Bayesian Methods

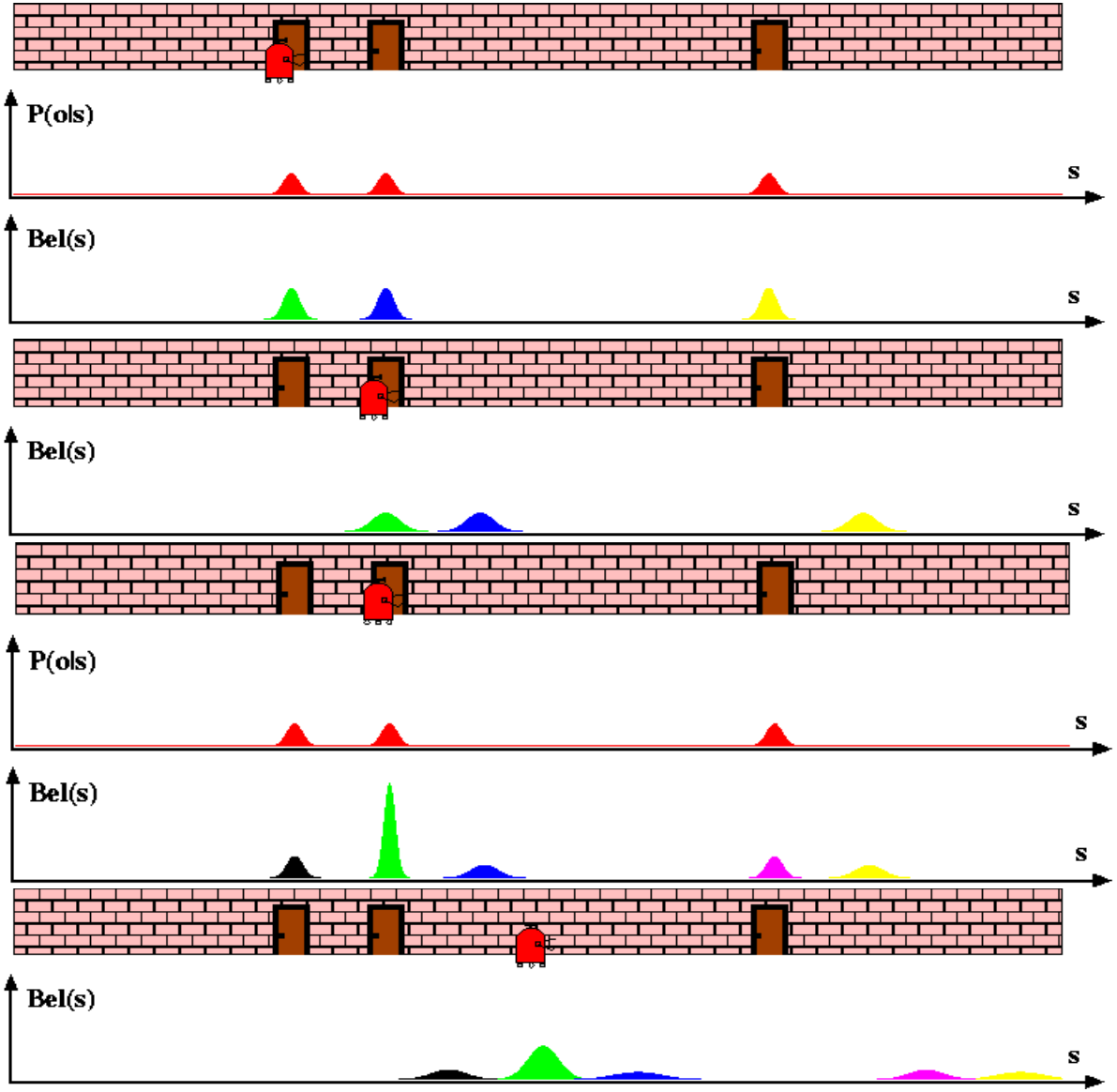
Recall Robot Localization

- Given
  - Sensor readings $\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_t = \mathbf{z}_{1:t}$
  - **Known control inputs $\mathbf{u}_0, \mathbf{u}_1, \ldots \mathbf{u}_t = \mathbf{u}_{0:t}$**
  - **Known model** $P(\mathbf{x}_{t+1} \mid \mathbf{x}_t, \mathbf{u}_t)$ with initial $P(\mathbf{x}_1 \mid \mathbf{u}_0)$
  - **Known map** $P(\mathbf{z}_t \mid \mathbf{x}_t)$ → Most likely sensor reading given state $\mathbf{x}$
- Compute
  - $P(\mathbf{x}_t \mid \mathbf{z}_{1:t-1}, \mathbf{u}_{0:t-1})$ → Most likely state $\mathbf{x}$ at time t given a sequence of commands $\mathbf{u}_{0:t-1}$ and measurements $\mathbf{z}_{1:t-1}$

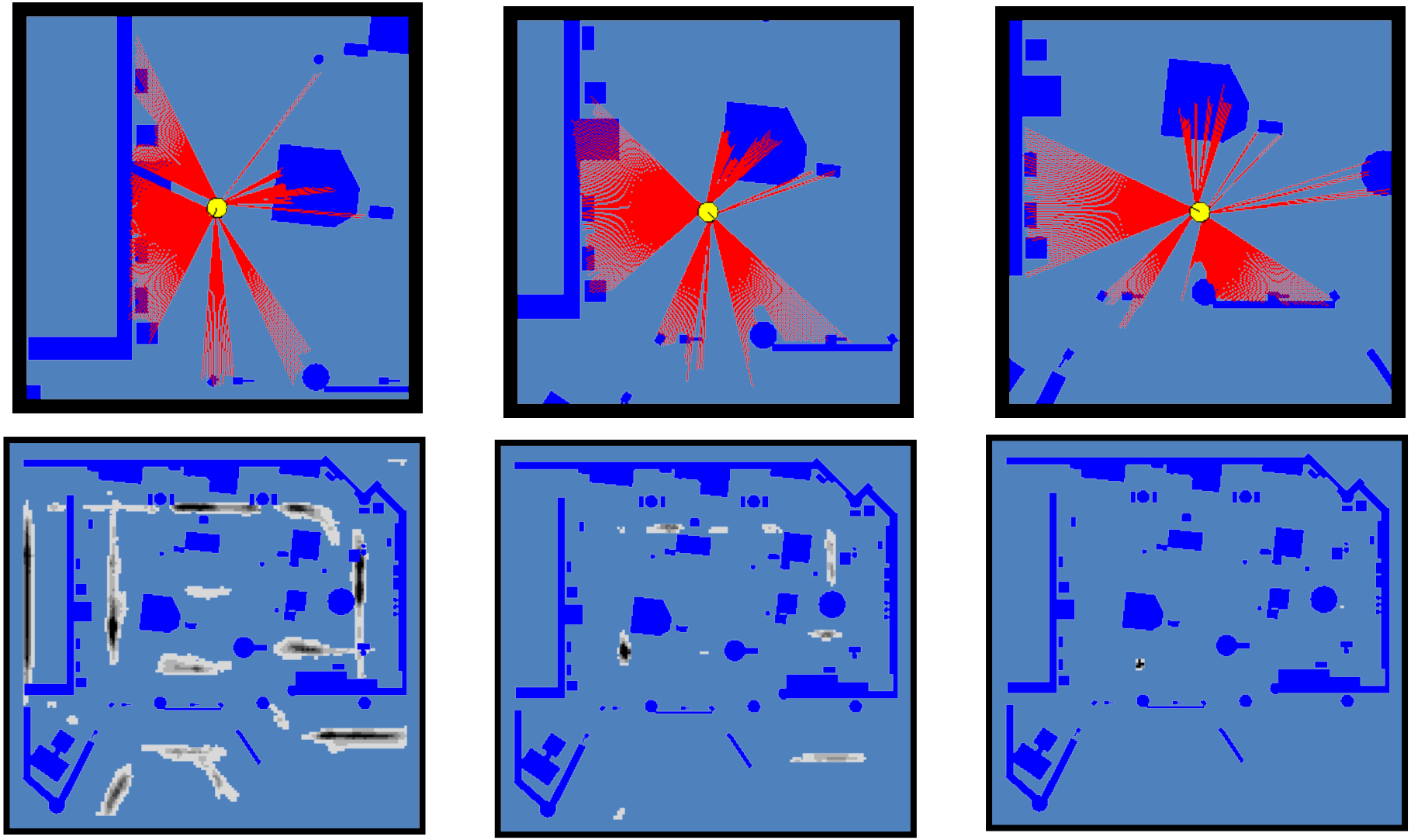
This is just a probabilistic representation of what you've already learned!

Let's try to connect the dots and do a couple of examples

A Simple Example

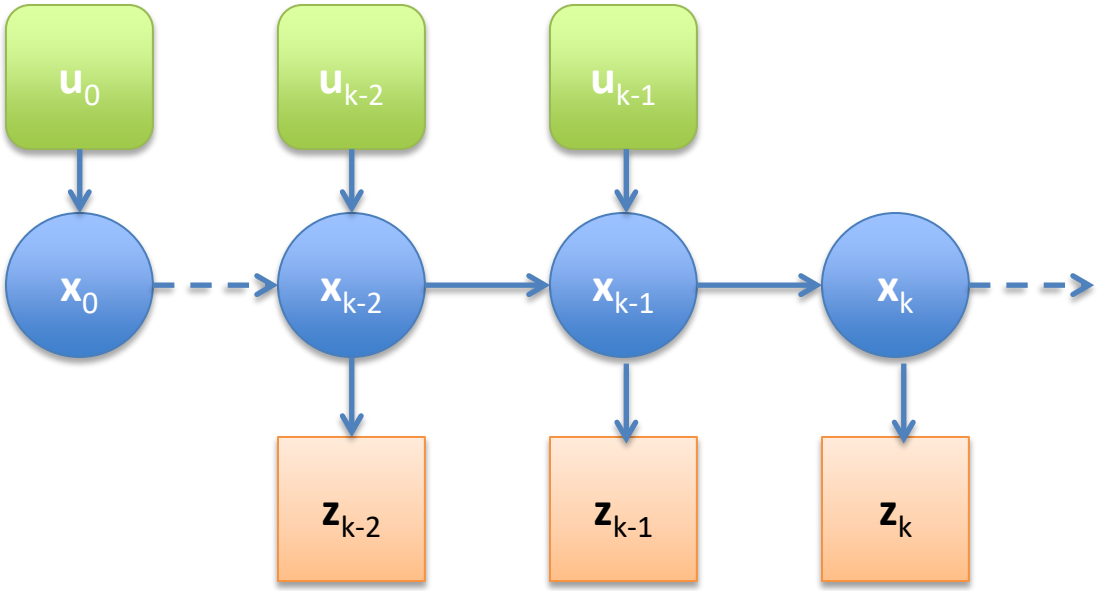


Why Not Just Use a Kalman Filter?



Bayes Filter

- Given a sequence of measurements $\mathbf{z}_1, \ldots, \mathbf{z}_k$
- Given a sequence of commands $\mathbf{u}_0, \ldots, \mathbf{u}_{k-1}$
- Given a sensor model $P(\mathbf{z}_k \mid \mathbf{x}_k)$
- Given a dynamic model $P(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \underline{\mathbf{u}}_{k-1})$
- Given a prior probability $P(\mathbf{x}_0)$
- Find $P(\mathbf{x}_k \mid \mathbf{z}_{1:k}, \mathbf{u}_{0:k-1})$



Bayesian Filtering

- Recall Bayes Theorem:

$$P(x\,|z) = \frac{P(z|x)P(x)}{P(z)}$$

- Also remember conditional independence
- Think of $\mathbf{x}$ as the state of the robot and $\mathbf{z}$ as the data we know

$P(\boldsymbol{x}_k|\boldsymbol{u}_{0:k-1}, \boldsymbol{z}_{1:k})$ → *Posterior Probability Distribution*

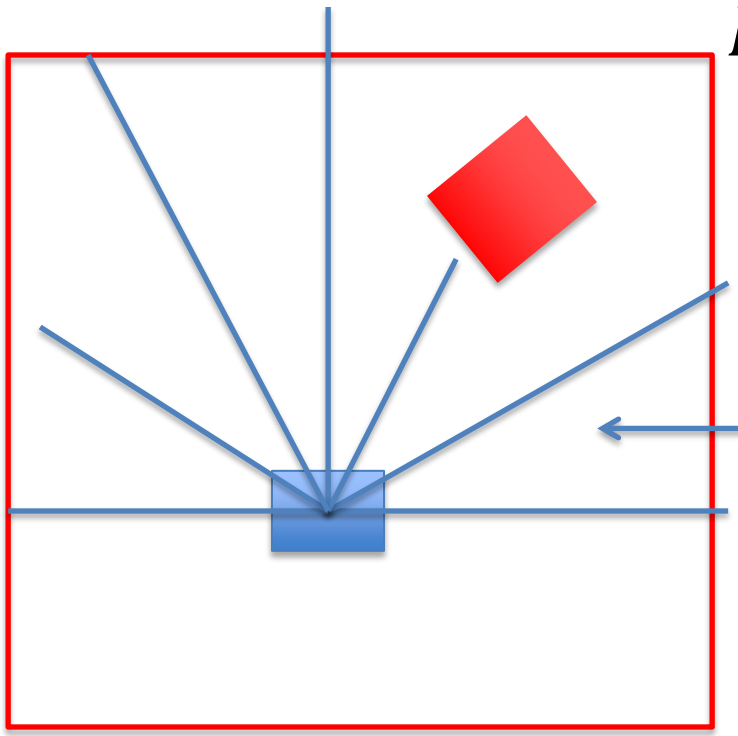
$$P(\boldsymbol{x}_k|\boldsymbol{u}_{0:k-1}, \boldsymbol{z}_{1:k}) = \frac{P(\boldsymbol{z}_k|\boldsymbol{x}_k, \boldsymbol{u}_{0:k-1}, \boldsymbol{z}_{1:k-1})P(\boldsymbol{x}_k|\boldsymbol{u}_{0:k-1}, \boldsymbol{z}_{1:k-1})}{P(\boldsymbol{z}_k|\boldsymbol{u}_{0:k-1}, \boldsymbol{z}_{1:k-1})}$$

$$= \underbrace{\eta_k P(\boldsymbol{z}_k|\boldsymbol{x}_k)}_{\textbf{observation}} \int_{\boldsymbol{x}_{k-1}} \underbrace{P(\boldsymbol{x}_k|\boldsymbol{u}_{k-1}, \boldsymbol{x}_{k-1})}_{\textbf{state prediction}} \underbrace{P(\boldsymbol{x}_{k-1}|\boldsymbol{u}_{0:k-2}, \boldsymbol{z}_{1:k-1})}_{\textbf{recursive instance}}$$

Observation Model

- What is the probability that a given state $\boldsymbol{x}_k$ generates the measurement $\boldsymbol{z}_k$?

