Histogram filter

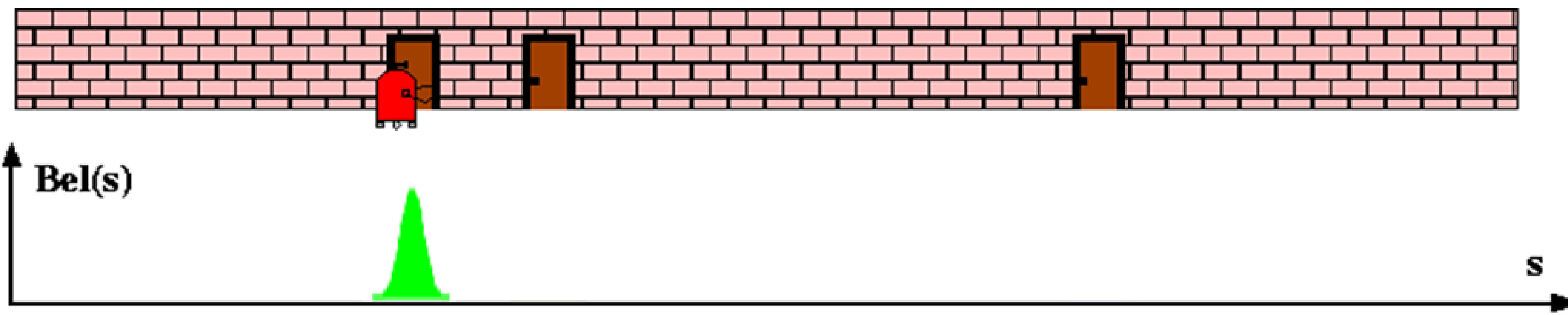
Non-parametric Filters

Localizing a Robot in a Hallway

- consider a robot moving down a hall equipped with a sensor that measures the presence of a door beside the robot
  - the pose of the robot is simply its location on a line down the middle of the hall
  - the robot starts out knowing how far down the hallway it is located
    - Kalman-like filters require an initial estimate of the location
  - robot has a map of the hallway showing it where the doors are

