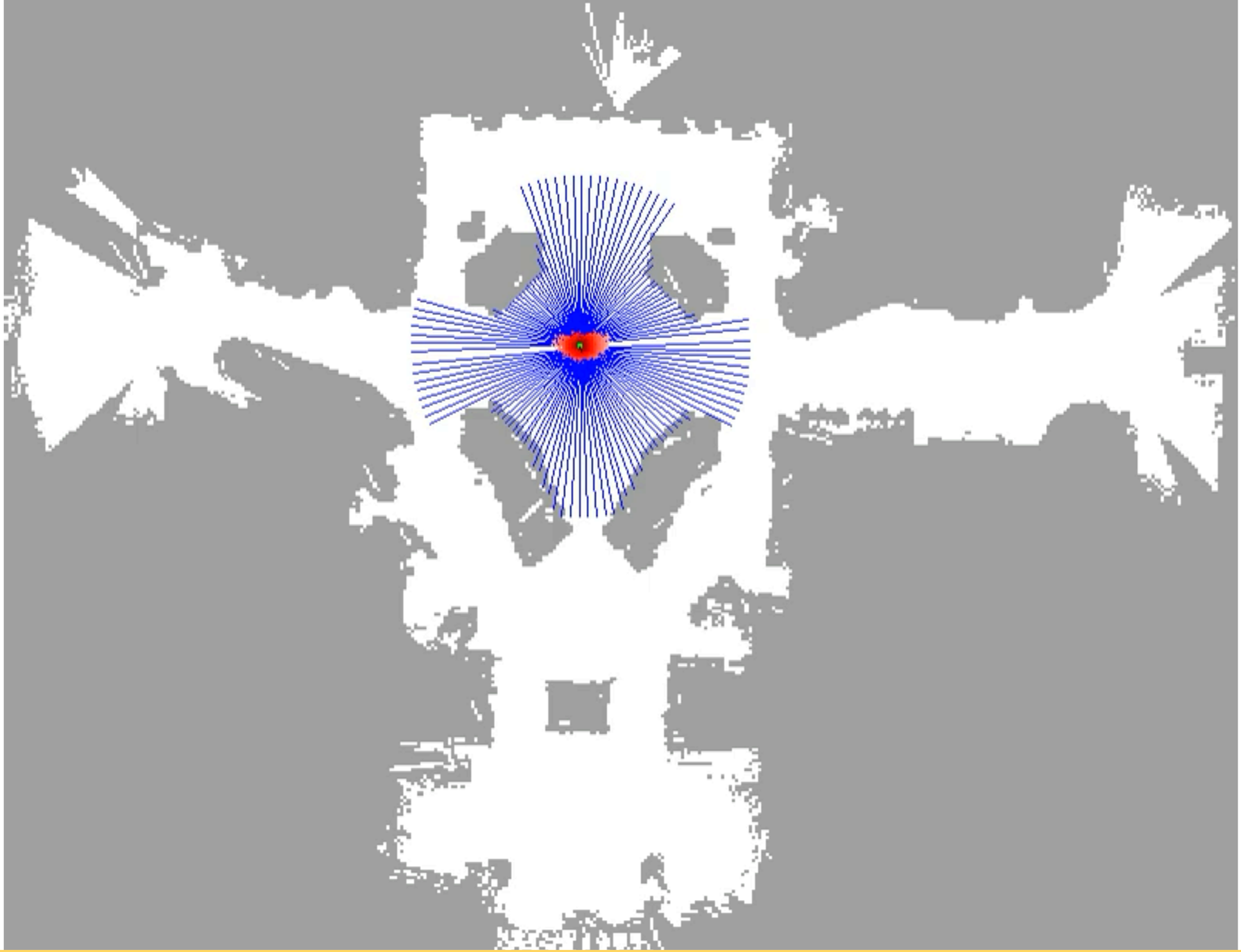
Slide borrowed from Dieter Fox



Slide borrowed from Dieter Fox



Slide borrowed from Dieter Fox



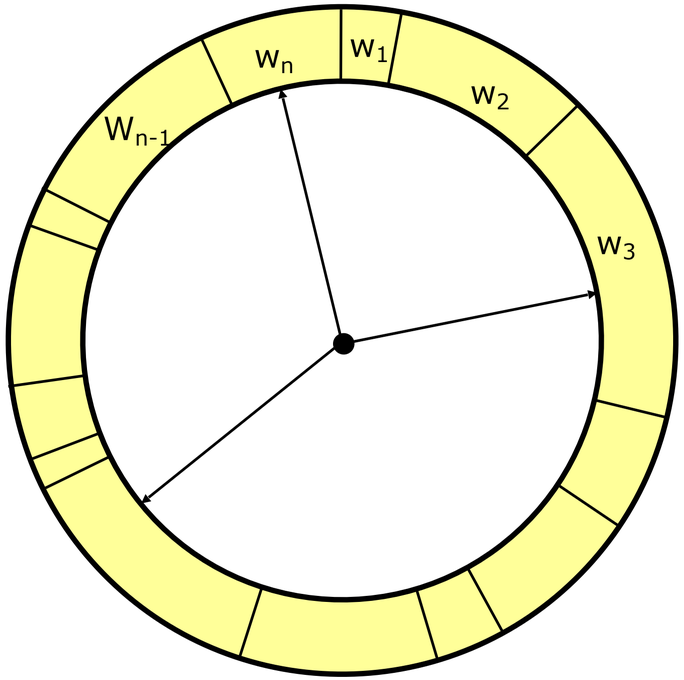
Slide borrowed from Dieter Fox

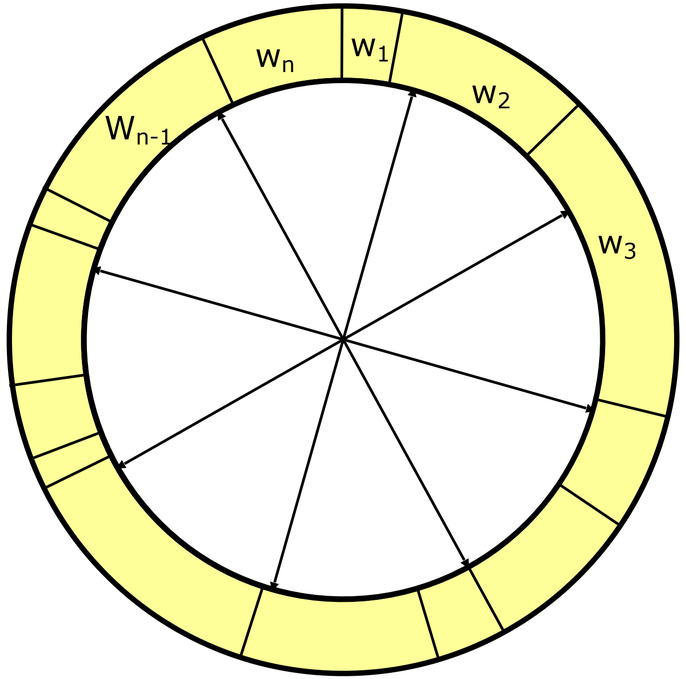
Resampling

- **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'.