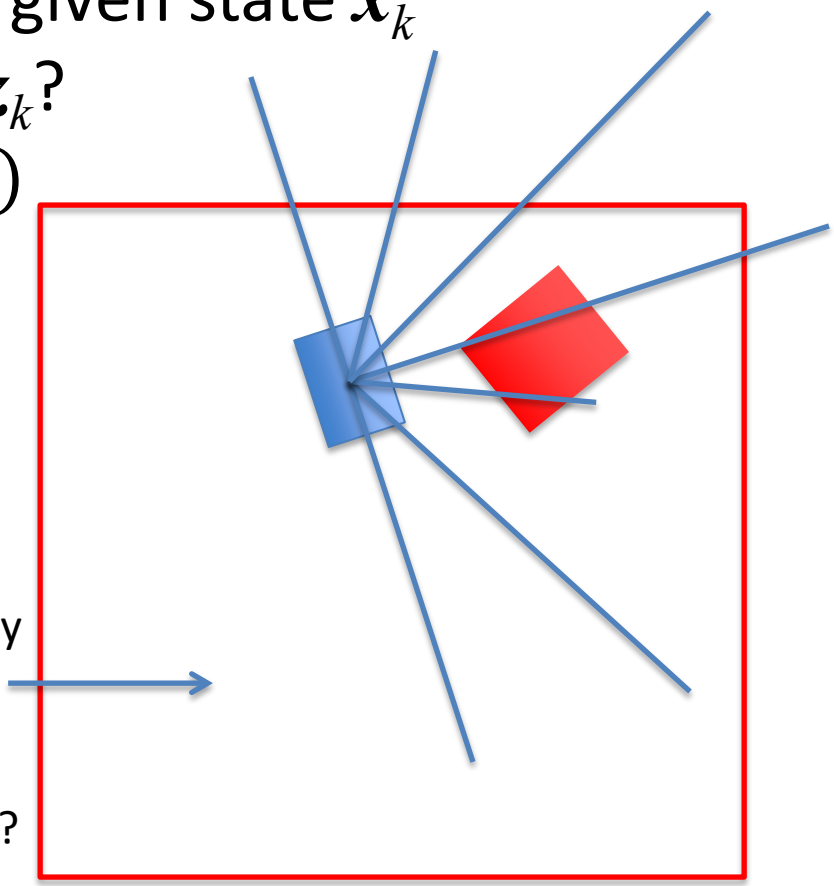
$$P(\boldsymbol{z}_k|\boldsymbol{x}_k)$$



Same $\boldsymbol{z}_k$
Different $\boldsymbol{x}_k$

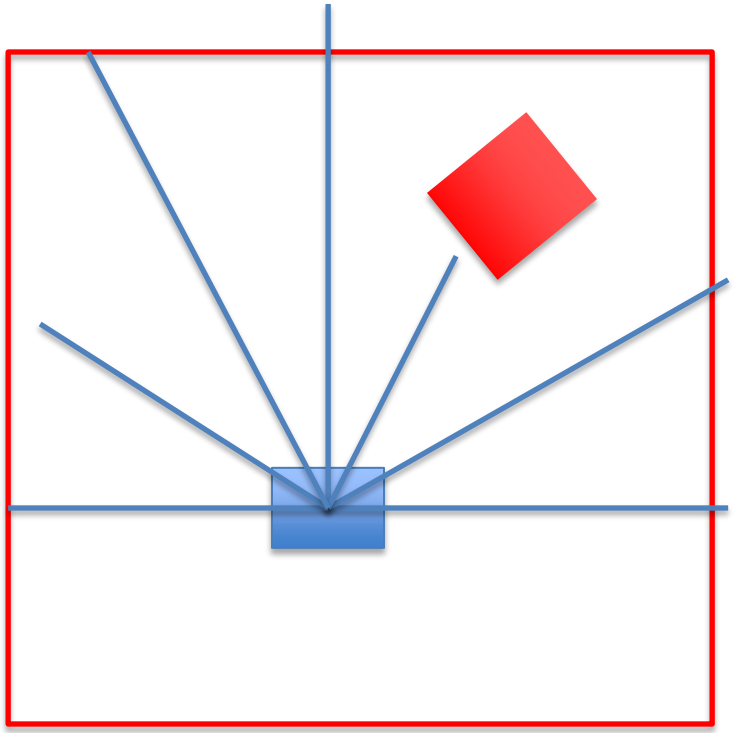
This
is more likely
than that

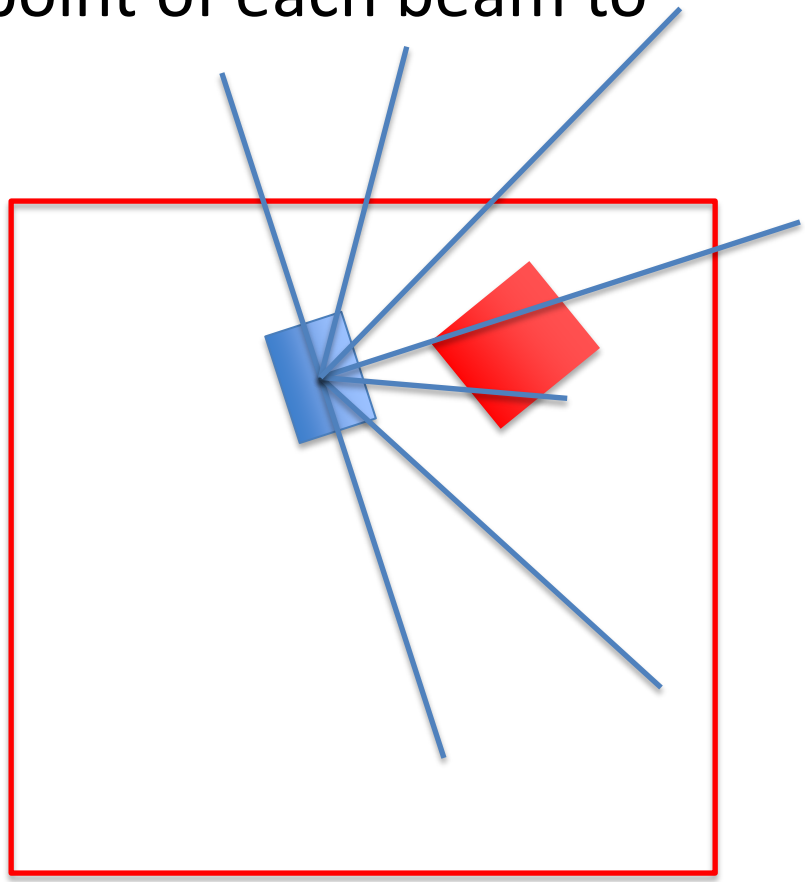
but how do
we measure?



Observation Model

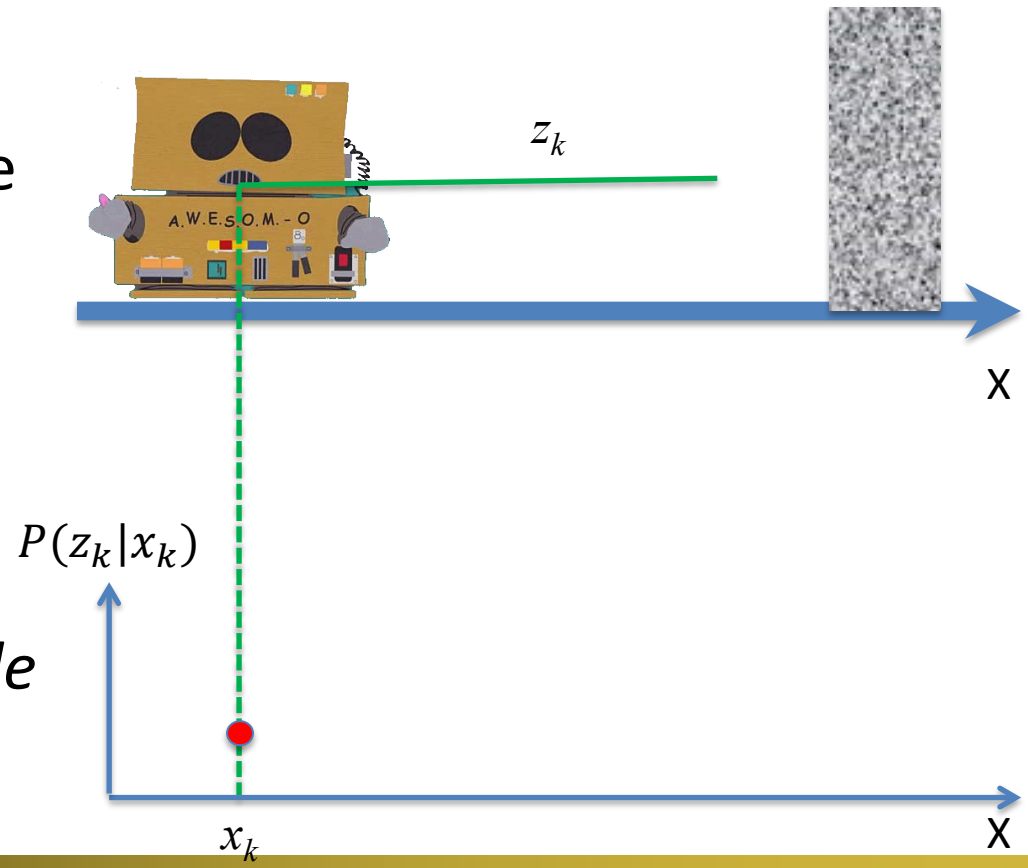
- Compute the distance of end-point of each beam to nearest obstacle





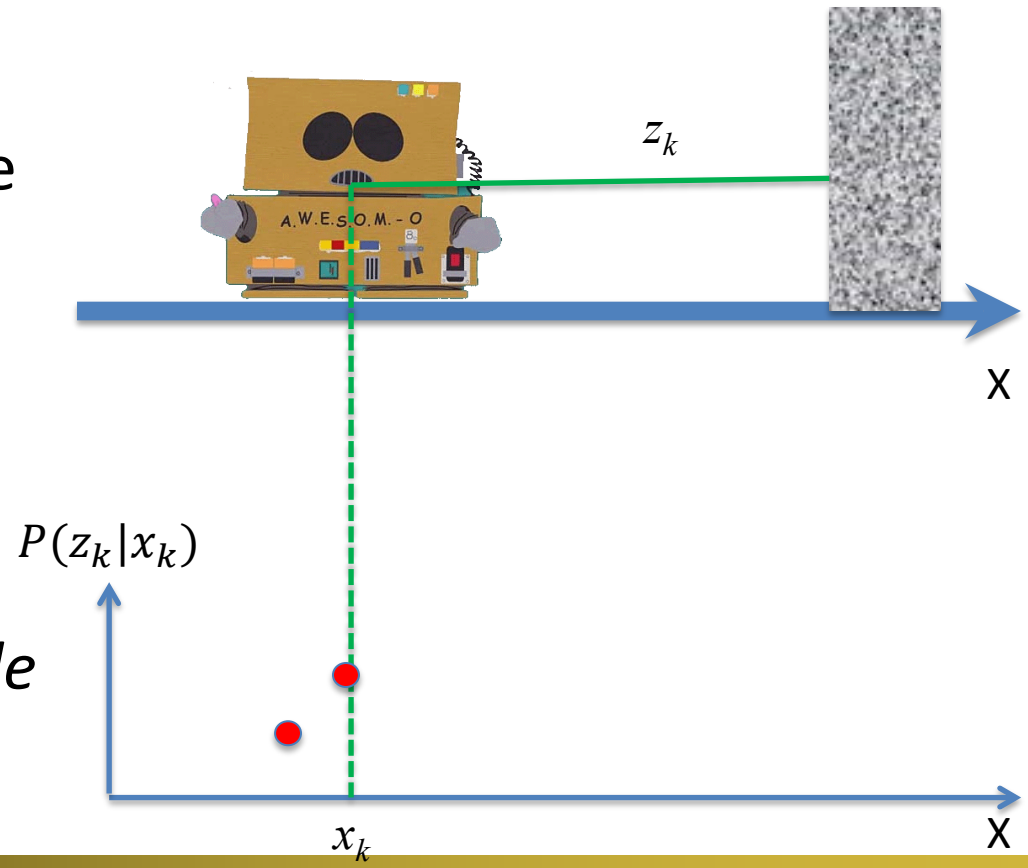
Observation Model

- How likely is the measurement z_k given a state x_k?
- The measurement z_k is what we get. So we have to stick to it
- However, not all states will likely produce that measurement
- We're not interested in finding the most likely state. *We want the whole distribution*



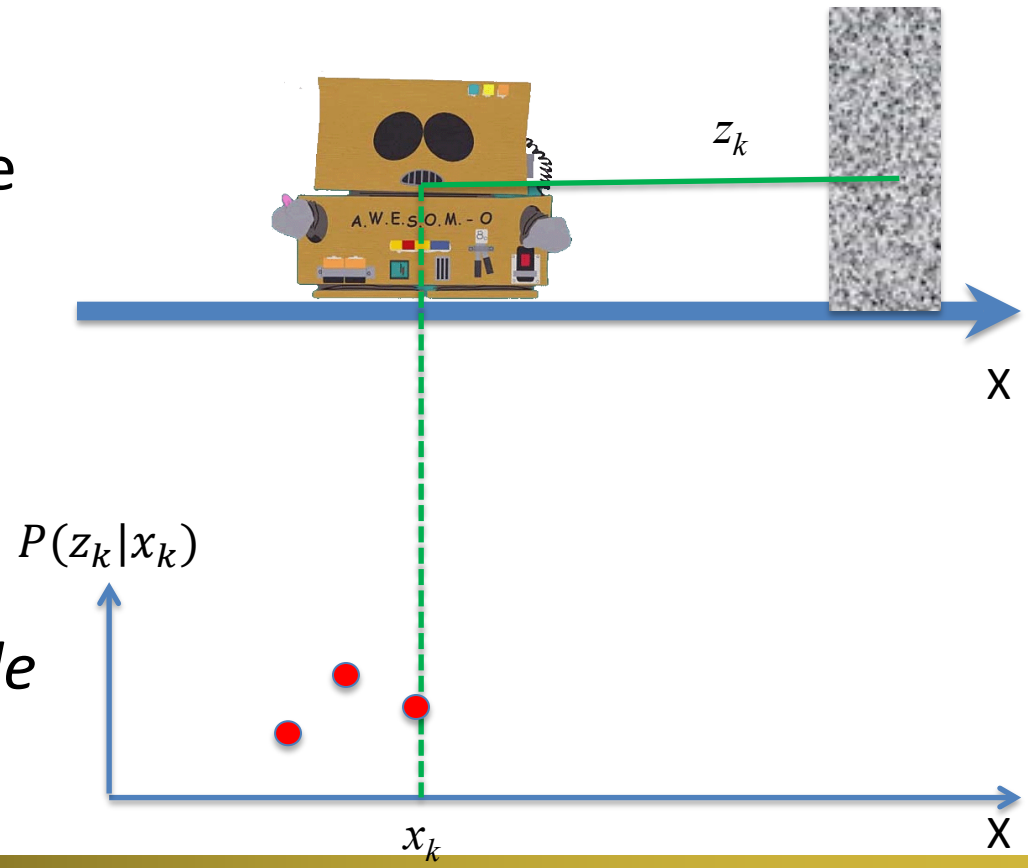
Observation Model

- How likely is the measurement z_k given a state x_k?
- The measurement z_k is what we get. So we have to stick by it
- However, not all states will likely produce the measurement
- We're not interested in finding the most likely state. *We want the whole distribution*



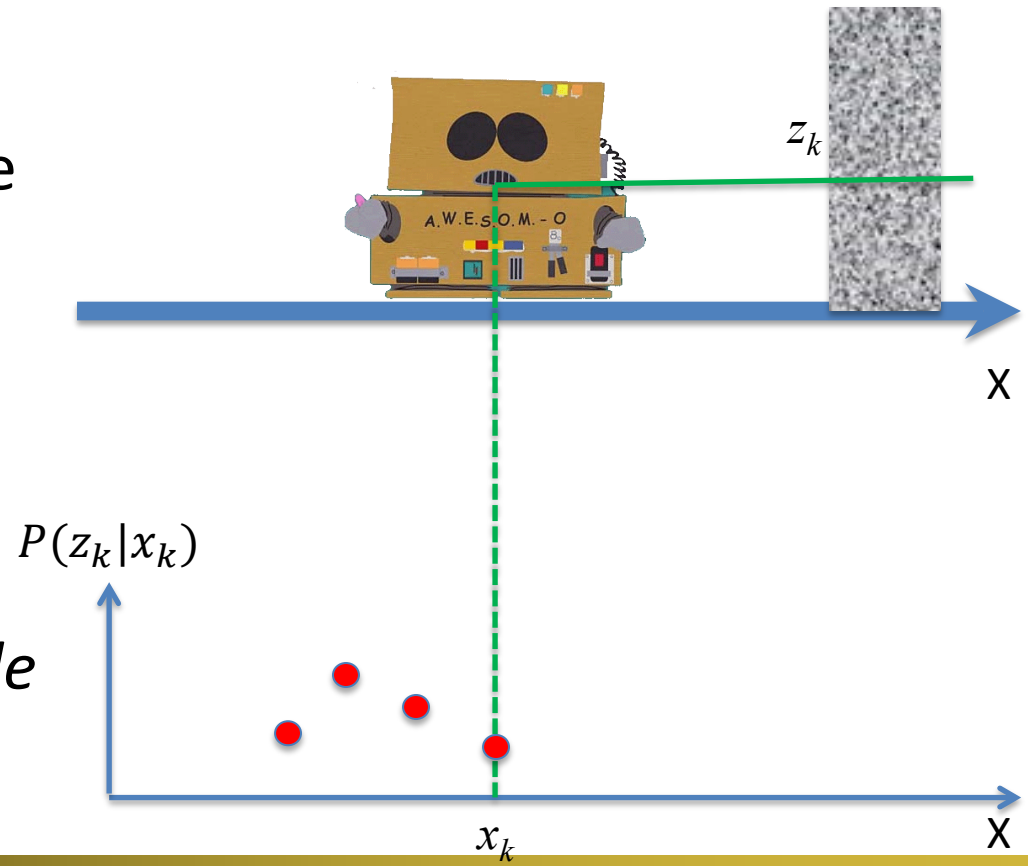
Observation Model

- How likely is the measurement z_k given a state x_k?
- The measurement z_k is what we get. So we have to stick by it
- However, not all states will likely produce the measurement
- We're not interested in finding the most likely state. *We want the whole distribution*