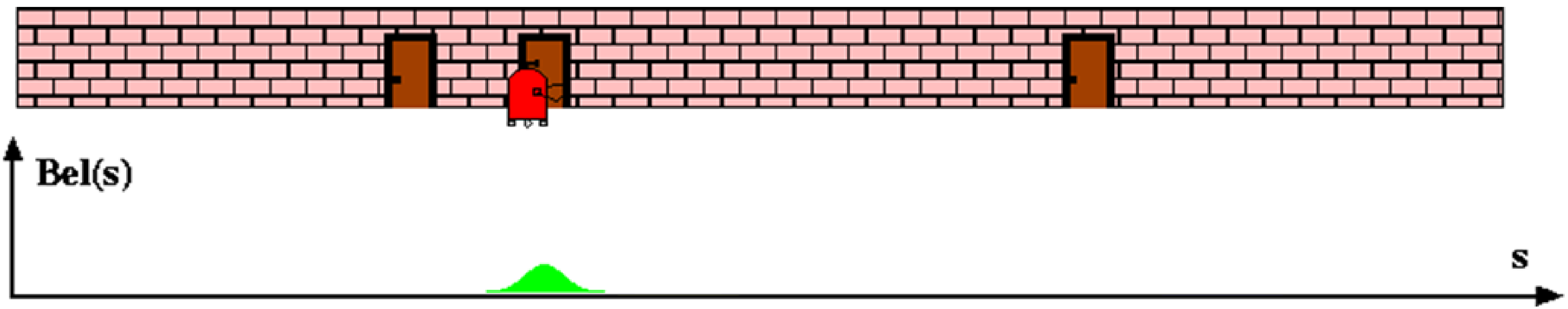
Kalman Localization

- robot starts out knowing how far down the hallway it is located



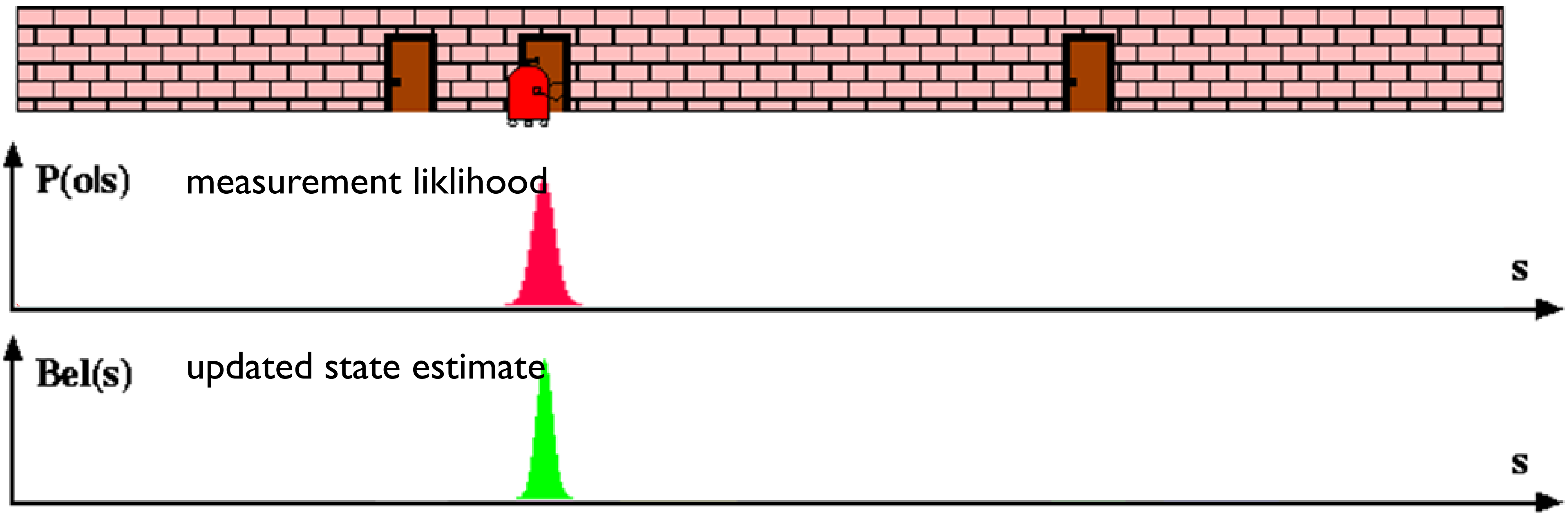
Kalman Localization

- as the robot moves forward, its uncertainty in its location shifts and grows according to its motion model



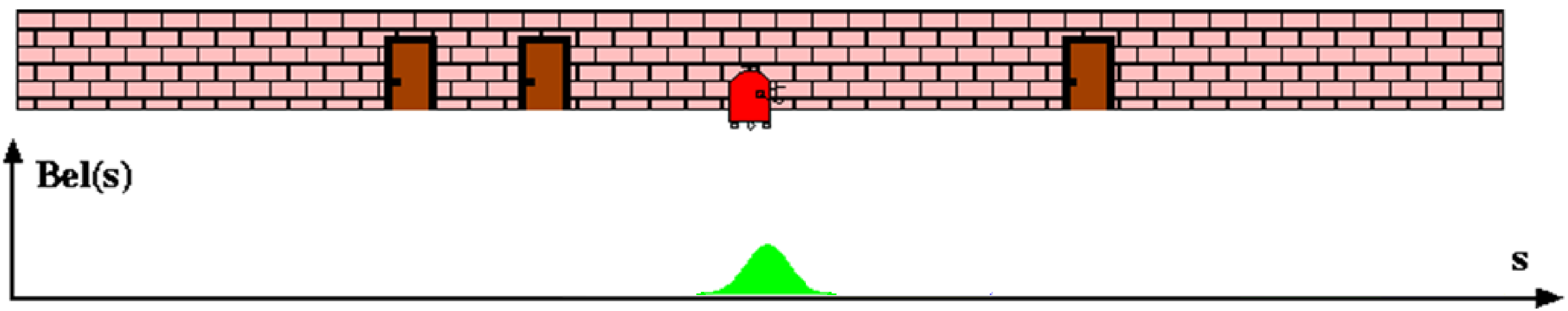
Grid Localization

- when it reaches a door *that can be uniquely identified*, it can incorporate this measurement into its state estimate



Grid Localization

- as the robot moves forward, its uncertainty in its location shifts and grows according to its motion model

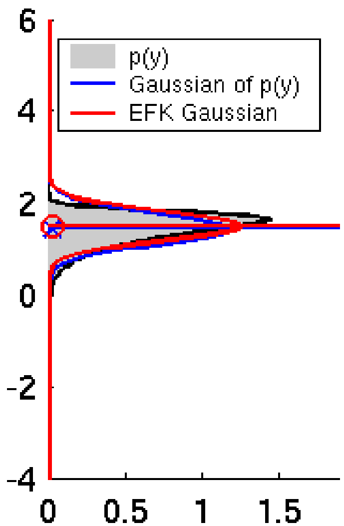


Gaussian Assumption

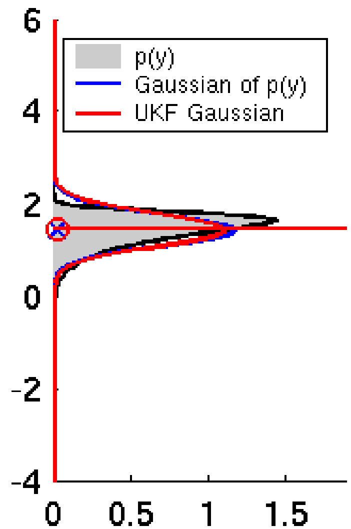
- Kalman-like filters assume that quantities can be represented accurately as a mean + covariance
  - e.g., the state is a random variable with Gaussian distribution
  - e.g., measurements are random variables with Gaussian distribution