Slide borrowed from Dieter Fox

Resampling



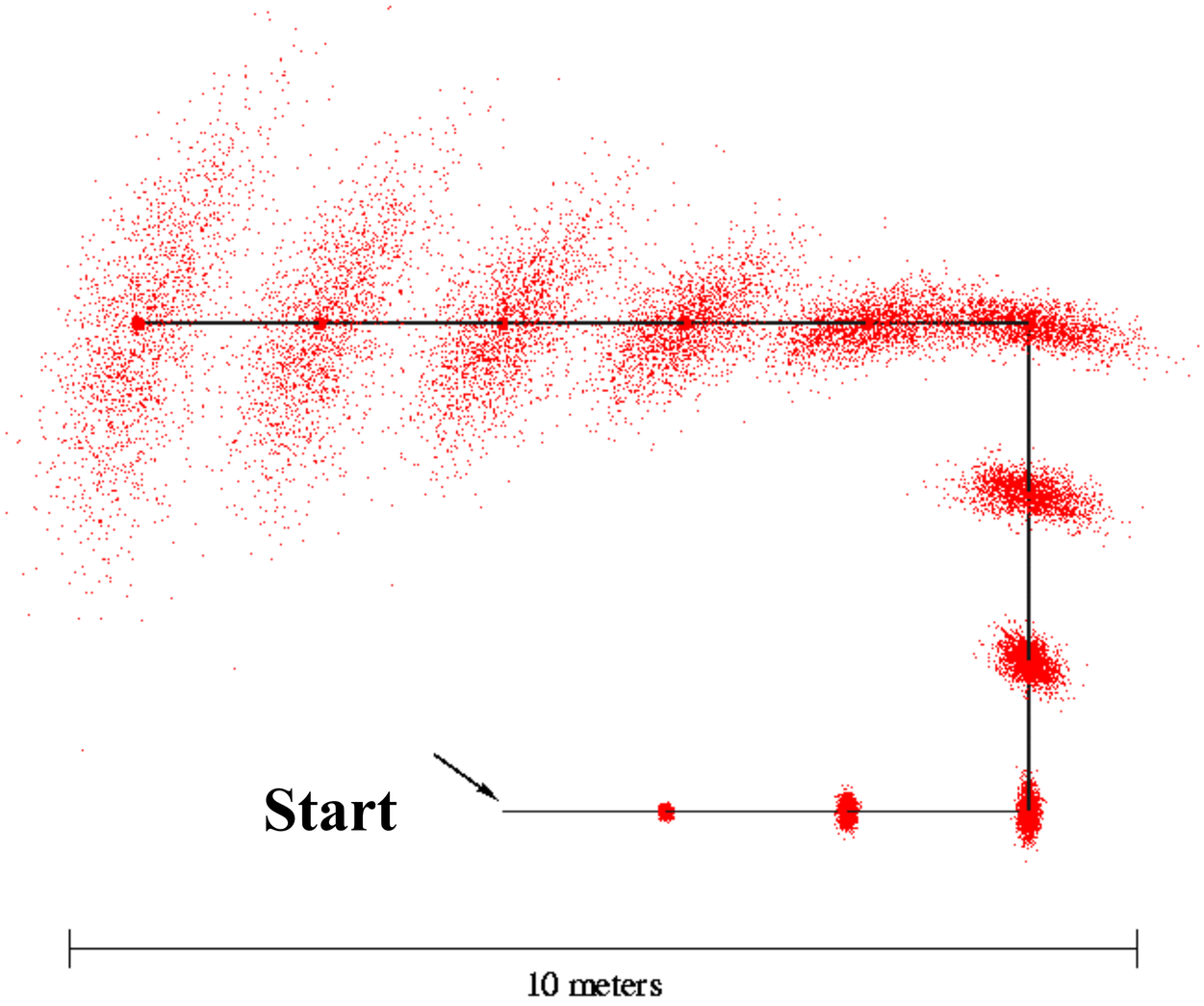


- Roulette wheel
- Binary search, n log n

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