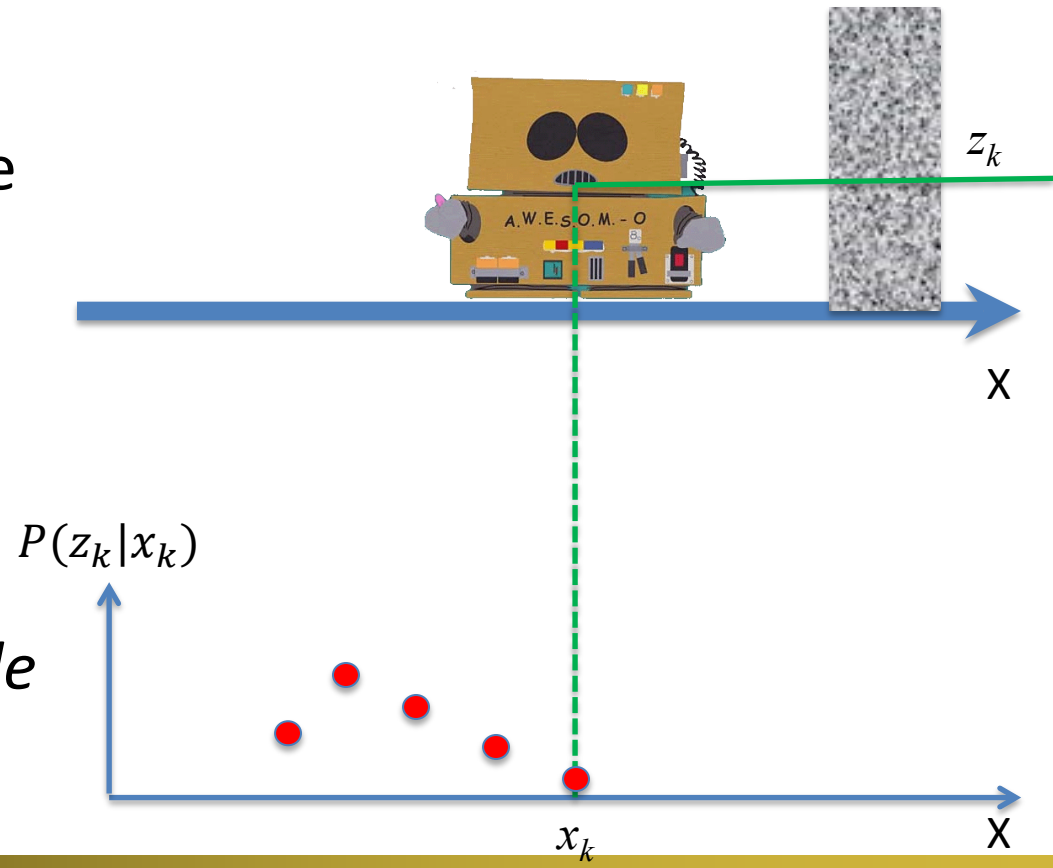
Observation Model

- How likely is the measurement z_k given a state x_k?
- The measurement z_k is what we get. So we have to stick by it
- However, not all states will likely produce the measurement
- We're not interested in finding the most likely state. *We want the whole distribution*



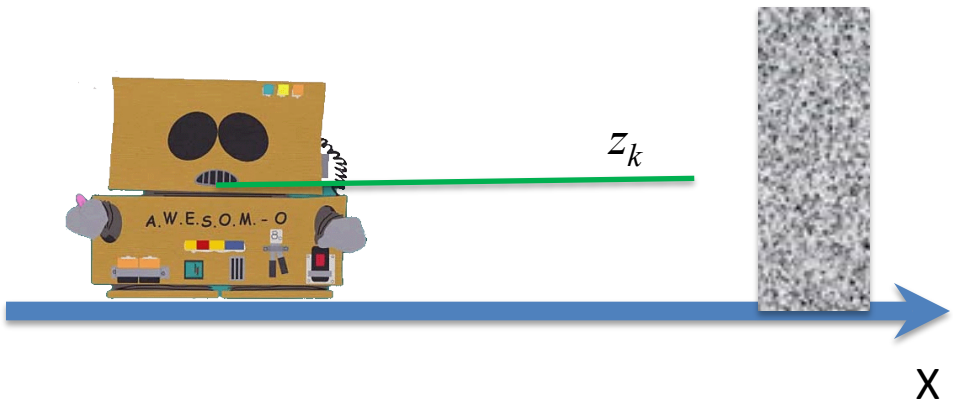
Observation Model

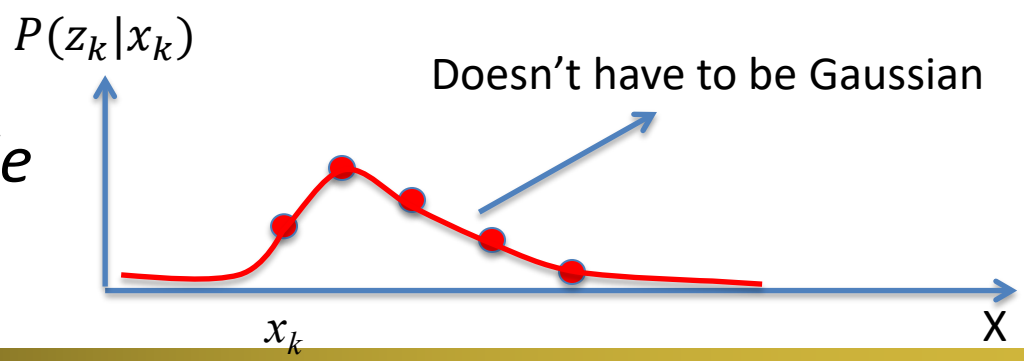
- How likely is the measurement z_k given a state x_k?
- The measurement z_k is what we get. So we have to stick by it
- However, not all states will likely produce the measurement
- We're not interested in finding the most likely state. *We want the whole distribution*



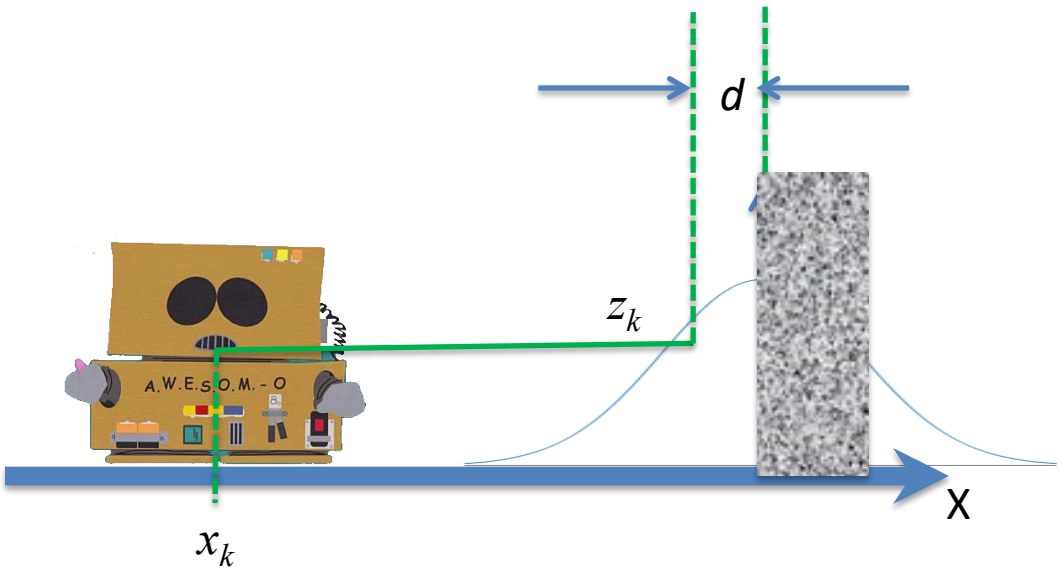
Observation Model

- How likely is the measurement z_k given a state x_k?
- The measurement z_k is what we get. So we have to stick by it
- However, not all states will likely produce the measurement
- We're not interested in finding the most likely state. *We want the whole distribution*



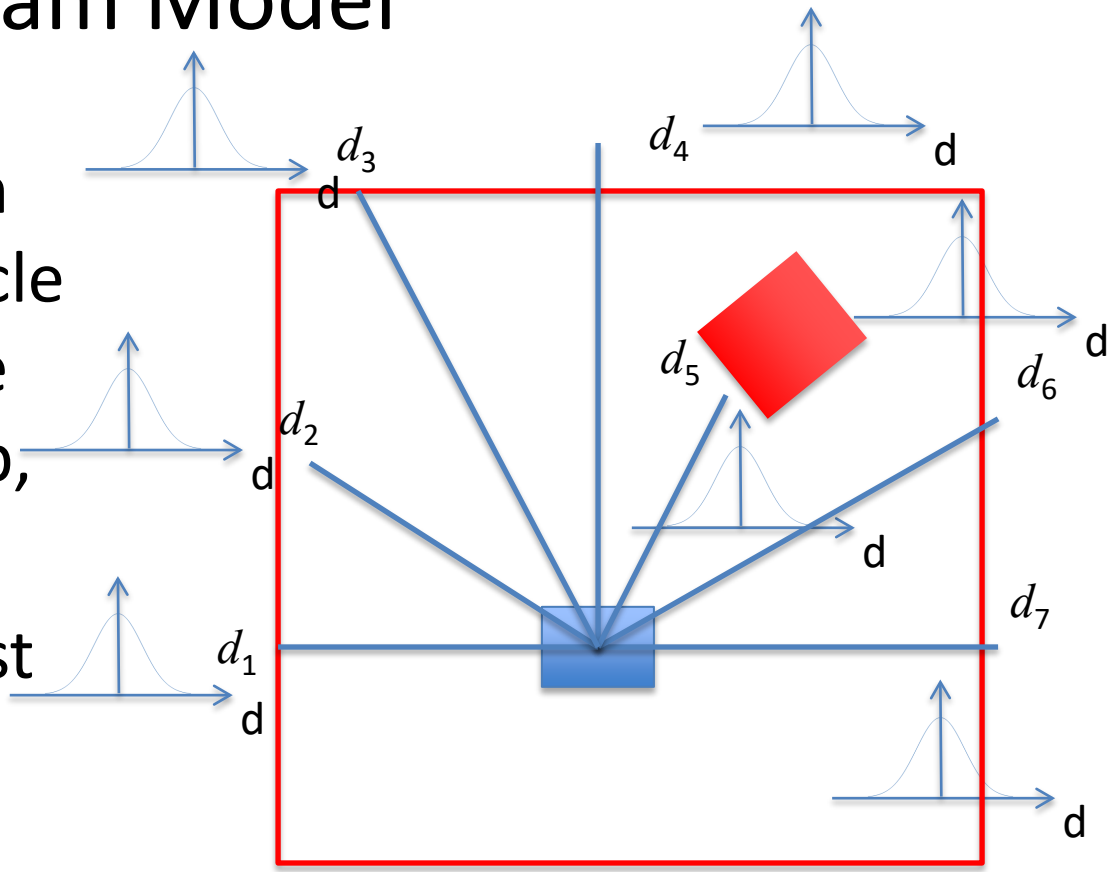


Observation Model
Beam Model



Observation Model
Beam Model

- Combine all the distances d_i between the measured obstacle and the real obstacle (we do have the map, or part thereof)
- But other things must be considered



Range Finder
Observation Model Beam Model

- For each laser beam we can have (given a map and a state)
  - **Correct range with noise**

True range Sensor noise

$$P_{\text{hit}}(z_{\boldsymbol{k}}|\boldsymbol{x}_k, m) = \begin{cases} \eta_{\text{hit}} N(z_k^*, \sigma_{hit}) & \text{if } 0 \le z_k \le z_{\max} \\ 0 \text{ otherwise} \end{cases}$$

  - **Unexpected object**

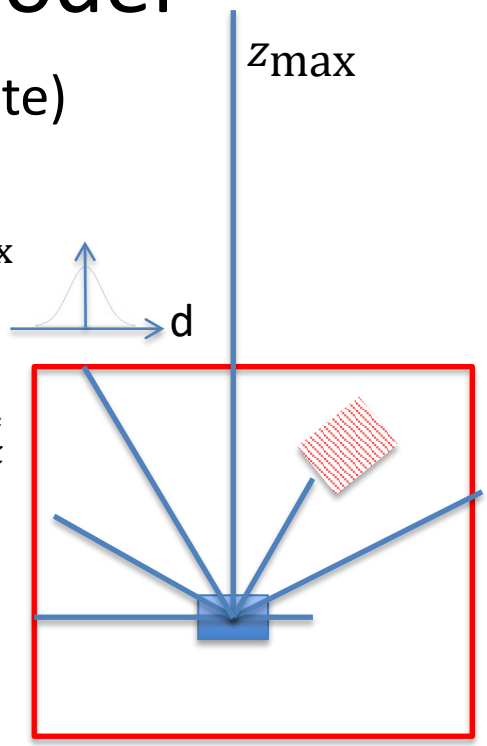
$$P_{\text{short}}(z_{\boldsymbol{k}}|\boldsymbol{x}_k, m) = \begin{cases} \eta_{\text{short}} \lambda e^{-\lambda z_k} & \text{if } 0 \le z_k \le z_k^* \\ 0 \text{ otherwise} \end{cases}$$

  - **Failures**

$$P_{\max}(z_{\boldsymbol{k}}|\boldsymbol{x}_k, m) = I(z_k = z_{\max}) \begin{cases} 1 \text{ if } z_k = z_{\max} \\ 0 \text{ otherwise} \end{cases}$$