Gaussian Assumption

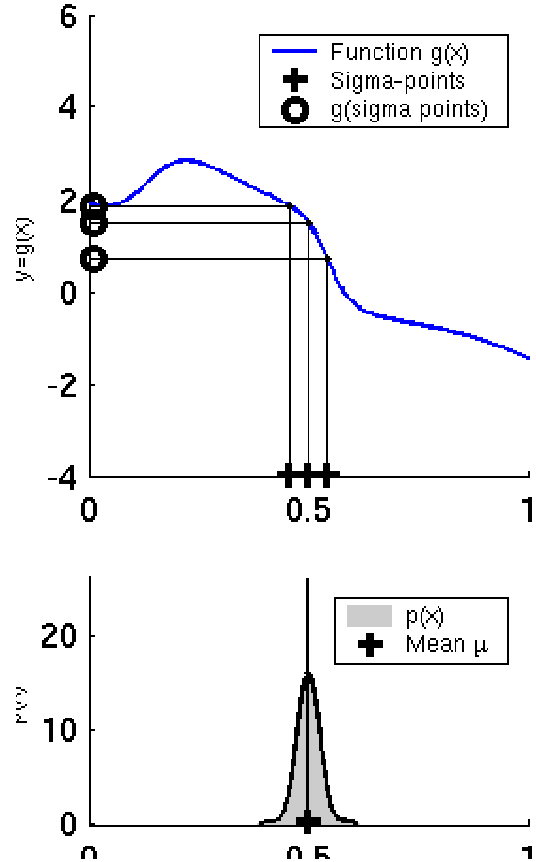
- assumption is ok here



EKF

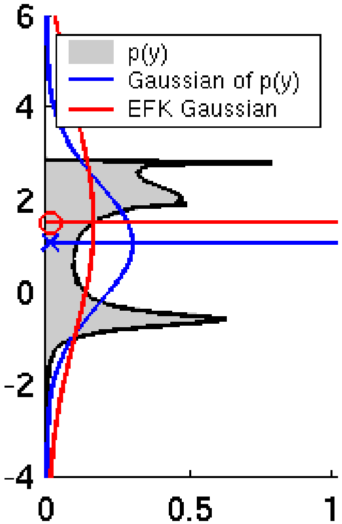


UKF

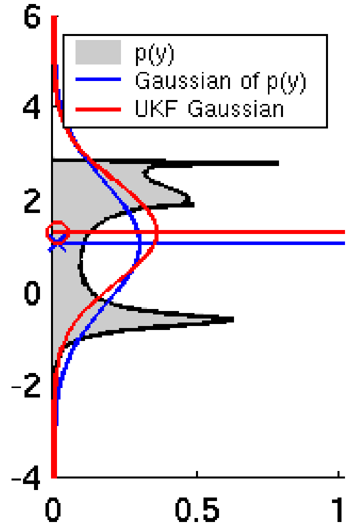


Gaussian Assumption

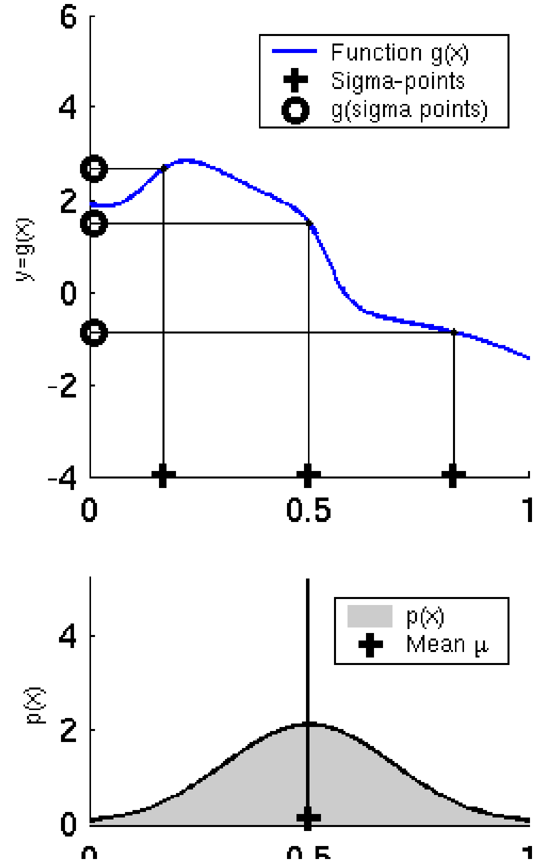
- assumption is (possibly) not ok here



EKF

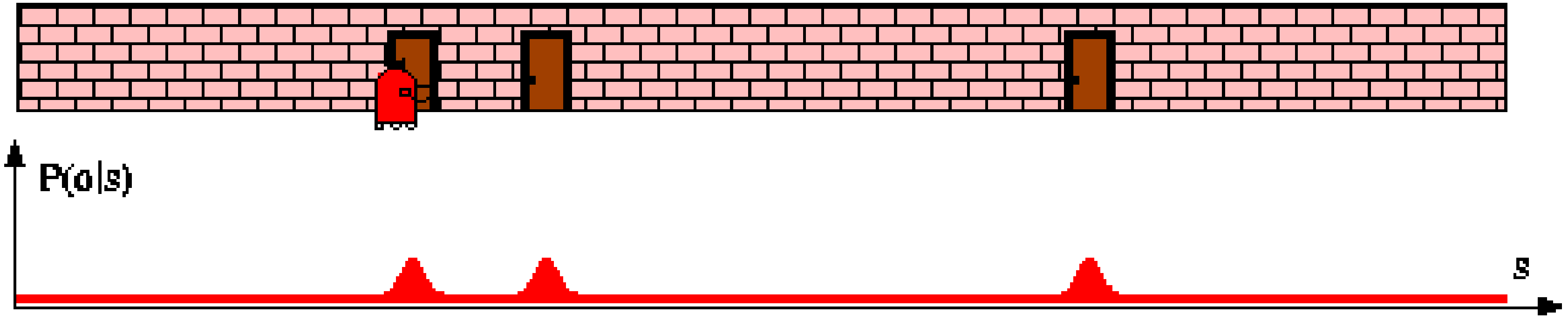


UKF



Gaussian Assumption

- assumption is not ok here (robot does not know which door it is measuring)