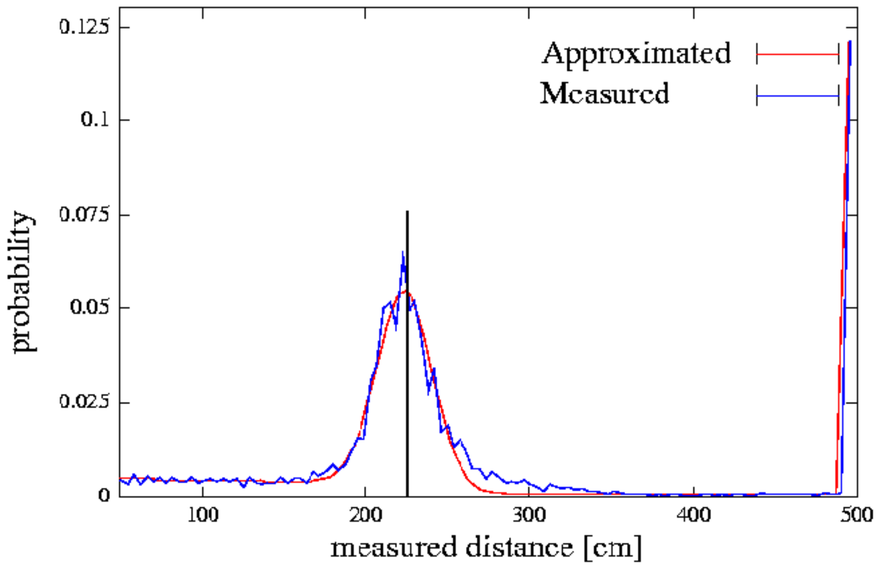
Slide borrowed from Dieter Fox

Motion Model Reminder

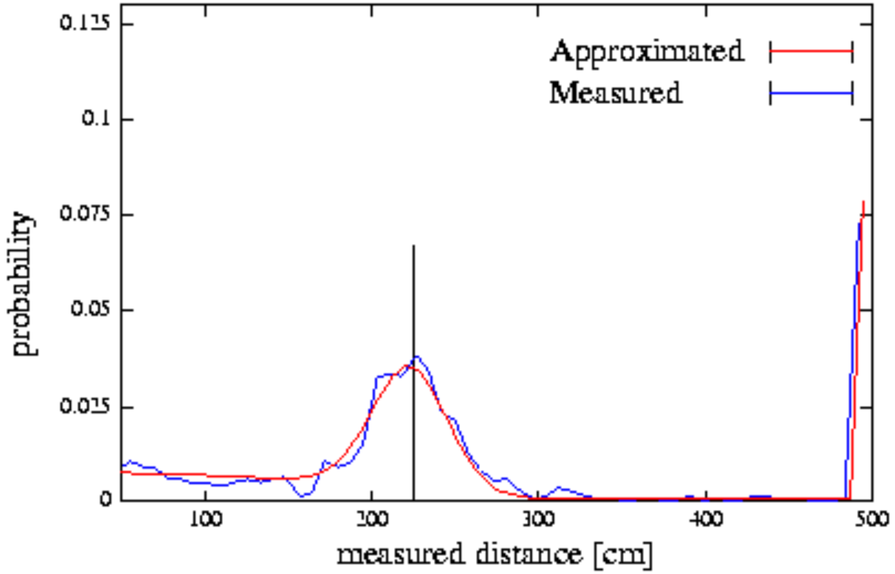


Slide borrowed from Dieter Fox

Proximity Sensor Model Reminder