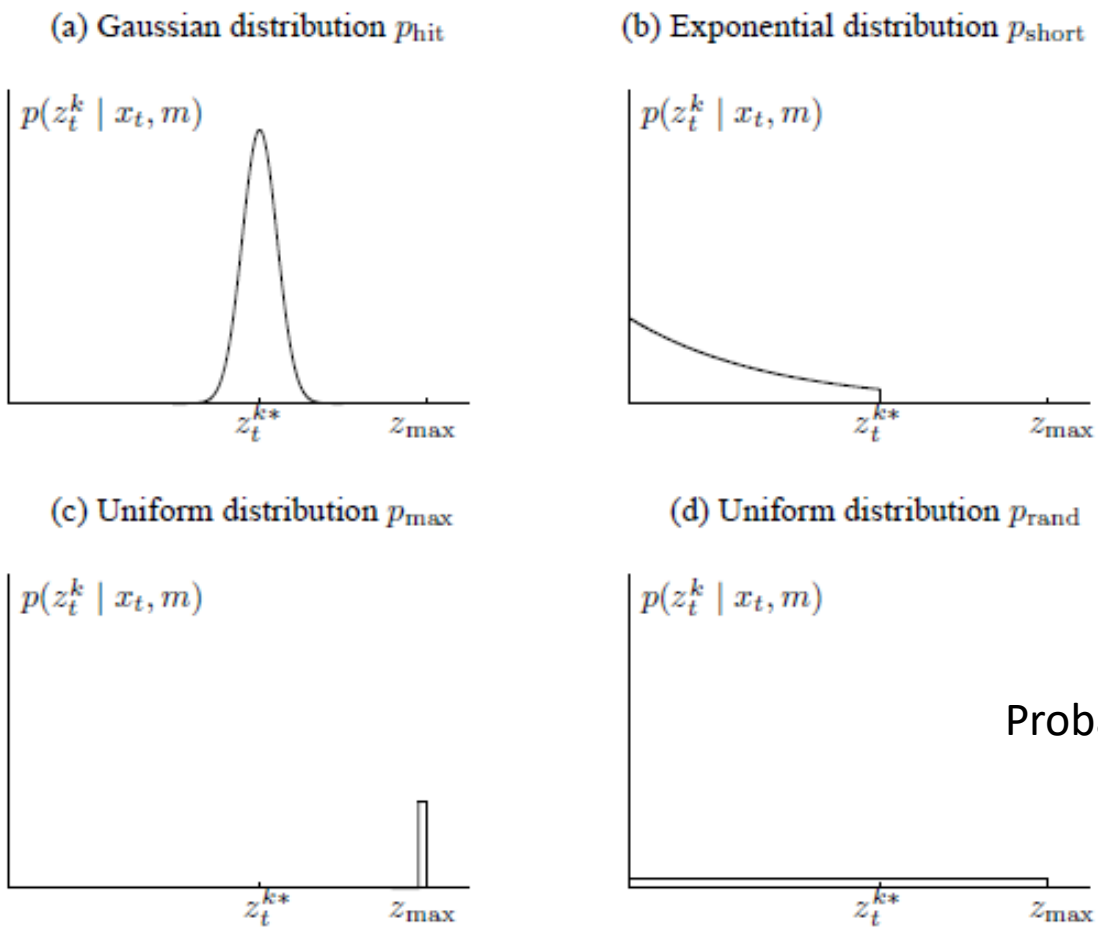
  - **Random measurement**

$$P_{\text{rand}}(z_{\boldsymbol{k}}|\boldsymbol{x}_k, m) = \begin{cases} \dfrac{1}{z_{\max}} & if\ 0 \le z_k \le z_{\max} \\ 0 \text{ otherwise} \end{cases}$$



$P(z_k|x_k, m)$

Range Finder
Observation Model Beam Model



Probabilistic Robotic

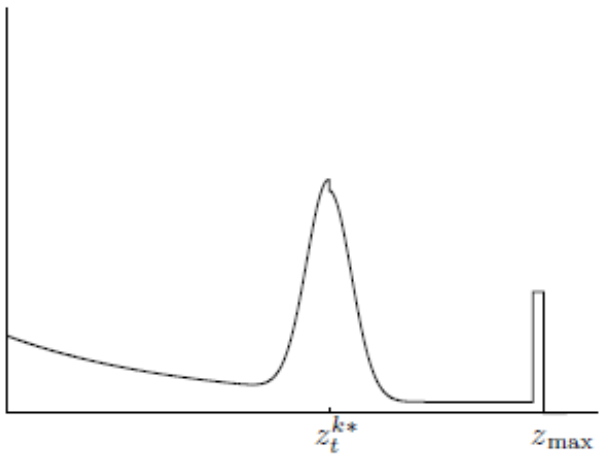
Range Finder
Observation Model Beam Model

- Mixing all these cases together we get

$$P(z_k|\boldsymbol{x}_k, m) = \begin{bmatrix} w_{\text{hit}} \\ w_{\text{short}} \\ w_{\text{max}} \\ w_{\text{rand}} \end{bmatrix}^T \begin{bmatrix} P_{\text{hit}}(z_k|\boldsymbol{x}_k, m) \\ P_{\text{short}}(z_k|\boldsymbol{x}_k, m) \\ P_{\text{max}}(z_k|\boldsymbol{x}_k, m) \\ P_{\text{rand}}(z_k|\boldsymbol{x}_k, m) \end{bmatrix}$$

where the weights are parameters

$$w_{\text{hit}} + w_{\text{short}} + w_{\text{max}} + w_{\text{rand}} = 1$$



Probabilistic Robotic

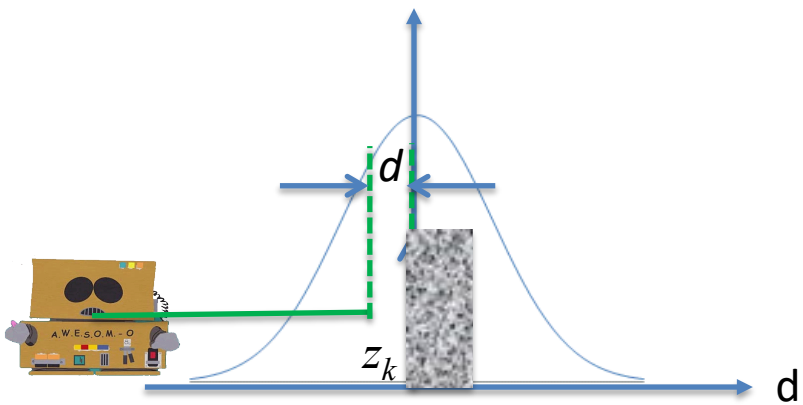
Constants

$$\eta_{\text{hit}} = \left(\int_0^{z\text{max}} N(z_k^*, \sigma^2) dz_k \right)^{-1}$$

$$\eta_{\text{short}} = \frac{1}{1 - e^{-\lambda_{short} z_k^*}}$$

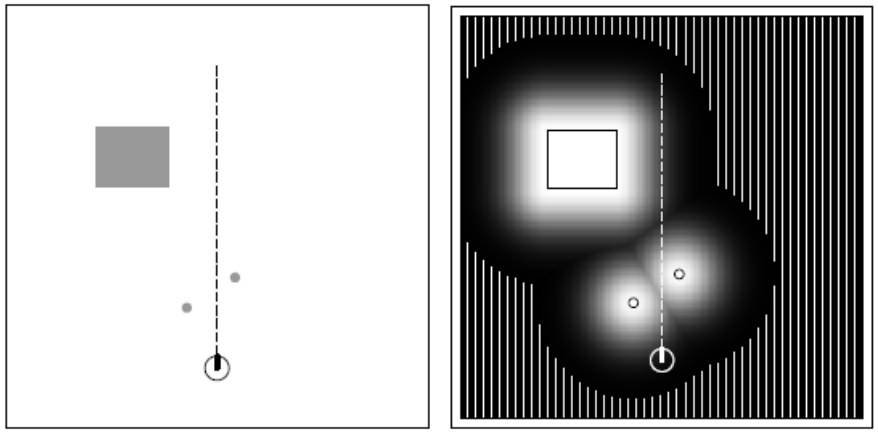
Range Finder
Likelihood Field Model

$$P_{\text{hit}}(z_k | \boldsymbol{x}_k, m) = N\,(0, \sigma_{hit})$$

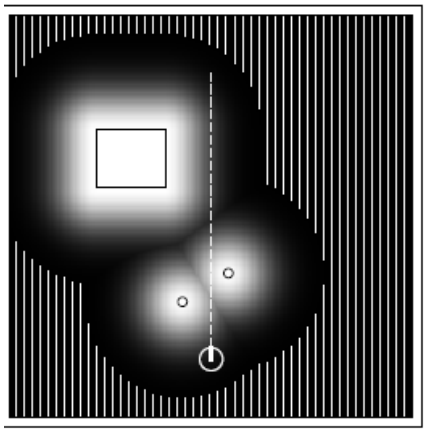


d: distance between endpoint and nearest obstacle

Probabilistic Robotic



Range Finder
Likelihood Field Model



Algorithm likelihood_field_range_finder_model(z_t, x_t, m):

$q = 1$

for all k *do*

  if $z_t^k \neq z_{\max}$

    $x_{z_t^k} = x + x_{k,\text{sens}} \cos\theta - y_{k,\text{sens}} \sin\theta + z_t^k \cos(\theta + \theta_{k,\text{sens}})$

    $y_{z_t^k} = y + y_{k,\text{sens}} \cos\theta + x_{k,\text{sens}} \sin\theta + z_t^k \sin(\theta + \theta_{k,\text{sens}})$

    $dist = \min_{x',y'} \left\{ \sqrt{(x_{z_t^k} - x')^2 + (y_{z_t^k} - y')^2} \,\middle|\, \langle x', y' \rangle \text{ occupied in } m \right\}$

    $q = q \cdot \left(z_{\text{hit}} \cdot \mathbf{prob}(dist, \sigma_{\text{hit}}) + \frac{z_{\text{random}}}{z_{\max}} \right)$

return q

Range Finder
Likelihood Field Model

- Precompute the shortest distance to obstacle
  - Distance transformation
  - Brushfire algorithm

-	-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	*	*	*	*	-	-	-	-	-
-	-	-	-	-	*	-	-	-	-	-	-	-	-
-	-	-	-	-	*	-	-	-	-	-	-	-	-
-	-	-	-	-	*	*	*	-	-	-	-	-	-
-	-	-	-	-	*	-	-	-	-	-	-	-	-
-	-	-	-	-	*	-	-	-	-	-	-	-	-
-	-	-	-	-	*	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-	-

6	6	5	4	4	4	4	4	4	4	4	5	6	6
6	5	4	4	3	3	3	3	3	3	4	4	5	6
5	4	4	3	2	2	2	2	2	2	3	4	4	5
5	4	3	2	1	1	1	1	1	1	2	3	4	5
5	4	3	2	1	0	0	0	0	1	2	3	4	5
5	4	3	2	1	0	1	1	1	1	2	3	4	5
5	4	3	2	1	0	1	1	1	2	3	4	4	5
5	4	3	2	1	0	0	0	1	2	3	4	5	6
5	4	3	2	1	0	1	1	1	2	3	4	5	6
5	4	3	2	1	0	1	2	2	3	4	4	5	6
5	4	3	2	1	0	1	2	3	4	4	5	6	7
5	4	3	2	1	1	1	2	3	4	5	6	6	7
5	4	4	3	2	2	2	3	4	4	5	6	7	8
6	5	4	4	3	3	3	4	4	5	6	7	8	8
6	6	5	4	4	4	4	4	5	6	6	7	8	9

Bayes Filtering

$$P(\boldsymbol{x}_k|\boldsymbol{u}_{0:k-1},\boldsymbol{z}_{1:k})$$

$$= \eta_k P(\boldsymbol{z}_k|\boldsymbol{x}_k) \int_{\boldsymbol{x}_{k-1}} P(\boldsymbol{x}_k|\boldsymbol{u}_{k-1},\boldsymbol{x}_{k-1}) P(\boldsymbol{x}_{k-1}|\boldsymbol{u}_{0:k-2},\boldsymbol{z}_{1:k-1})$$

Predict Motion (Prior Distribution)

- Suppose we have $P(\boldsymbol{x}_k)$
- We have $P(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k, \boldsymbol{u}_k)$
- Put together

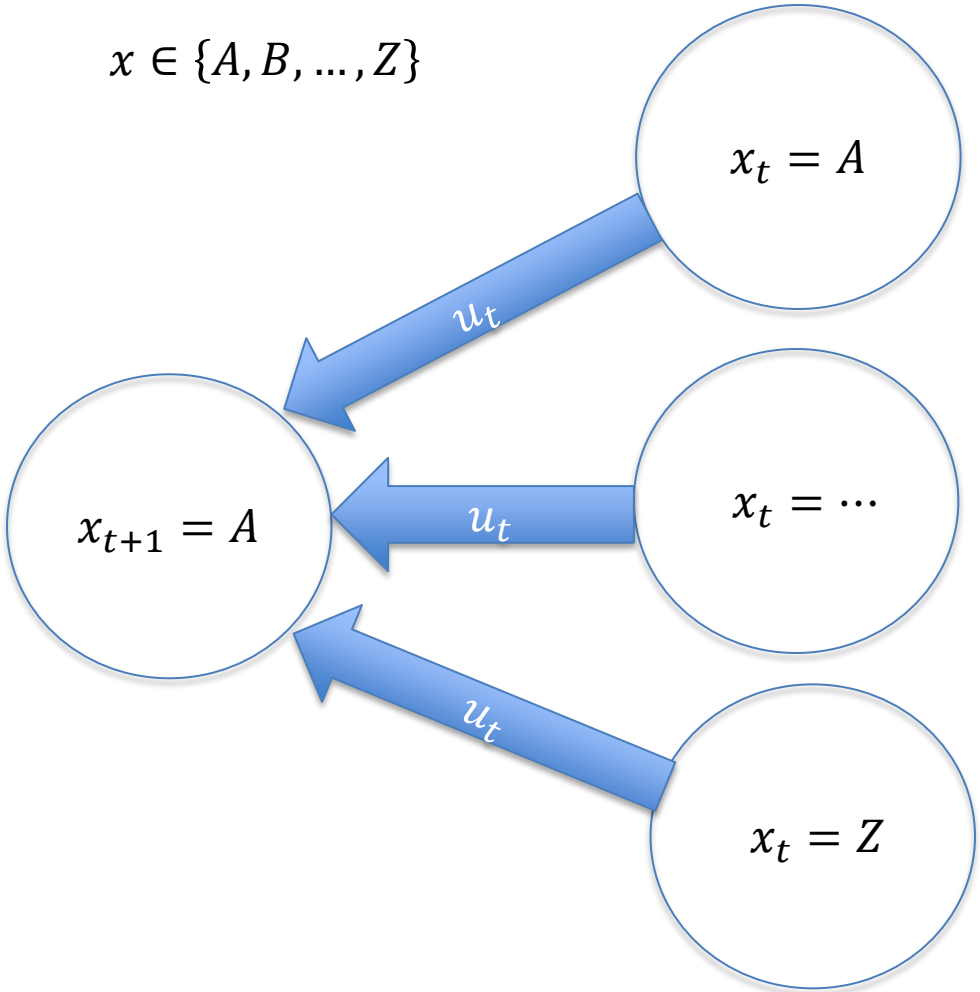
$$P(\boldsymbol{x}_{k+1}) = \int_{\boldsymbol{x}_k} P(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k, \boldsymbol{u}_k) P(\boldsymbol{x}_k) d\boldsymbol{x}_k$$

What is the probability distribution for $\boldsymbol{x}_{k+1}$ given the command $\boldsymbol{u}_k$ and all the previous states $\boldsymbol{x}_k$?