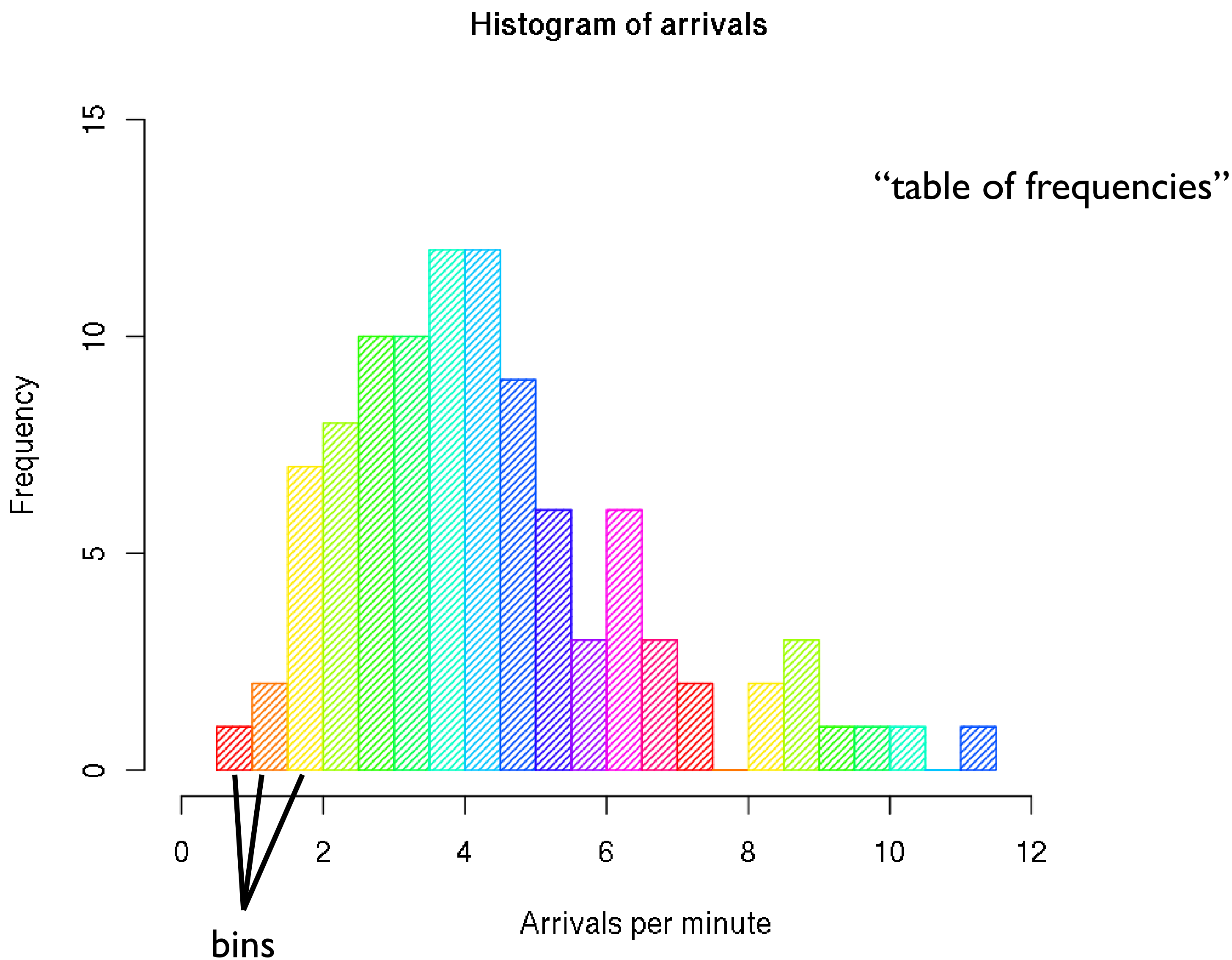


p(robot is sensing a door | state)

Non-parametric Filters

- non-parametric filters do not rely on a fixed functional form of the state posterior
- instead, they represent the posterior using a finite number of values each roughly corresponding to a region (or point) in state space
- two variations
  1. partition state space into a finite number of regions
     - e.g., histogram filter
  2. represent the posterior using a finite number of samples
     - e.g., particle filter

Histogram



Histogram of arrivals
Frequency
0
5
10
15
0
2
4
6
8
10
12
Arrivals per minute
"table of frequencies"
bins

Histogram Filter

- histogram filter uses a histogram to represent probability densities
- in its simplest form, the domain of the densities is divided into subdomains of equal size with each subdomain being a bin of the histogram
  - the value stored in the bin is proportional to the density