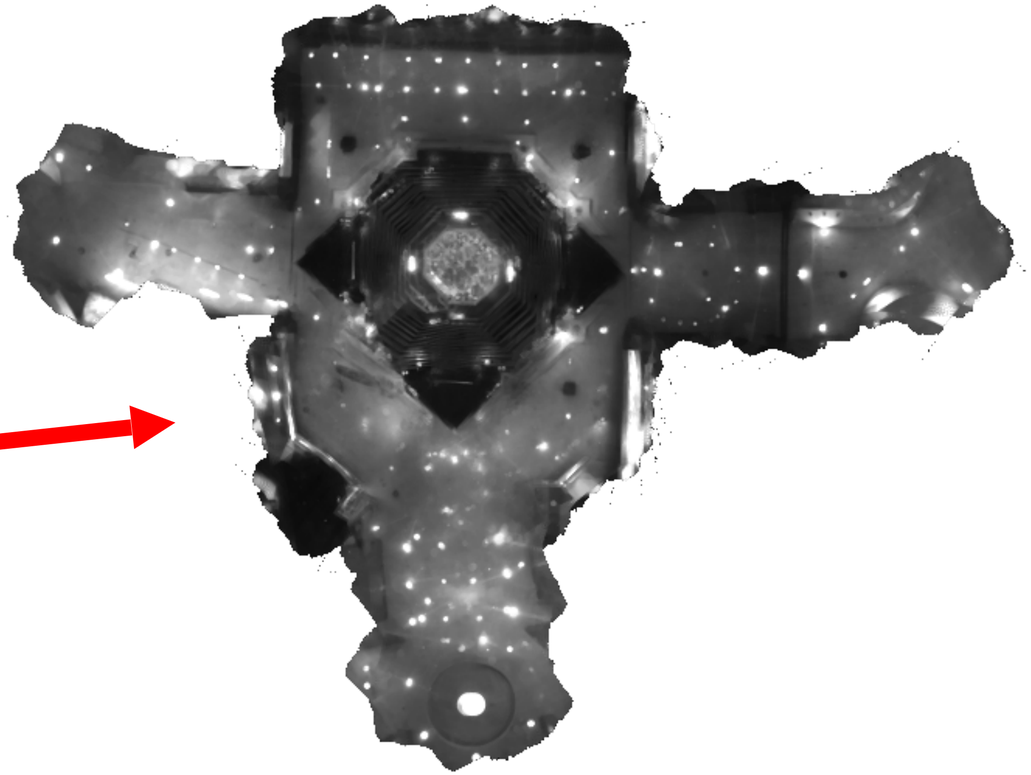


Laser sensor



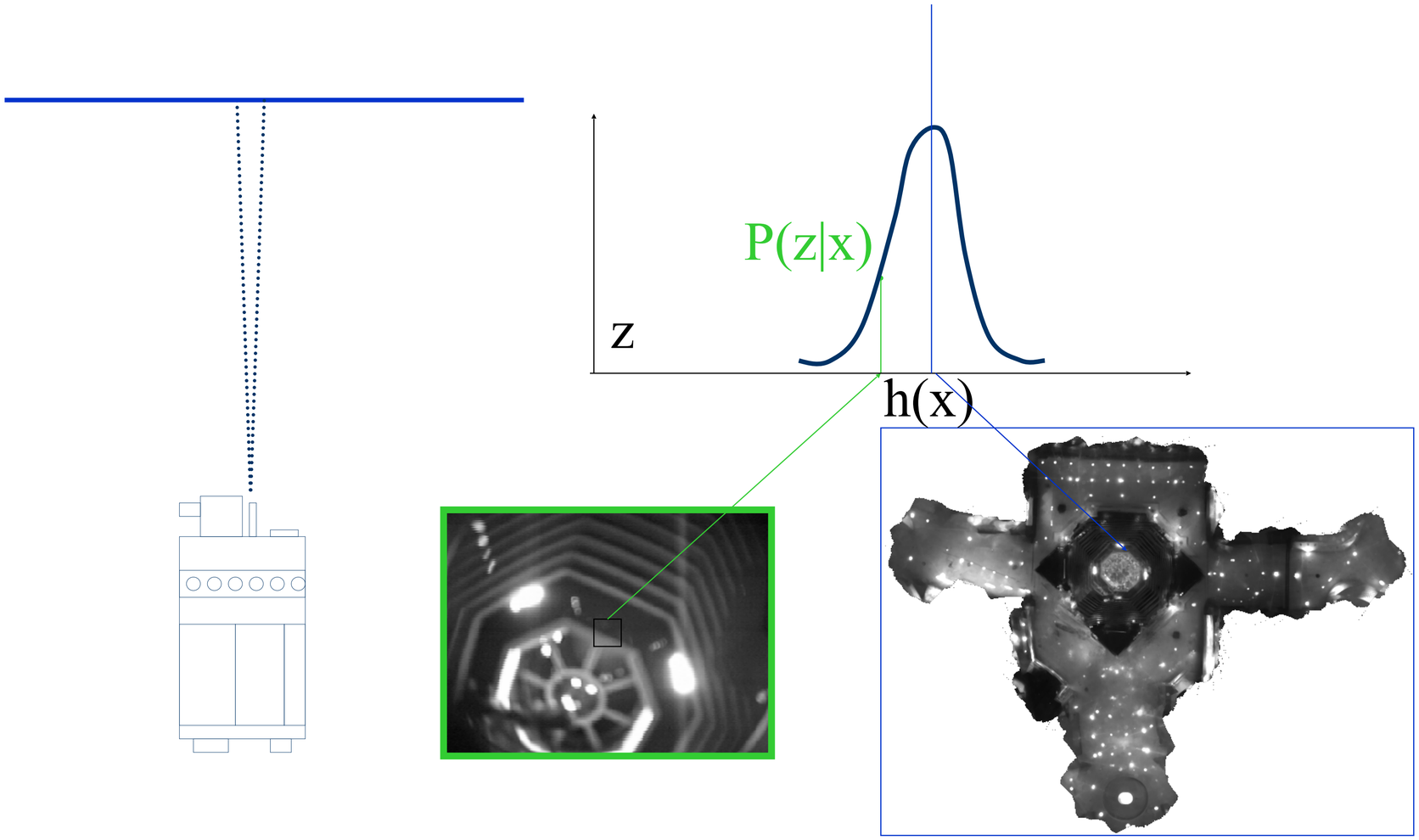
Sonar sensor

Using Ceiling Maps for Localization



Slide borrowed from Dieter Fox