System Model

$x \in \{A, B, \ldots, Z\}$

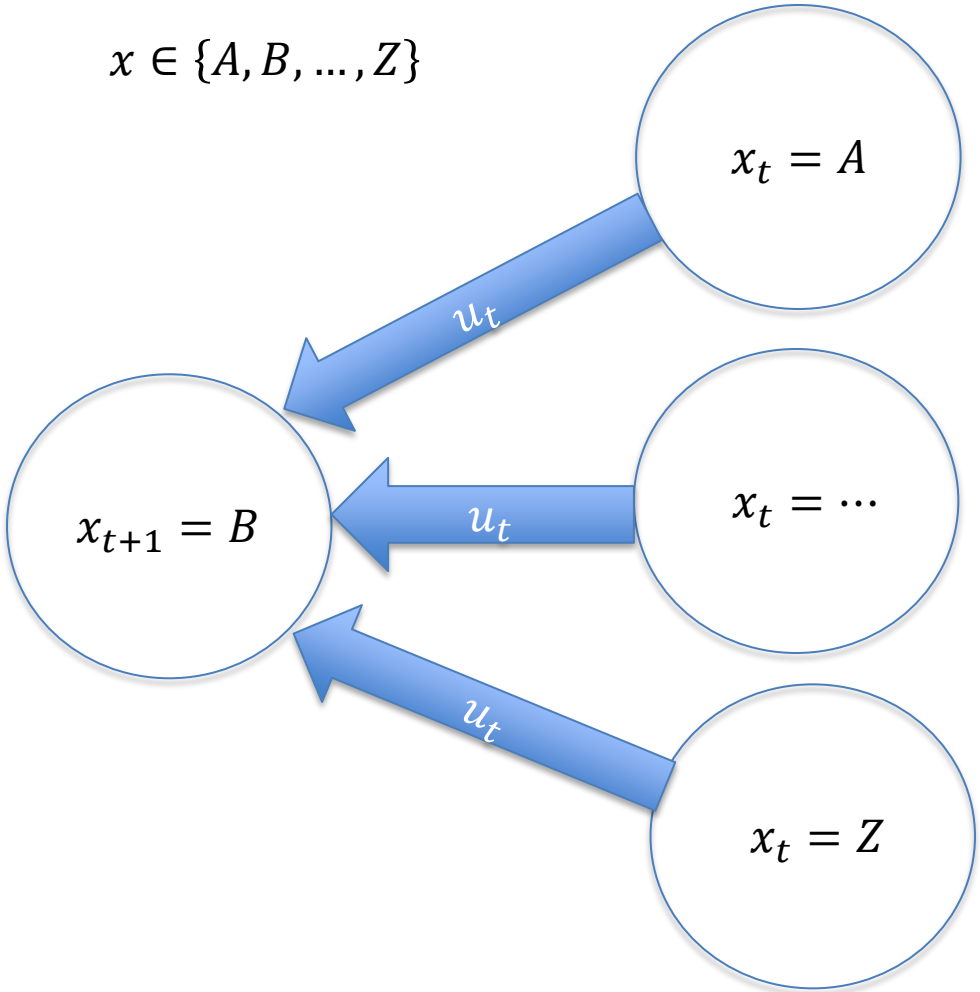
$$P(x_{t+1} = A \mid x_t = A, u_t)$$
$$P(x_{t+1} = A \mid x_t = B, u_t)$$
$$\ldots$$
$$P(x_{t+1} = A \mid x_t = Z, u_t)$$



System Model

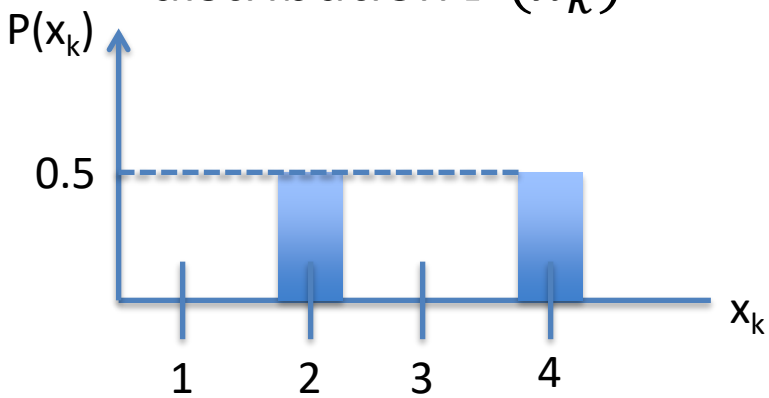
$x \in \{A, B, \ldots, Z\}$

$$P(x_{t+1} = B \mid x_t = A, u_t)$$
$$P(x_{t+1} = B \mid x_t = B, u_t)$$
$$\ldots$$
$$P(x_{t+1} = B \mid x_t = Z, u_t)$$



System Model

Prior probability distribution $P(x_k)$



State space X = { 1, 2, 3, 4}

$P(x_{k+1} \mid x_k, u_k)$	$X_{k+1}=1$	$X_{k+1}=2$	$X_{k+1}=3$	$X_{k+1}=4$
$X_k=1$	0.25	0.5	0.25	0
$X_k=2$	0	0.25	0.5	0.25
$X_k=3$	0	0	0.25	0.75
$X_k=4$	0	0	0	1

Transition matrix: The probability $P(j|i)$ of moving from i to j is given by $P_{i,j}$. Each row must sum to 1.

Compute $P(x_{k+1}) = \int_{x_k} P(x_{k+1}|x_k, u_k) P(x_k) dx_k$

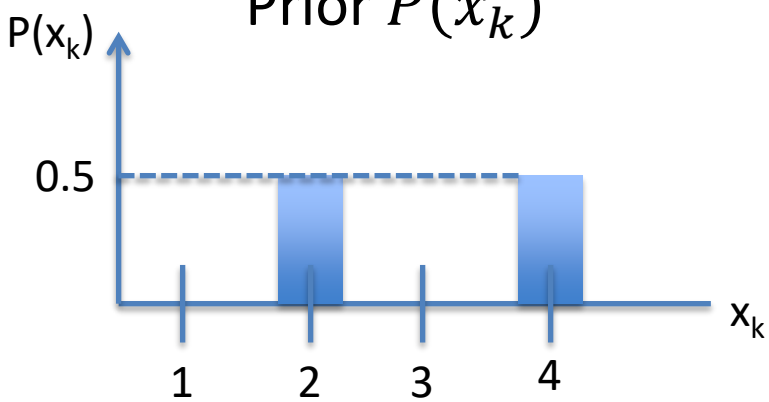
System Model

State space X = { 1, 2, 3, 4}

$P(x_{k+1} \mid x_k, u_k)$	$X_{k+1}=1$	$X_{k+1}=2$	$X_{k+1}=3$	$X_{k+1}=4$
$X_k=1$	0.25	0.5	0.25	0
$X_k=2$	0	0.25	0.5	0.25
$X_k=3$	0	0	0.25	0.75
$X_k=4$	0	0	0	1

$$P(x_{k+1}=1) = \sum_{x_k \in X} P(x_{k+1}=1 \mid x_k, u_k) P(x_k)$$

Prior $P(x_k)$



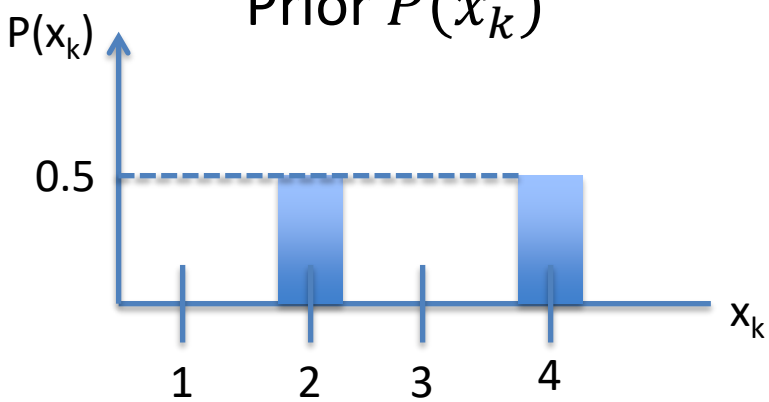
System Model

State space X = { 1, 2, 3, 4}

$P(x_{k+1} \mid x_k, u_k)$	$X_{k+1}=1$	$X_{k+1}=2$	$X_{k+1}=3$	$X_{k+1}=4$
$X_k=1$	0.25	0.5	0.25	0
$X_k=2$	0	0.25	0.5	0.25
$X_k=3$	0	0	0.25	0.75
$X_k=4$	0	0	0	1

$$P(x_{k+1}=2)=\sum_{x_k \in X} P(x_{k+1}=2 \mid x_k, u_k) P(x_k)$$

Prior $P(x_k)$



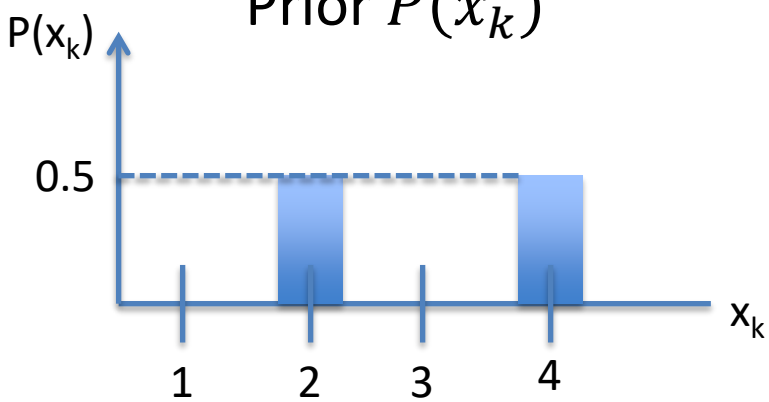
System Model

State space X = { 1, 2, 3, 4}

$P(x_{k+1} \mid x_k, u_k)$	$X_{k+1}=1$	$X_{k+1}=2$	$X_{k+1}=3$	$X_{k+1}=4$
$X_k=1$	0.25	0.5	0.25	0
$X_k=2$	0	0.25	0.5	0.25
$X_k=3$	0	0	0.25	0.75
$X_k=4$	0	0	0	1

$$P(x_{k+1}=3) = \sum_{x_k \in X} P(x_{k+1}=3 \mid x_k, u_k) P(x_k)$$

Prior $P(x_k)$



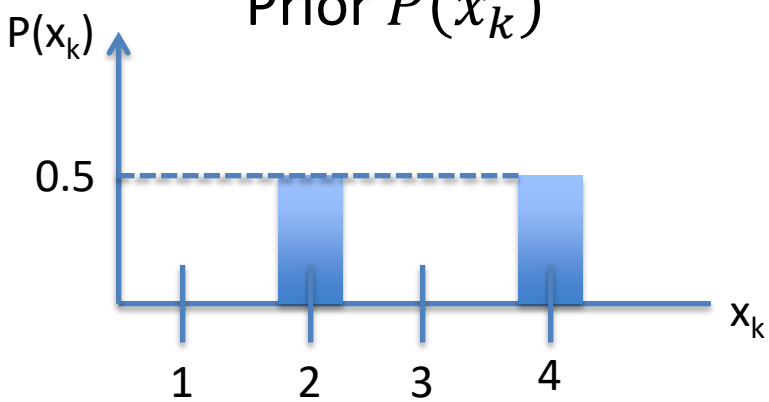
System Model

State space X = { 1, 2, 3, 4}

$P(x_{k+1} \mid x_k, u_k)$	$X_{k+1}=1$	$X_{k+1}=2$	$X_{k+1}=3$	$X_{k+1}=4$
$X_k=1$	0.25	0.5	0.25	0
$X_k=2$	0	0.25	0.5	0.25
$X_k=3$	0	0	0.25	0.75
$X_k=4$	0	0	0	1

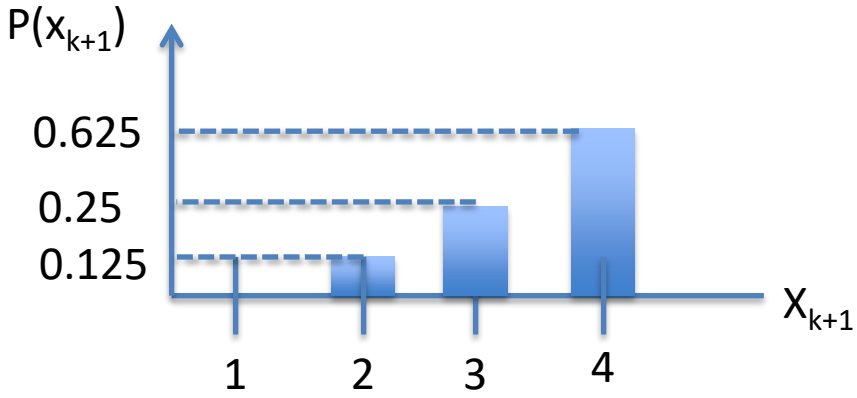
$$P(x_{k+1}=4) = \sum P(x_{k+1}=4 \mid x_k, u_k) P(x_k)$$

Prior $P(x_k)$

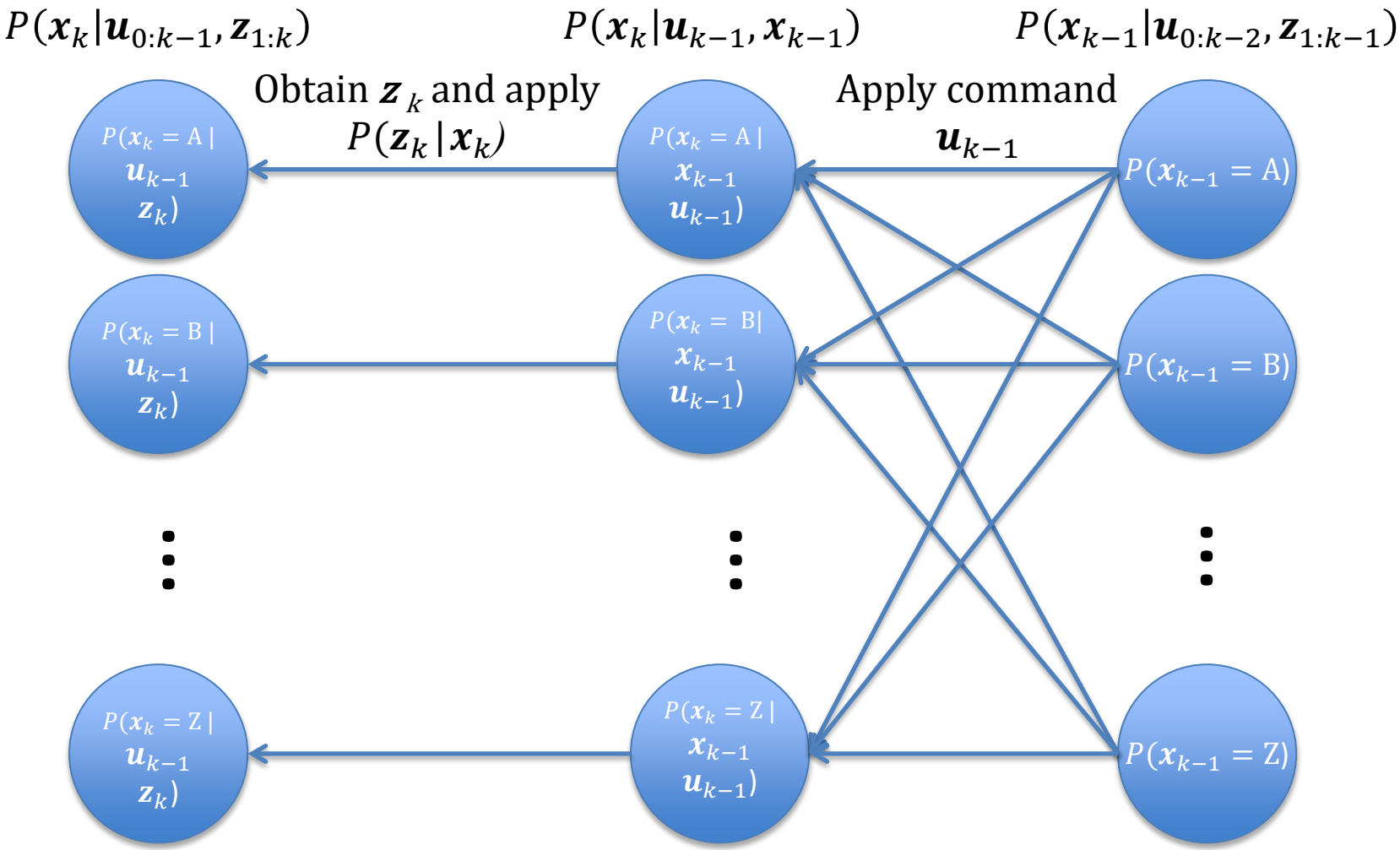


System Model

$$P(x_{k+1}) = \int P(x_{k+1} \mid x_k, u_k) P(x_k) dx_k$$



Bayes Filter Recap



Bayes Filter Recap

$$P(\boldsymbol{x}_k|\boldsymbol{u}_{0:k-1},\boldsymbol{z}_{1:k}) = \eta_k P(\boldsymbol{z}_k|\boldsymbol{x}_k) \int_{\boldsymbol{x}_{k-1}} P(\boldsymbol{x}_k|\boldsymbol{u}_{k-1},\boldsymbol{x}_{k-1}) P(\boldsymbol{x}_{k-1}|\boldsymbol{u}_{0:k-2},\boldsymbol{z}_{1:k-1}) d\boldsymbol{x}_{k-1}$$

- Given a measurement $\boldsymbol{z}_k$, compute the probability that $\boldsymbol{z}_k$ was generated from the robot in state $\boldsymbol{x}_k$

- Prior distribution (recursive)
  - The probability of all previous states
- State transition:
  - Given a prior distribution over ***all*** states $\boldsymbol{x}_{k-1}$ and an action $\boldsymbol{u}_{k-1}$, compute the probability of the robot transitioning to state $\boldsymbol{x}_k$