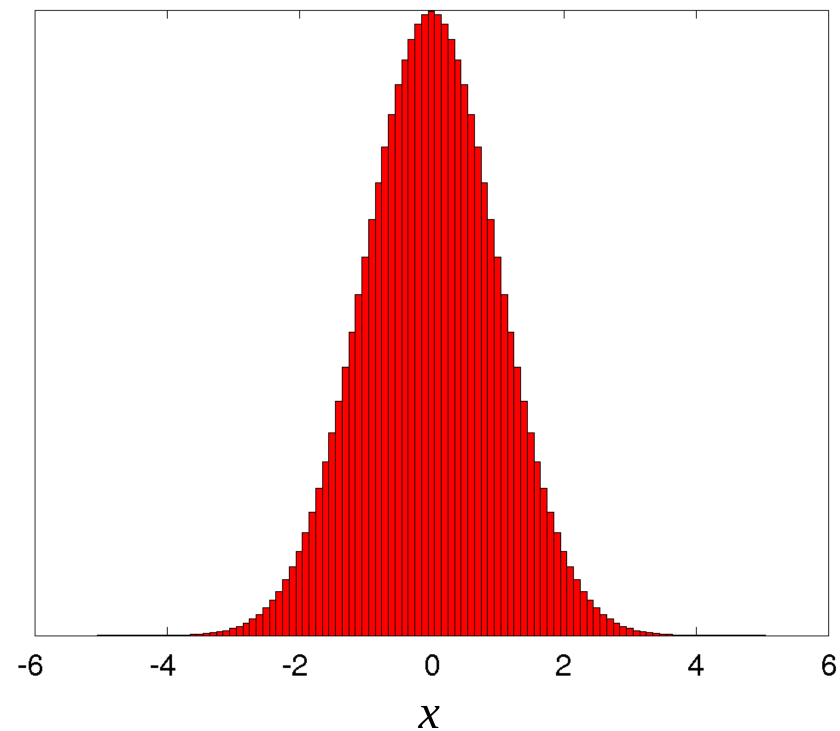
Histogram Filter

- suppose the domain of the state x is [-5, 5] and that x is a random variable with Gaussian density (mean 0, variance 1)
  - using bins of width $w = 0.1$ we can represent the density using the following histogram

$$\overbrace{n_i}^{\text{height of bar}} = \overbrace{\mathrm{pdf}(\underbrace{x_{c,i}}_{\text{center of bin } i})}^{\text{Gaussian PDF}} / w$$

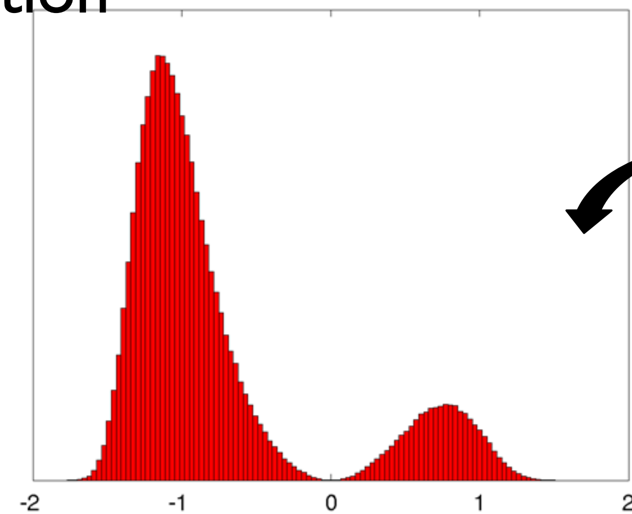
$$\sum_i n_i = 1$$

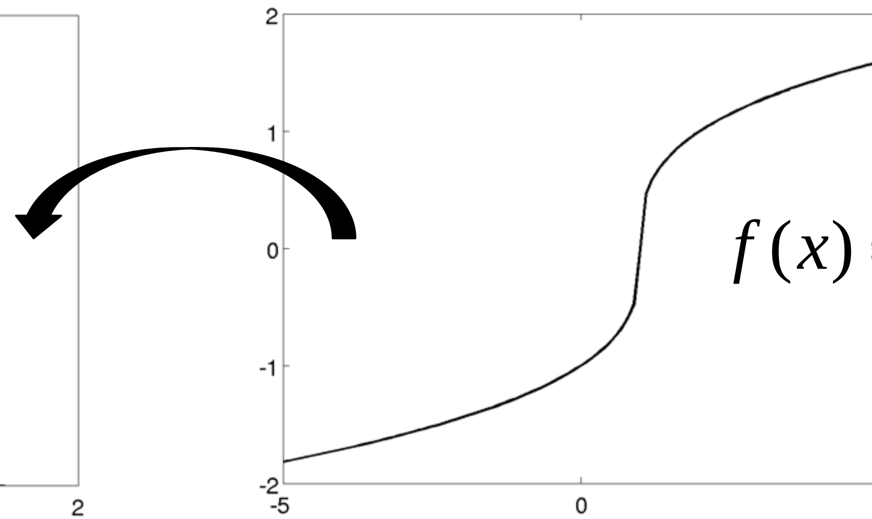
histogram p



Histogram Filter

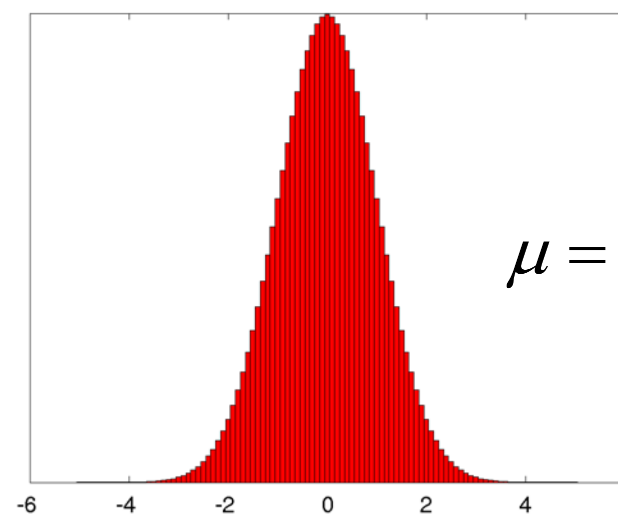
- suppose we want to pass the density through some non-linear function





$$f(x) = \sqrt[3]{x-1}$$

reminder: this is the solution obtained by passing 500,000 random samples through $f(x)$, not the result of using a histogram filter



$$\mu = 0, \Sigma = 1$$

Histogram Filter

1. create an empty histogram h with bins $x_{c,i}$
2. for each i
   1. $y_i = f\,(x_{c,i})$
   2. $n_i = p(x_{c,i})$
   3. find the bin b_k that y_i belongs in
   4. $h\,(b_k) = h\,(b_k) + n_i$