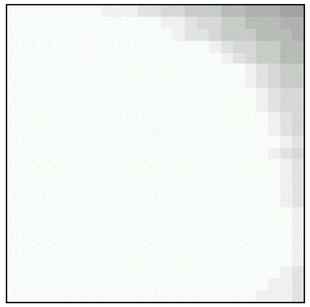
Vision-based Localization



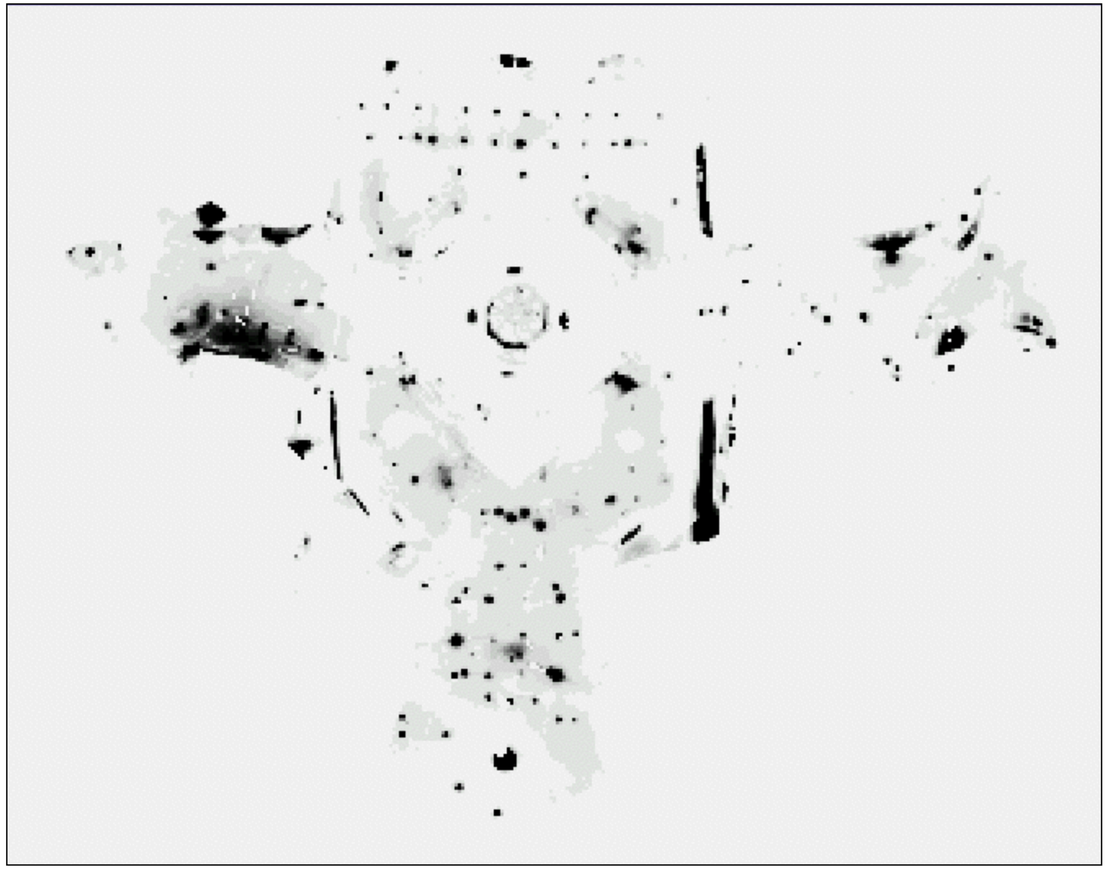
Slide borrowed from Dieter Fox

Under a Light

Measurement z:



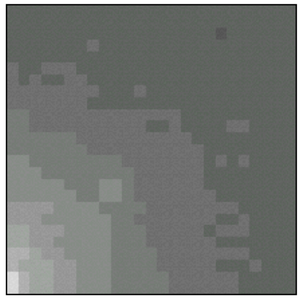
***P(z|x)*:**



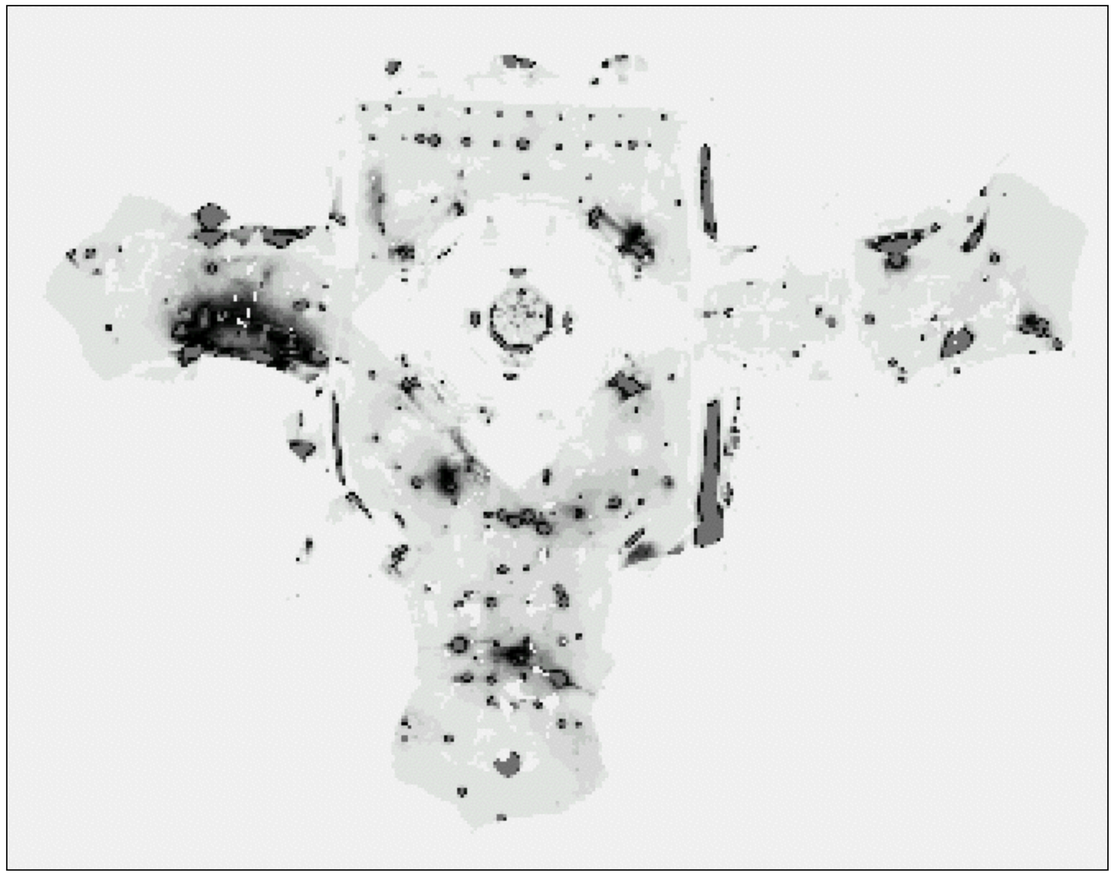
Slide borrowed from Dieter Fox

Next to a Light

Measurement z:



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

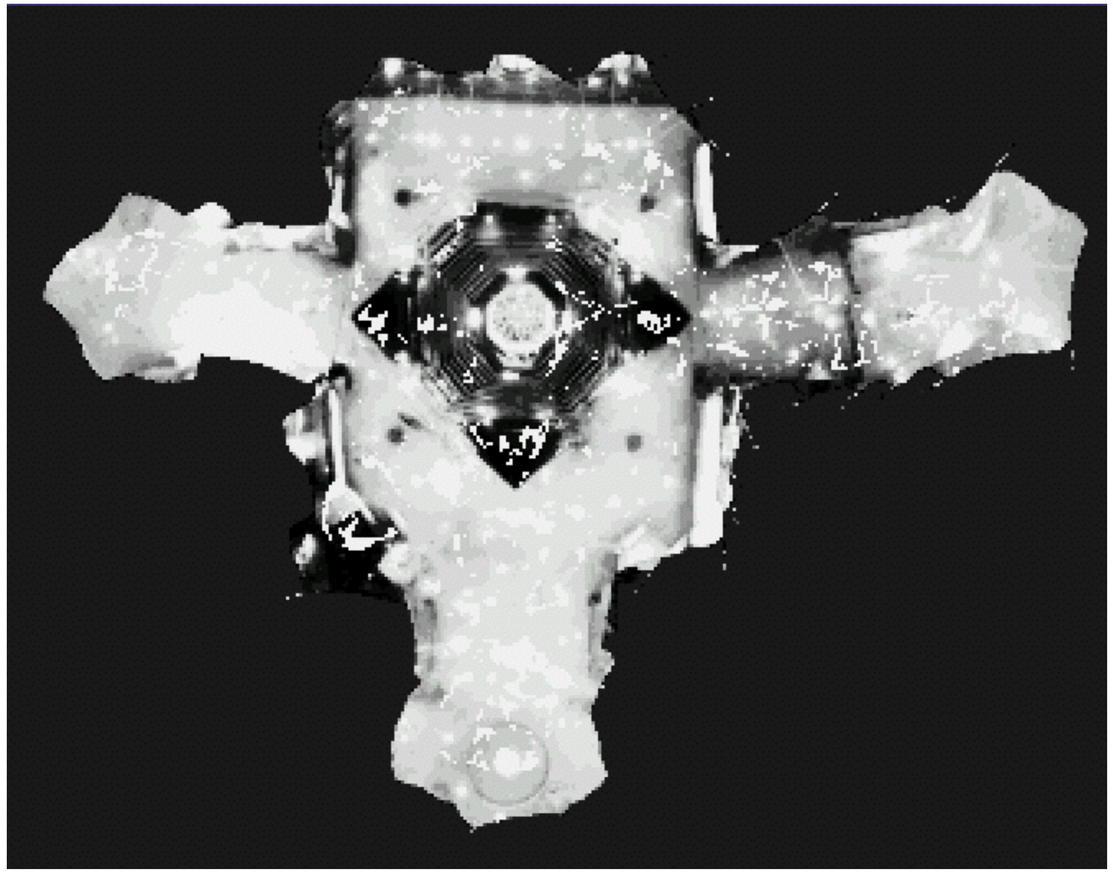


Slide borrowed from Dieter Fox

Elsewhere

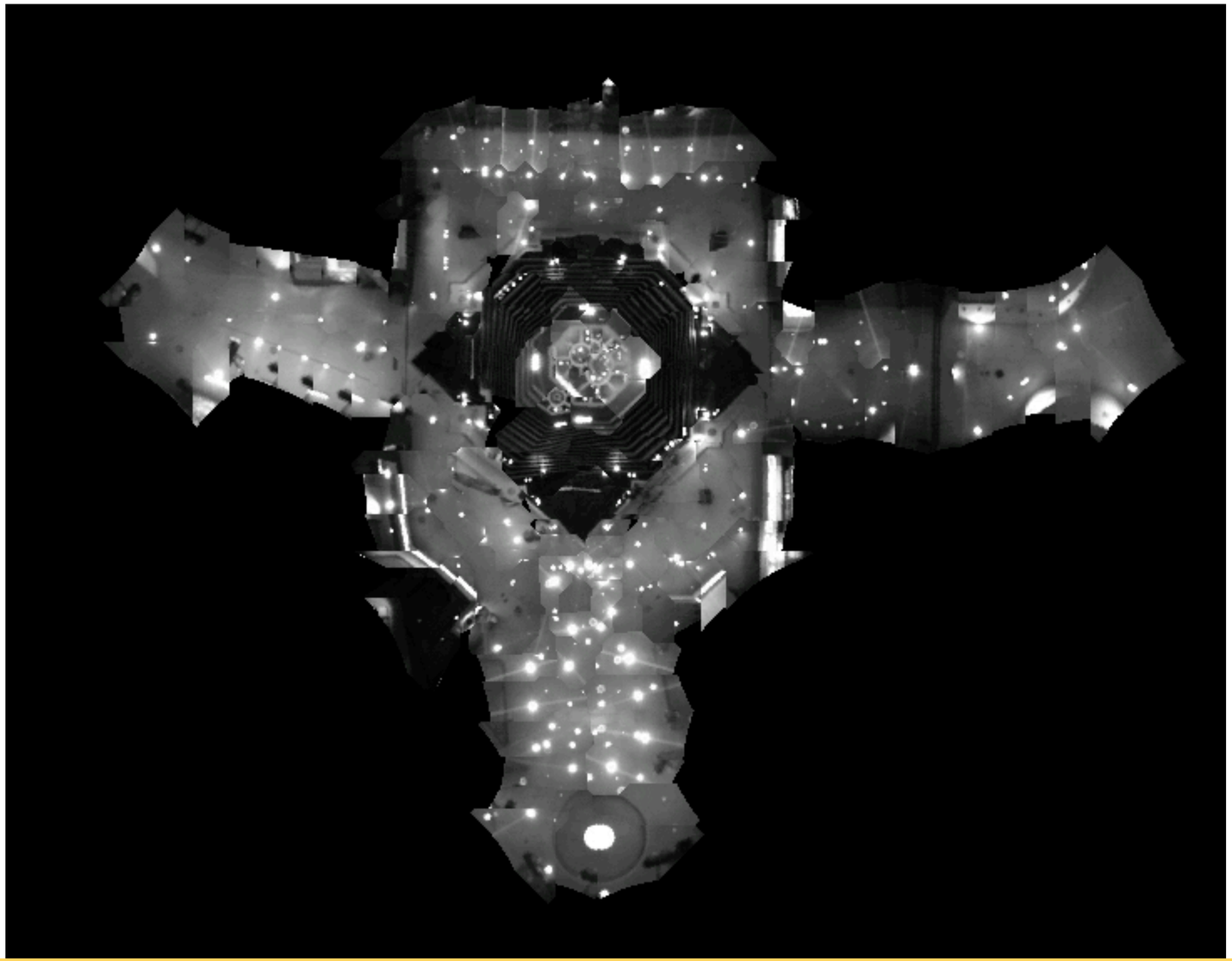
Measurement z:

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



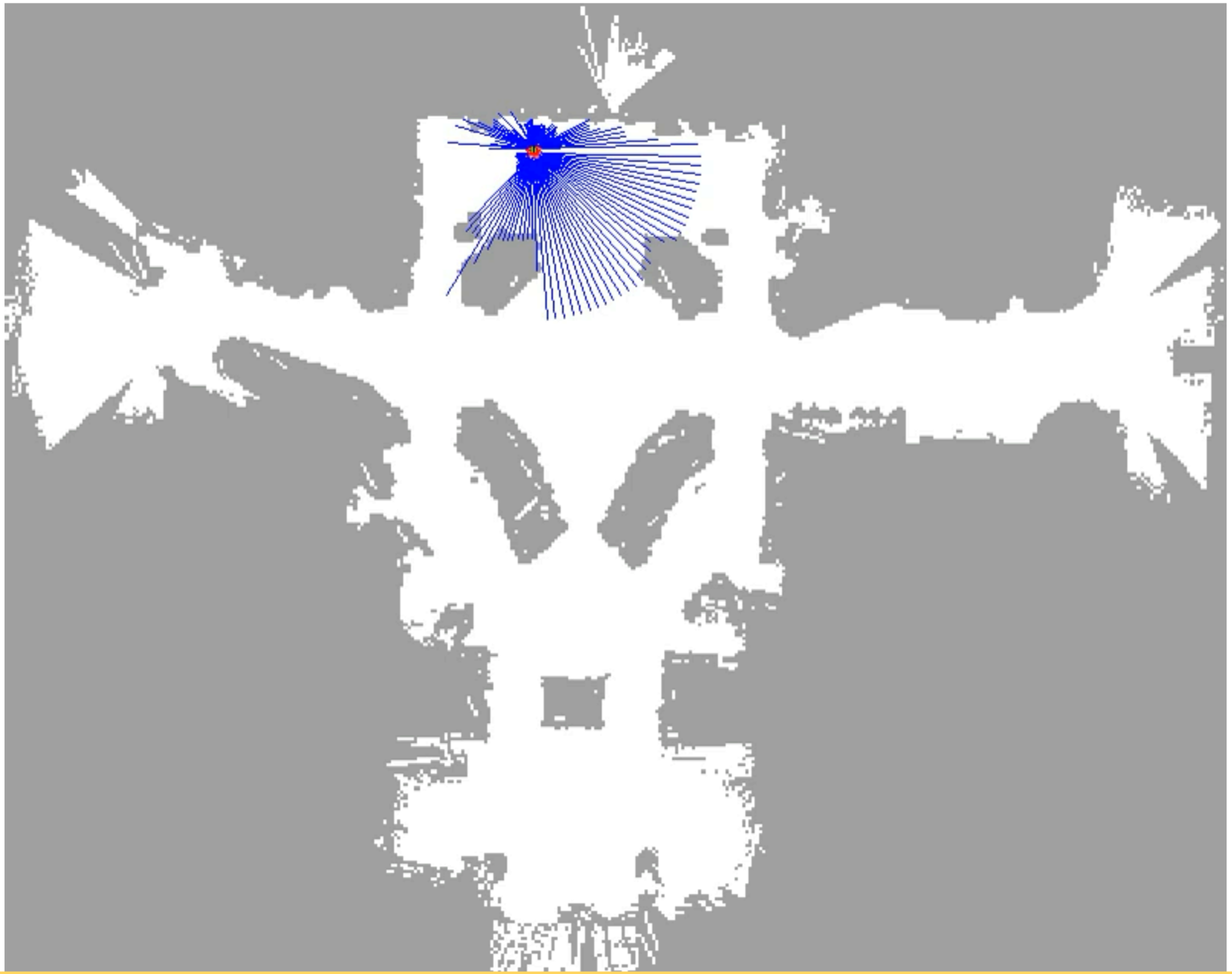
Slide borrowed from Dieter Fox

Global Localization Using Vision



Slide borrowed from Dieter Fox

Recovery from Failure



Slide borrowed from Dieter Fox

Next Lecture:
More PF and Mapping