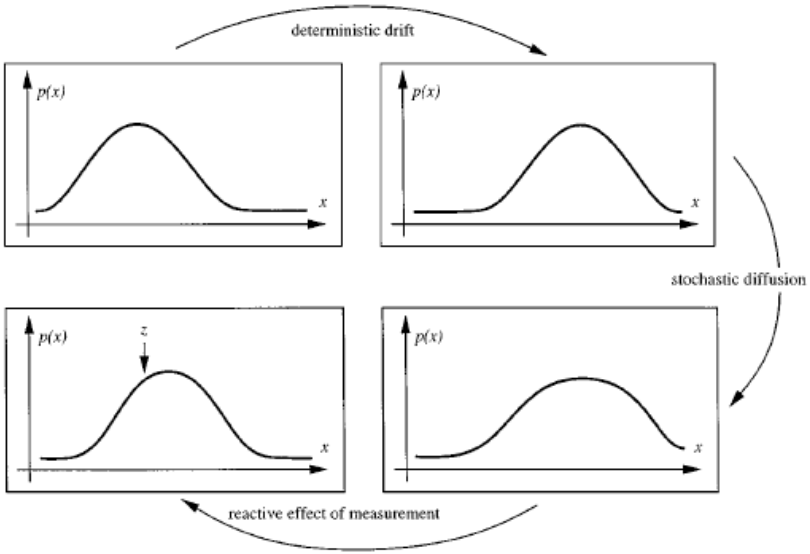
Discrete Bayes Filter Algorithm

Algorithm **Discrete_Bayes_filter**($u_{0:k-1}$, $y_{1:k}$, $P(x_0)$)

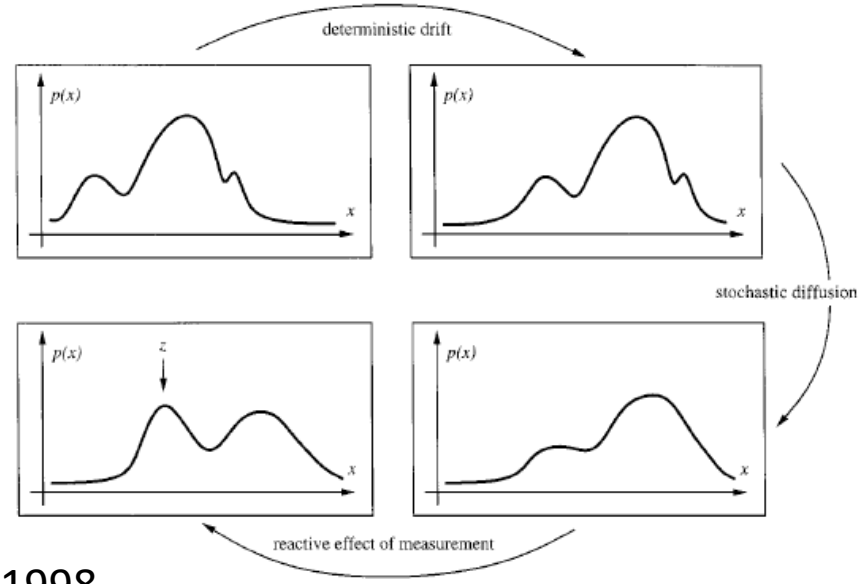
1.	$P(x) = P(x_0)$	(if you don't know: uniform distribution)
2.	**for** i=1:k	
3.	**for all** states $x \in X$	
4.	$P'(x) = \sum_{x' \in X} P(x \mid u, x') P(x')$	Prediction given prior dist. and command
5.	**end for**	
6.	$\eta = 0$	Normalization constant
7.	**for all** states $x \in X$	
8.	$P(x) = P(z \mid x) P'(x)$	Update using measurement
9.	$\eta = \eta + P(x)$	
10.	**end for**	
11.	**for all** states $x \in X$	
12.	$P(x) = P(x) / \eta$	Normalize to 1
13.	**end for**	
14.	**end for**	

Kalman vs Bayes

Kalman Filter

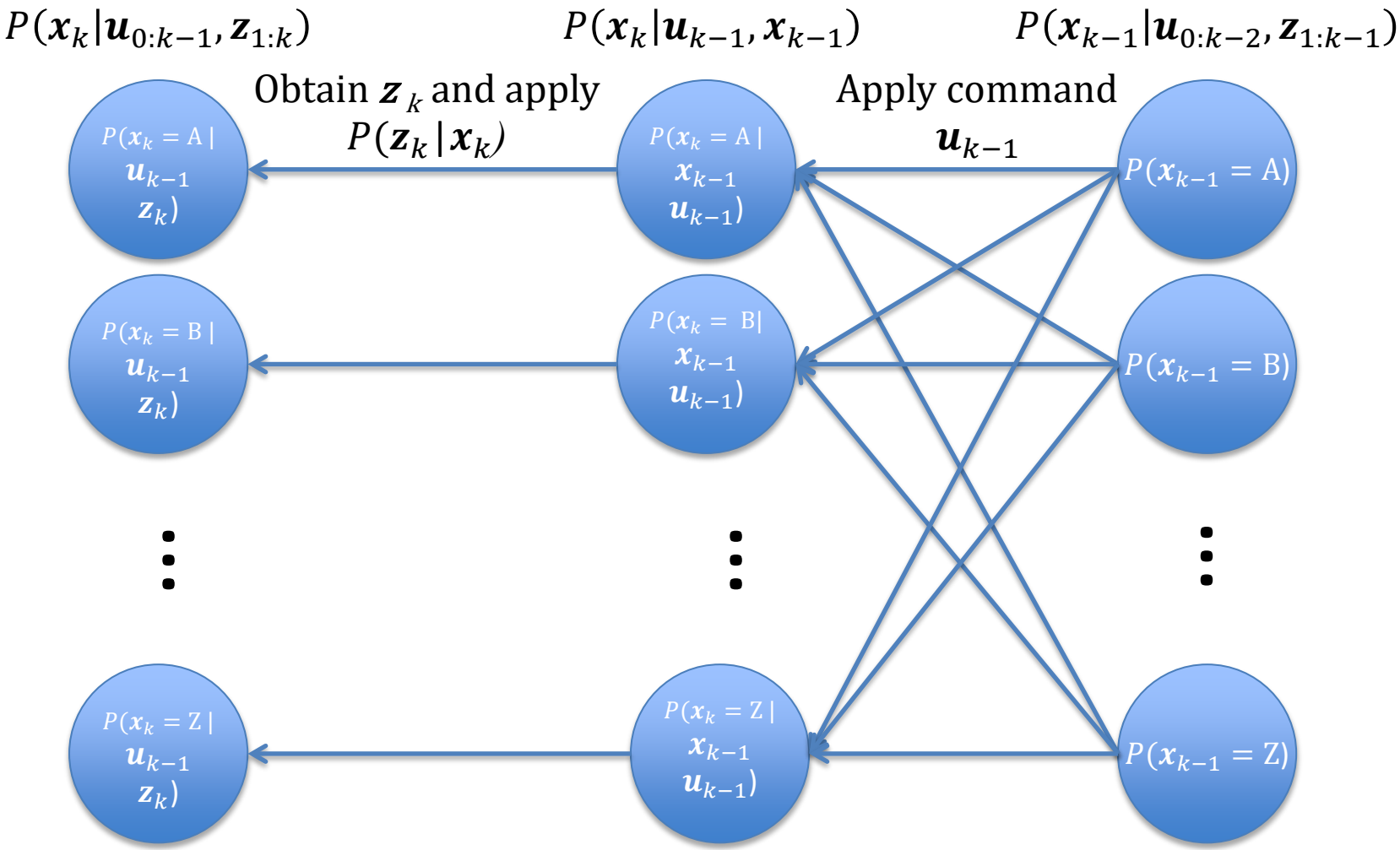


Bayes Filter



Isard 1998

Bayes Filter Recap

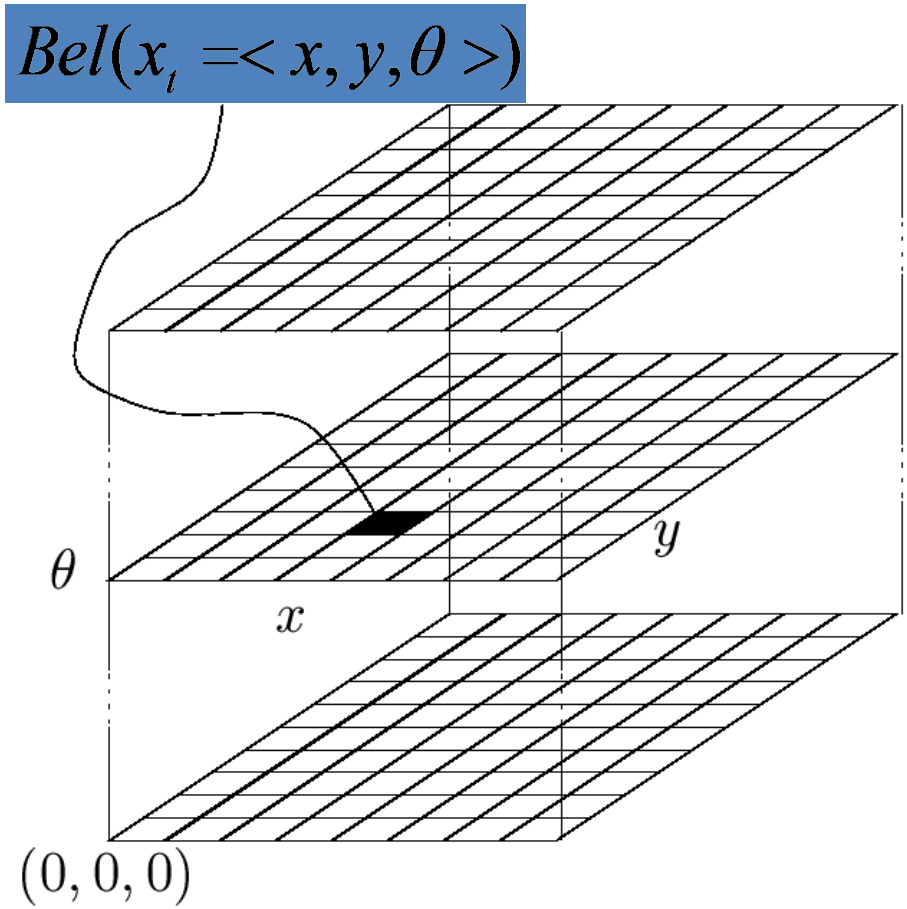


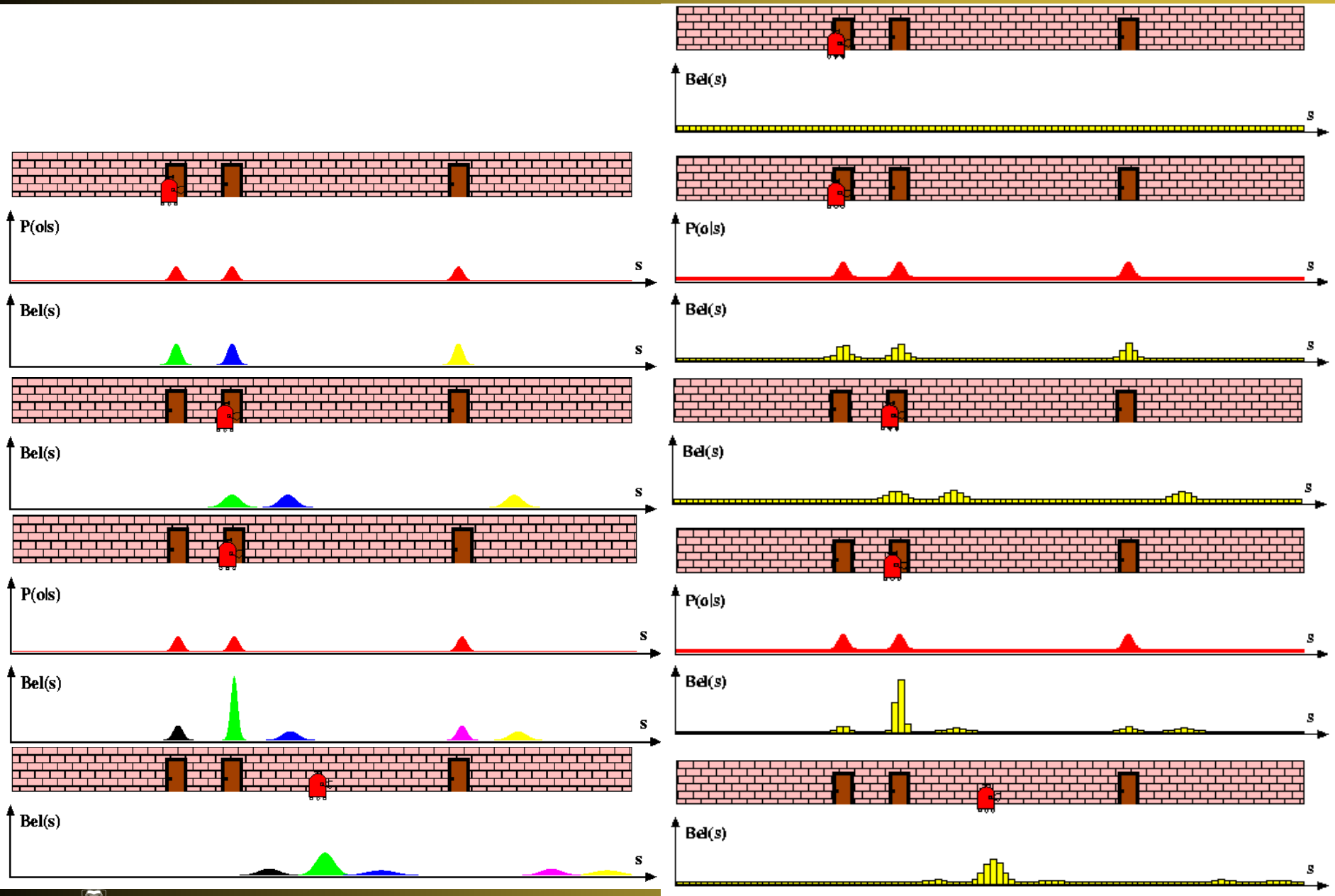
Piecewise Constant Representation

- Updating from observations is a simple multiplication of prior probability by likelihood of observation
- Updating using dynamical model is simply a discrete convolution (blurring) of the prior by the driving noise of the planned motion

Piecewise Constant Representation (Mobile Robot)

Position of a mobile robot: (x, y, θ)