A Simple Implementation

```
dx = 0.05;                      % width of x bins
xc = -5:dx:5;                   % bin centers x
y = nthroot(xc – 1, 3);         % y = f(xc)
n = normpdf(xc, 0, 1);          % n = p(xc)
dy = 0.1;                       % width of y bins
yc = -2:dy:2;                   % bin centers y
h = zeros(size(yc));            % histogram
for i = 1:length(y)
    bk = find(y(i) > yc – (dy / 2) & y(i) < yc + (dy / 2));
    h(bk) = h(bk) + n(i);
end
bar(yc, h, 1);
```

Histogram Filter

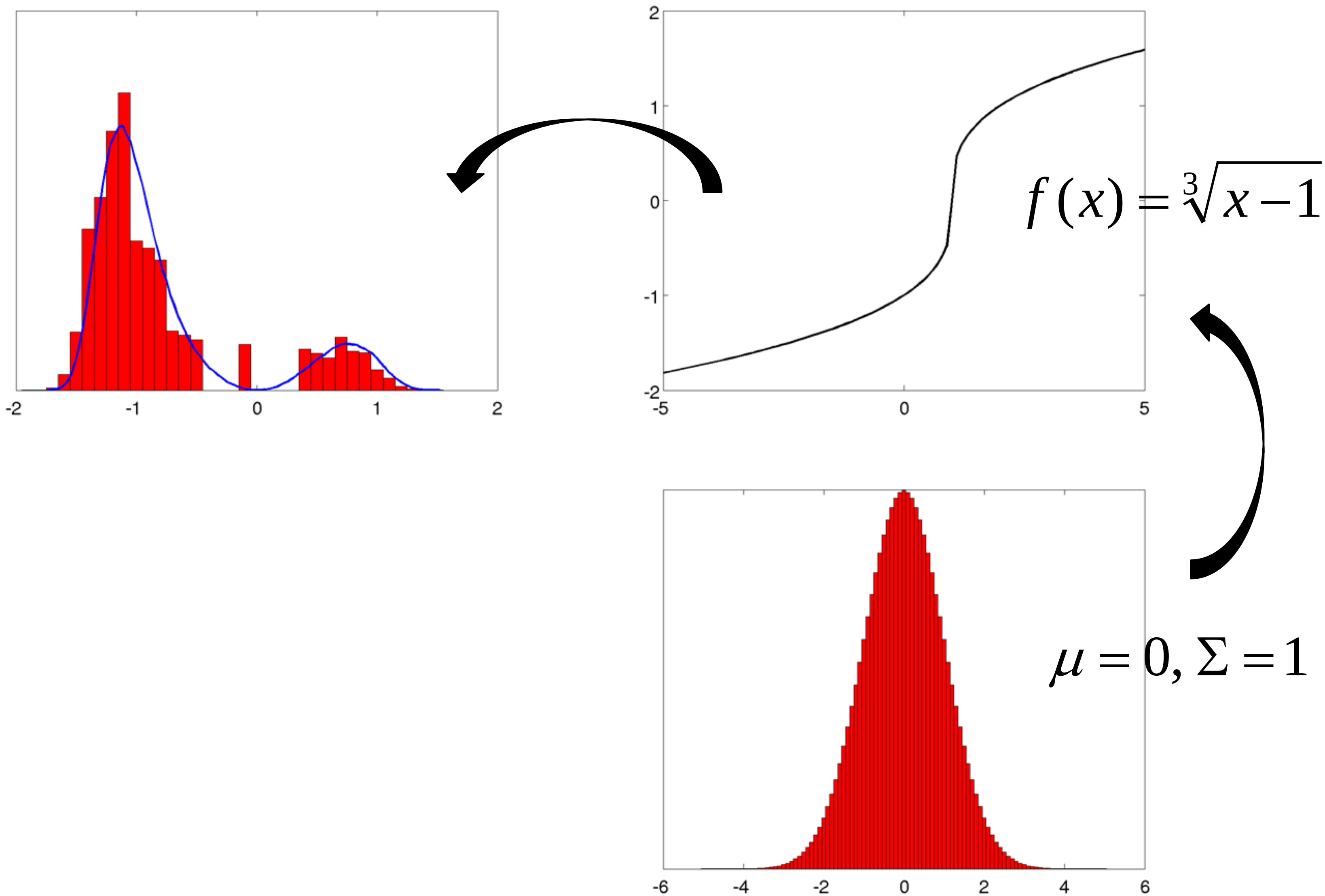
- alternatively

1. create an empty histogram h with bins $x_{c,i}$
2. for each bin b_k
   1. $h(b_k) = \sum_i p(x_{c,i})\ \text{in\_bin}(y_i, b_k)$