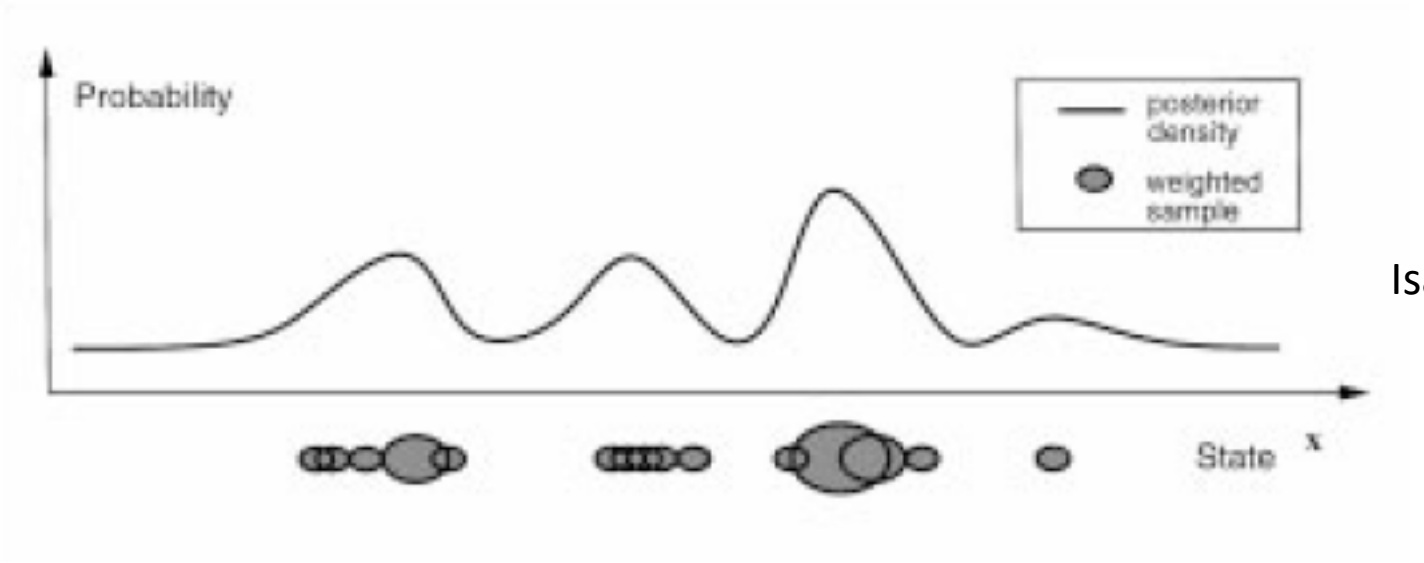

P(o|s)
Bel(s)
s

Particle Filter

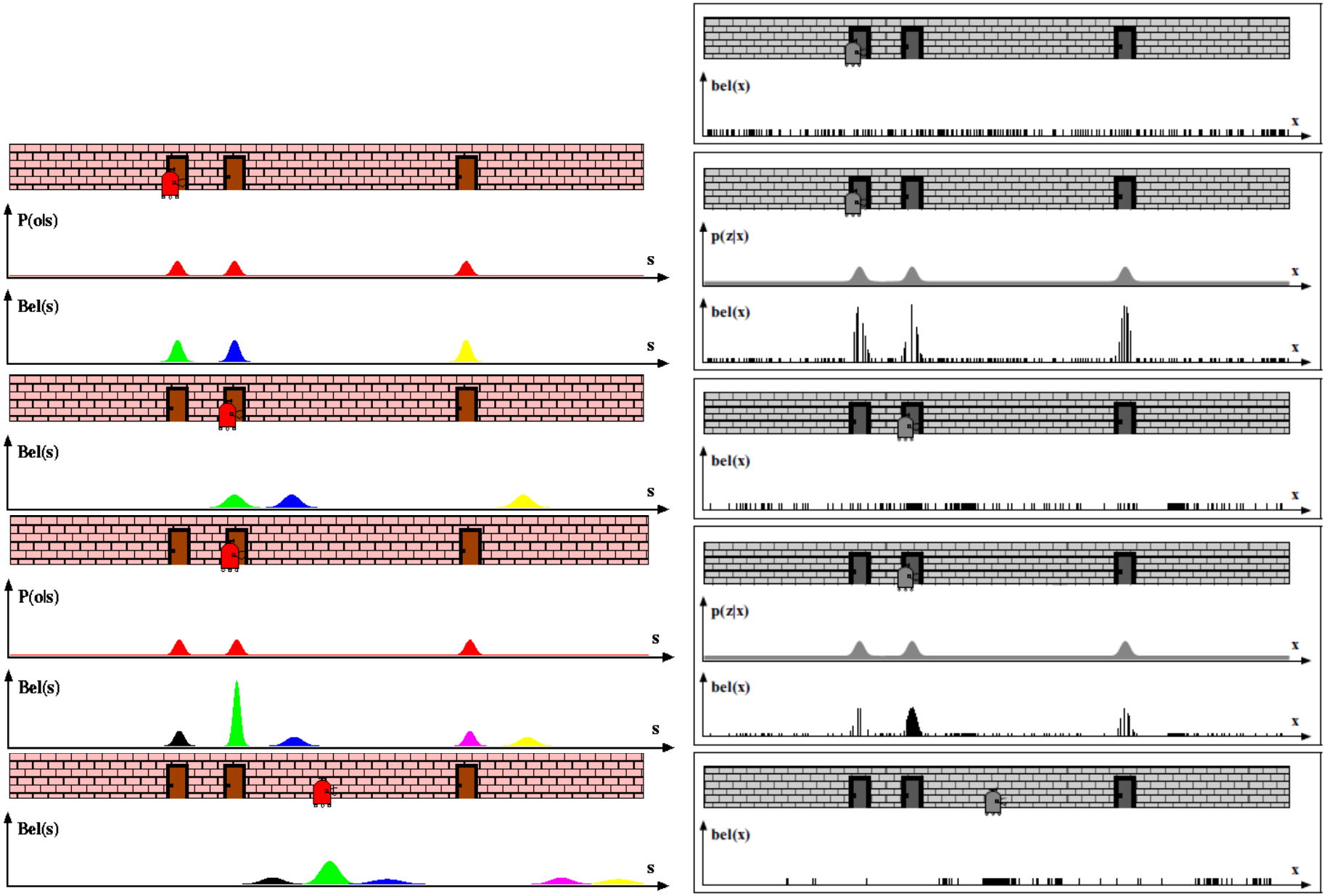
- Computing $P(\boldsymbol{z}_k|\boldsymbol{x}_k, m)$ and $P(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k, \boldsymbol{u}_k)$ is not easy
  - In practice, it is never directly computable
  - Need to propagate an entire conditional distribution, not just one state like we did with Kalman,
- Represent probability distribution by random samples
- Estimation of non-Gaussian, nonlinear processes
- Monte Carlo filter, Survival of the fittest, Condensation, Bootstrap filter, Particle filter
  - Filtering: [Rubin, 88], [Gordon et al., 93], [Kitagawa 96]
  - ***Computer vision***: [Isard and Blake 96, 98]
  - Dynamic Bayesian Networks: [Kanazawa et al., 95]

Particles/Samples

- Given a distribution $P(\boldsymbol{x})$, randomly sample the distribution N times
- Assign a weight w to each sample according to the probability
- Ensure that $\sum_{i=1}^{N} w_i = 1$



Isard 1998


P(ols)
s
Bel(s)
s
Bel(s)
s
P(ols)
s
Bel(s)
s
Bel(s)
s
bel(x)
x
p(z|x)
x
bel(x)
x
bel(x)
x
p(z|x)
x
bel(x)
x
bel(x)
x

Particle Filter Algorithm

Algorithm **Particle_filter**($u_{0:k-1}$, $y_{1:k}$, $P(x_0)$, set of N samples $\mathcal{M} = \{x_j, w_j\}$)

```
for i=1:k
    for j=1:N
        compute a new state x by sampling according to P( x | u_{i-1}, x_j )
        x_j = x
    end
    η=0
    for j=1:N
        w_j = P( z_i | x_j )
        η += w_j
    end
    for j=1:N
        w_j = w_j / η
    end
    resample (M) according to weights w_j
end
```

Odometry Motion Model

Algorithm sample_motion_model_odometry(u_t, x_{t-1}):

$$\delta_{\text{rot1}} = \text{atan2}(\bar{y}' - \bar{y}, \bar{x}' - \bar{x}) - \bar{\theta}$$
$$\delta_{\text{trans}} = \sqrt{(\bar{x} - \bar{x}')^2 + (\bar{y} - \bar{y}')^2}$$
$$\delta_{\text{rot2}} = \bar{\theta}' - \bar{\theta} - \delta_{\text{rot1}}$$

$$\hat{\delta}_{\text{rot1}} = \delta_{\text{rot1}} - \textbf{sample}(\alpha_1 \delta_{\text{rot1}}^2 + \alpha_2 \delta_{\text{trans}}^2)$$
$$\hat{\delta}_{\text{trans}} = \delta_{\text{trans}} - \textbf{sample}(\alpha_3 \delta_{\text{trans}}^2 + \alpha_4\ \delta_{\text{rot1}}^2 + \alpha_4\ \delta_{\text{rot2}}^2)$$
$$\hat{\delta}_{\text{rot2}} = \delta_{\text{rot2}} - \textbf{sample}(\alpha_1 \delta_{\text{rot2}}^2 + \alpha_2 \delta_{\text{trans}}^2)$$

$$x' = x + \hat{\delta}_{\text{trans}}\ \cos(\theta + \hat{\delta}_{\text{rot1}})$$
$$y' = y + \hat{\delta}_{\text{trans}}\ \sin(\theta + \hat{\delta}_{\text{rot1}})$$
$$\theta' = \theta + \hat{\delta}_{\text{rot1}} + \hat{\delta}_{\text{rot2}}$$

return $x_t = (x', y', \theta')^T$

Table 5.6 Algorithm for sampling from $p(x_t \mid u_t, x_{t-1})$ based on odometry information. Here the pose at time t is represented by $x_{t-1} = (x\ \ y\ \ \theta)^T$. The control is a differentiable set of two pose estimates obtained by the robot's odometer, $u_t = (\bar{x}_{t-1}\ \ \bar{x}_t)^T$, with $\bar{x}_{t-1} = (\bar{x}\ \ \bar{y}\ \ \bar{\theta})$ and $\bar{x}_t = (\bar{x}'\ \ \bar{y}'\ \ \bar{\theta}')$.

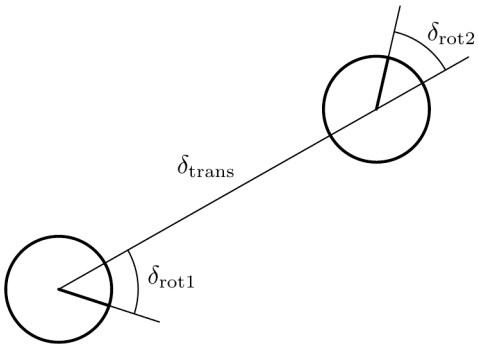
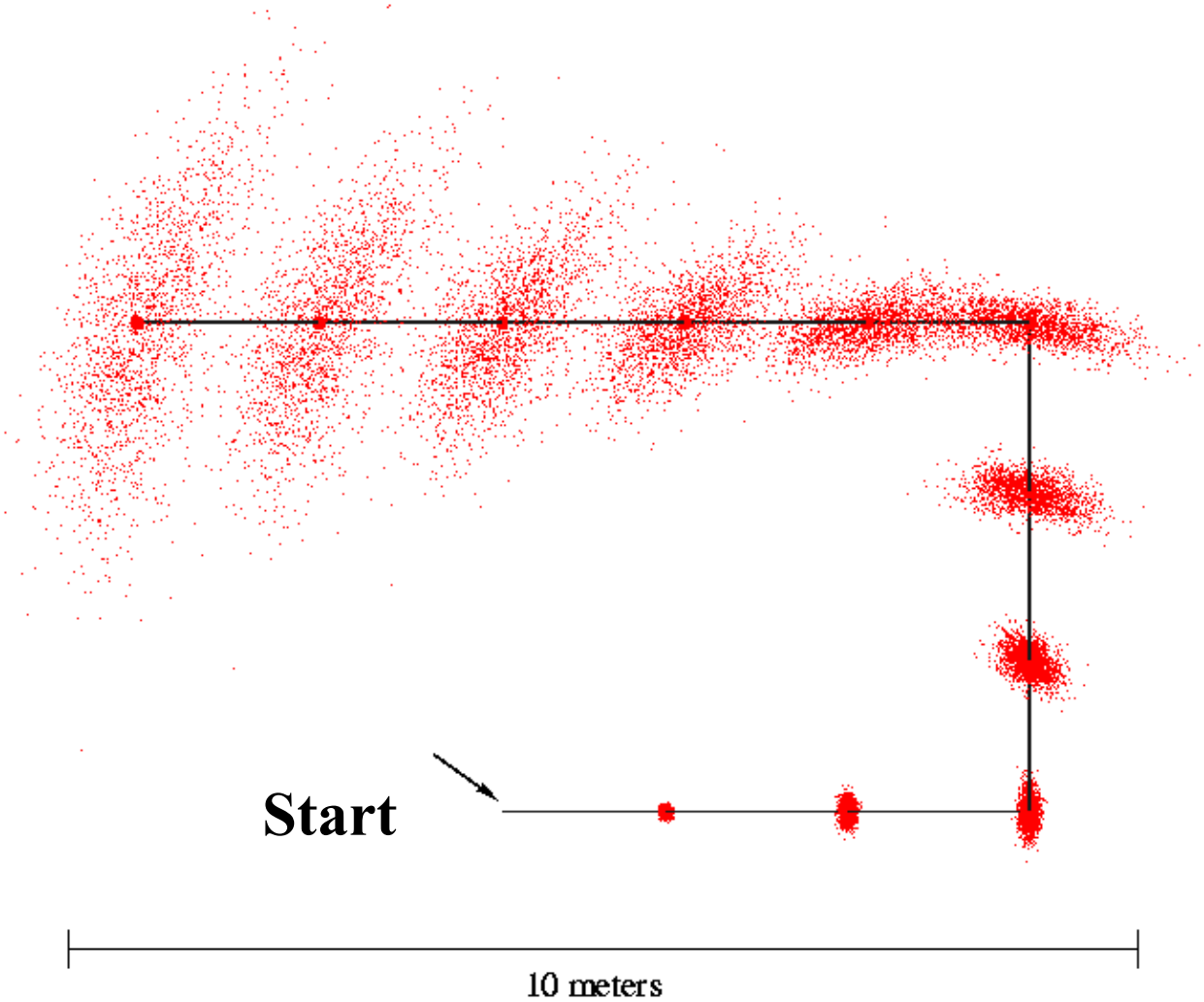


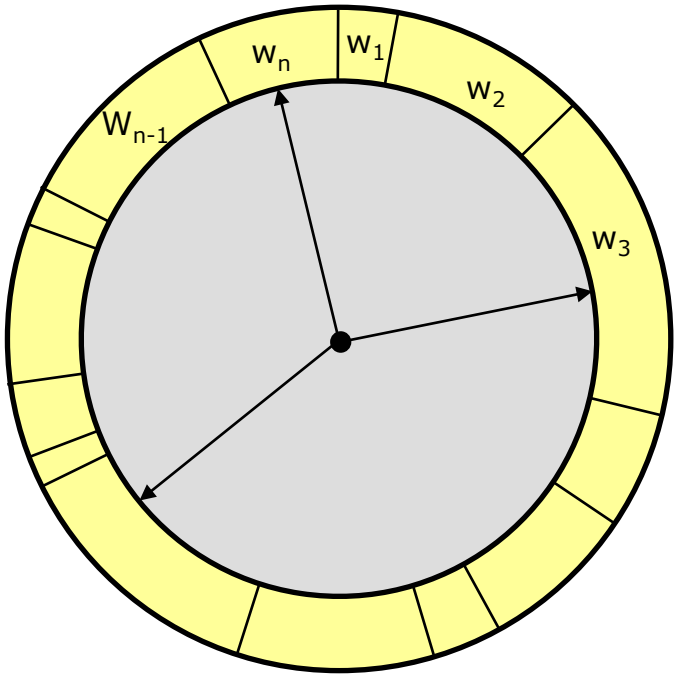
Figure 5.7 Odometry model: The robot motion in the time interval $(t-1, t]$ is approximated by a rotation δ_{rot1}, followed by a translation δ_{trans} and a second rotation δ_{rot2}. The turns and translations are noisy.

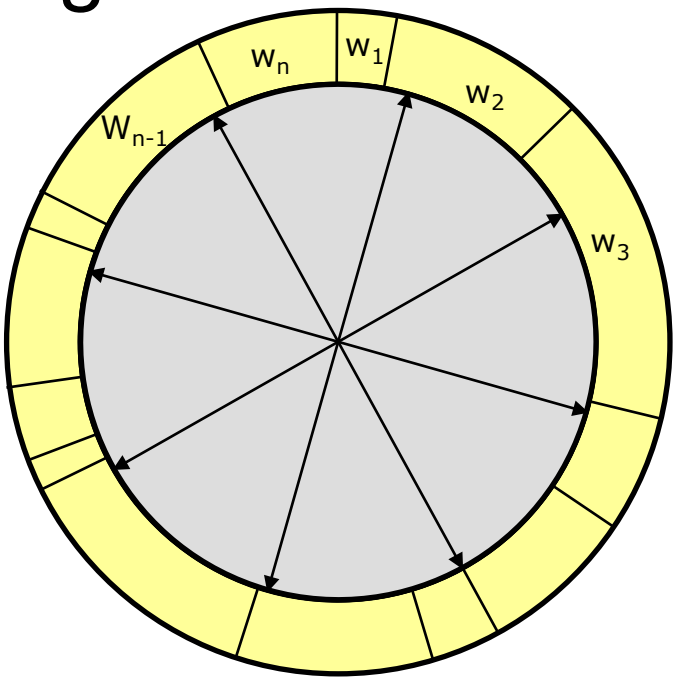
Probabilistic Robotics

Motion Model



Resampling

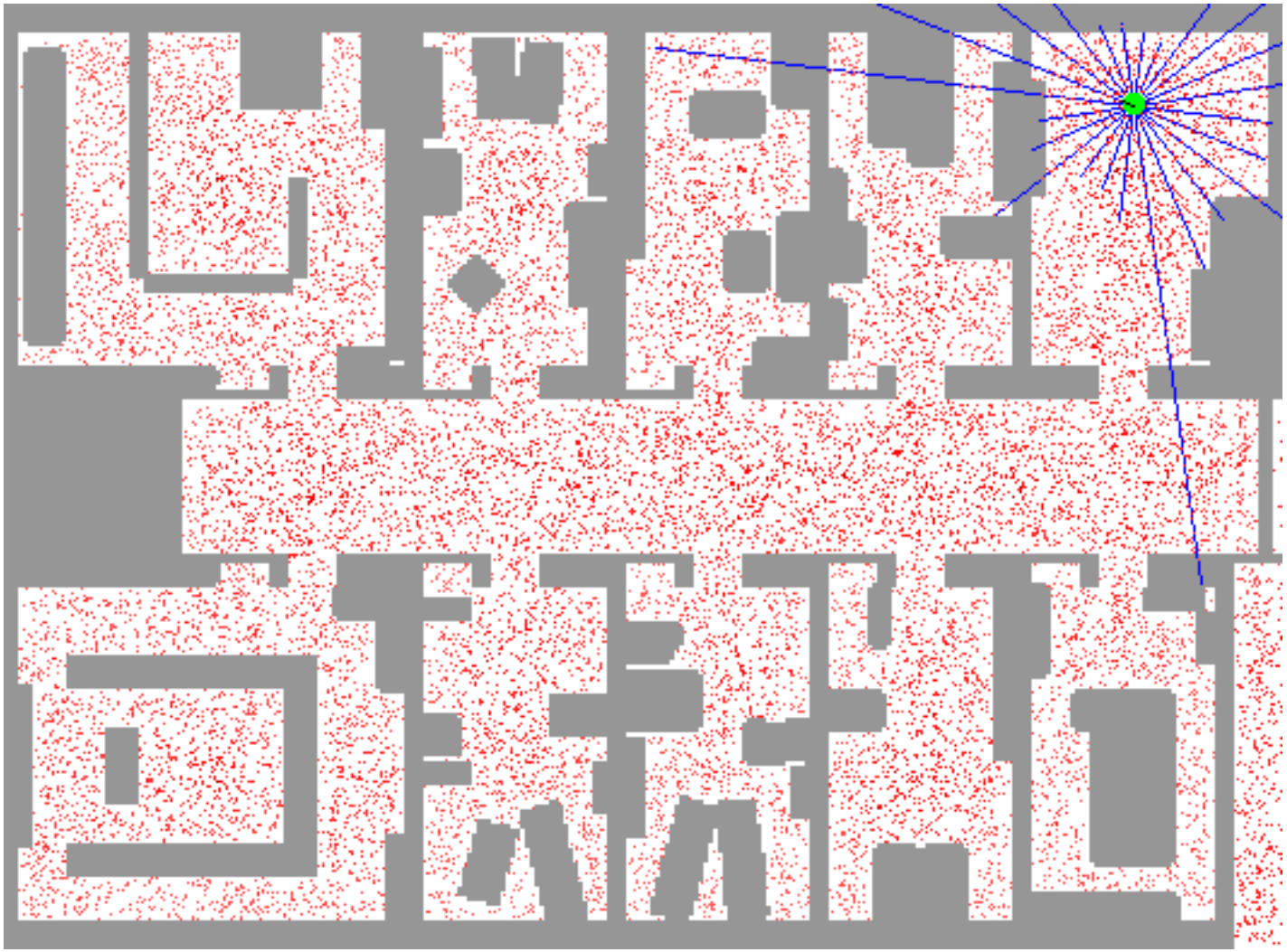




- Roulette wheel
- Binary search, log n

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

Sample-based Localization (sonar)

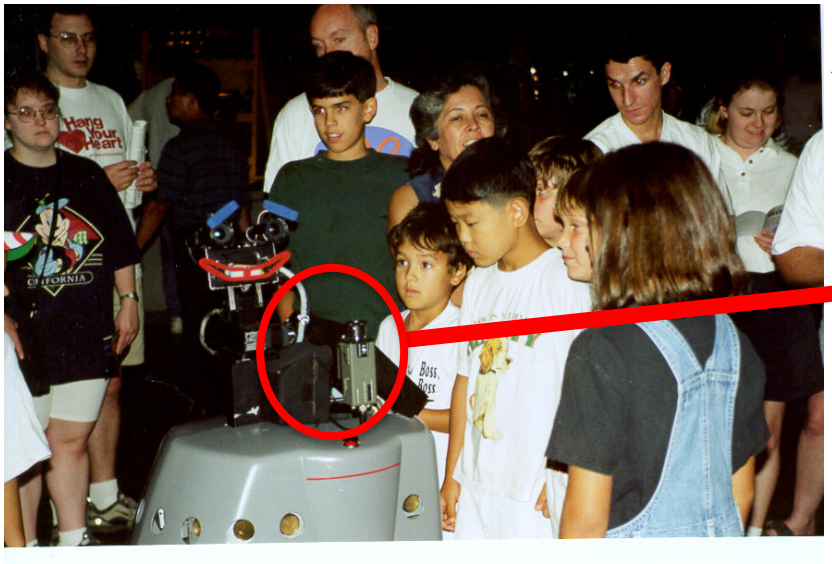


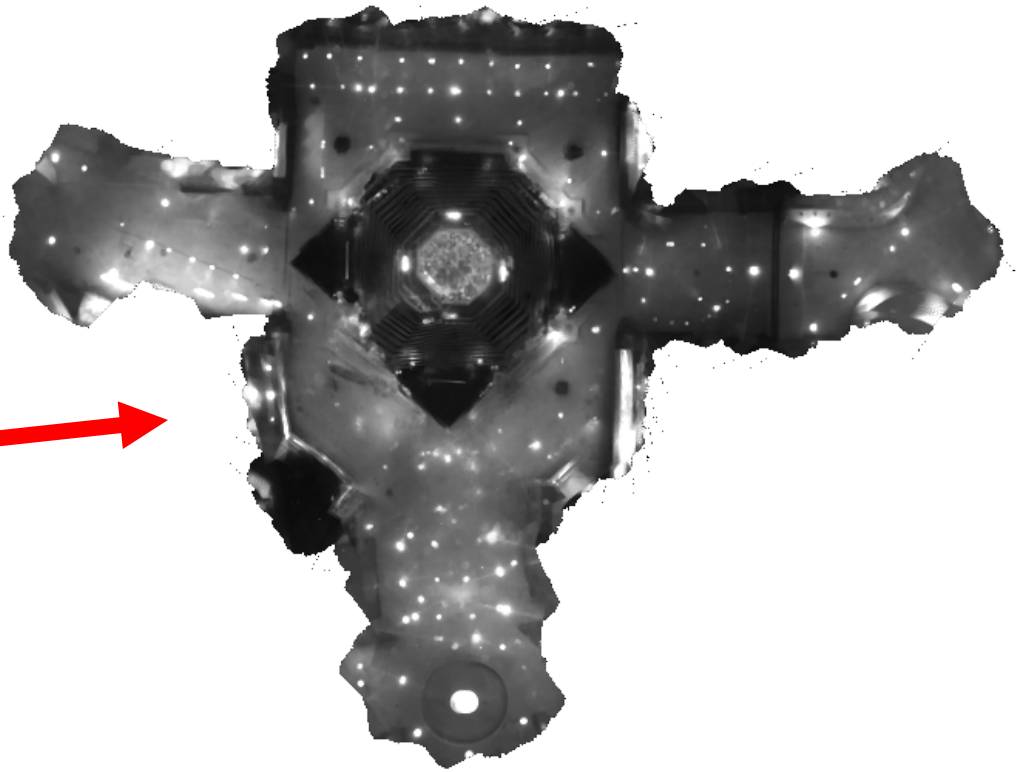
rse-lab.cs.washington.edu

Tour of the Smithsonian National Museum of American History

Burgard

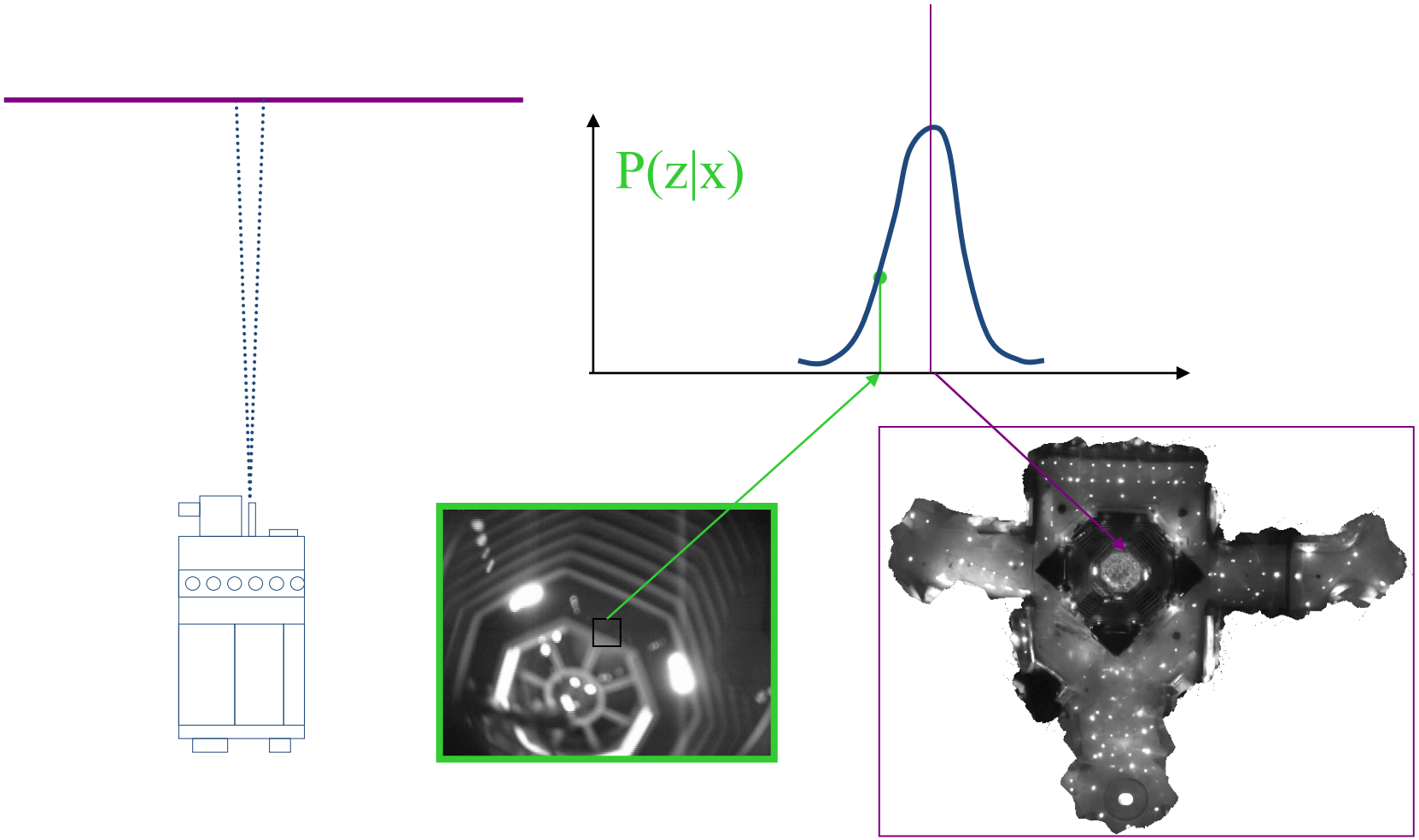
Using Ceiling Maps for Localization





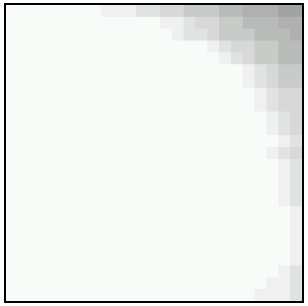
[Dellaert et al. 99]

Vision-Based Localization



Under a Light

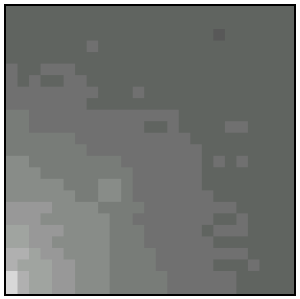
Measurement z:



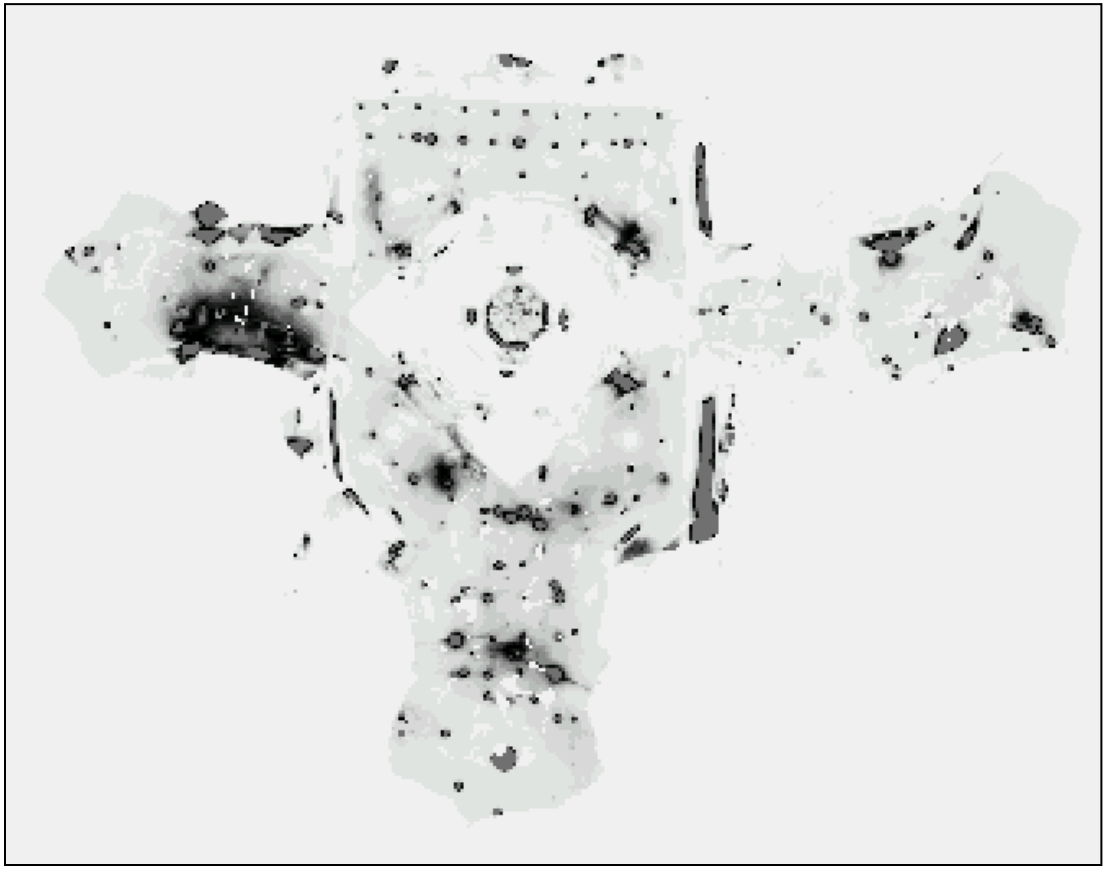
***P(z|x)*:**

Next to a Light

Measurement z:



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



Elsewhere

Measurement z:

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