$y_i = f(x_{c,i})$

$\text{in\_bin}(y_i, b_k)$ probability that y_i is in bin b_k

Histogram Filter



$$f(x)=\sqrt[3]{x-1}$$

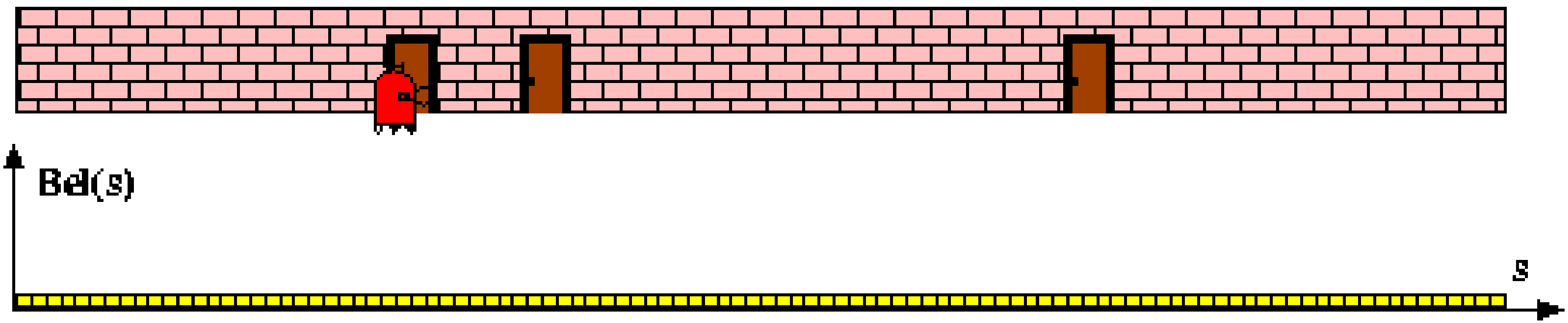
$$\mu=0, \Sigma=1$$

Grid Localization

- grid localization uses a histogram filter over a grid decomposition of pose space
- consider a robot moving down a hall equipped with a sensor that measures the presence of a door beside the robot
  - the pose of the robot is simply its location on a line down the middle of the hall
  - the robot starts out having no idea how far down the hallway it is located
  - robot has a map of the hallway showing it where the doors are
  - grid decomposes the hallway into a finite set of non-overlapping intervals
    - e.g., every 50cm would yield intervals [0, 0.5], (0.5, 1], (1, 1.5], …

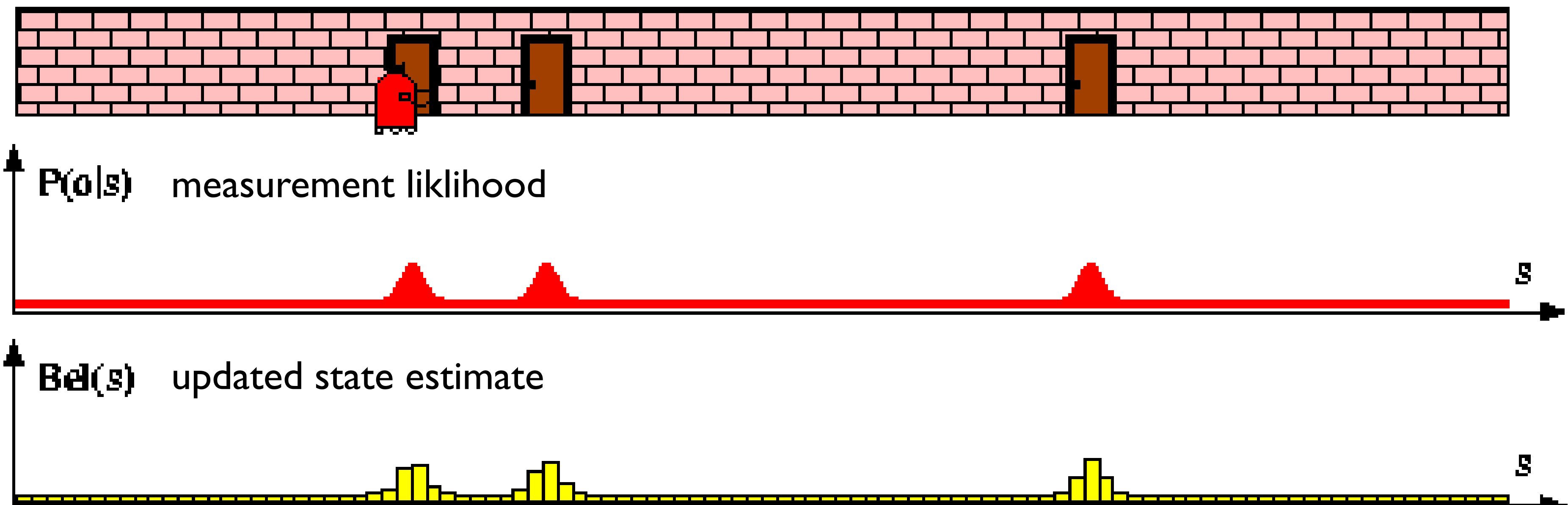
Grid Localization

- the robot starts out having no idea how far down the hallway it is located
  - the histogram of its state density is uniform



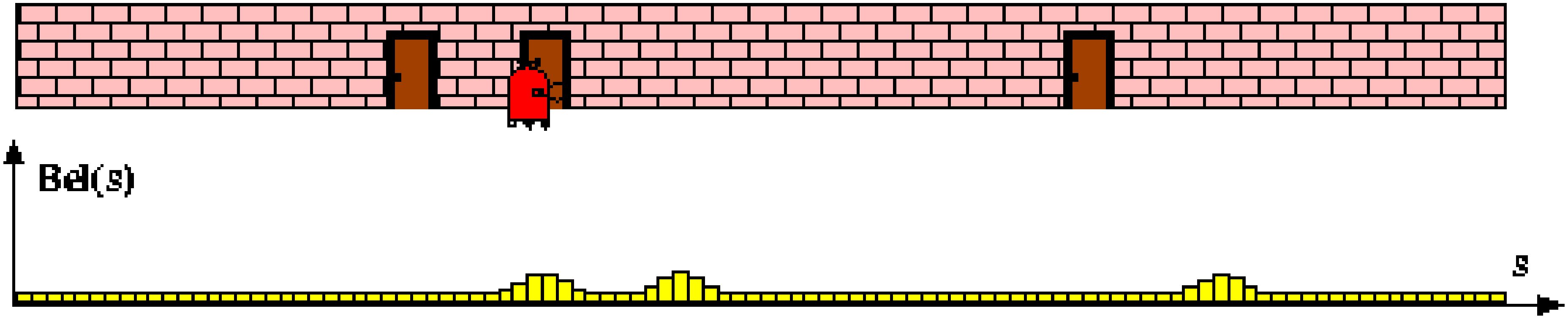
Grid Localization

- because the robot is beside a door, it has a measurement
  - it can incorporate this measurement into its state estimate



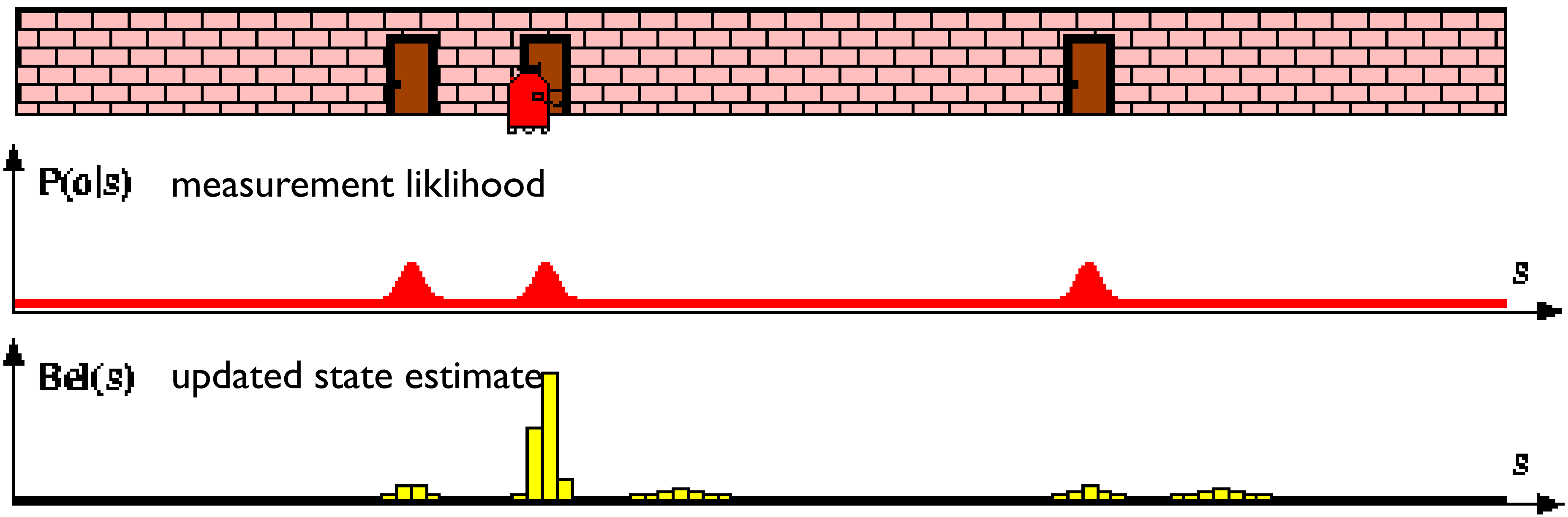
Grid Localization

- as the robot moves forward, its uncertainty in its location shifts and grows according to its motion model



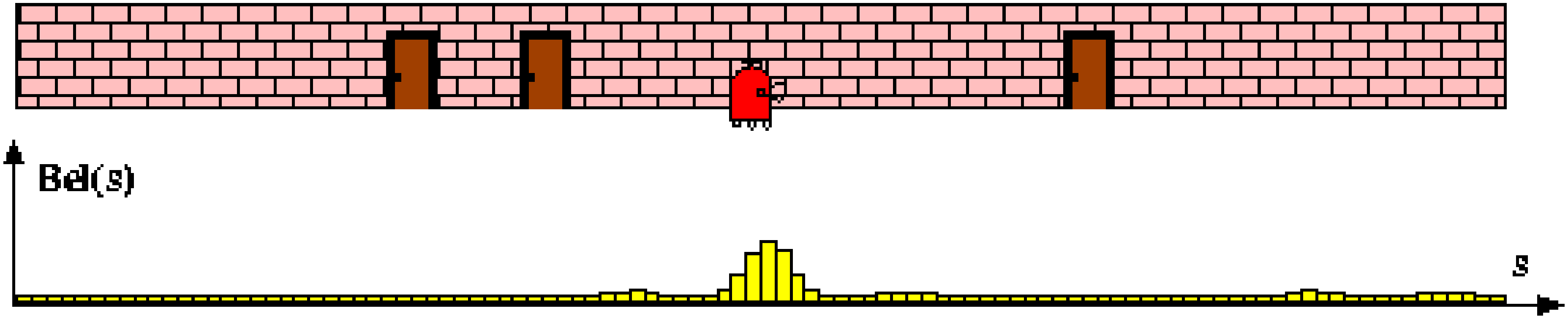
Grid Localization

- when it reaches a door, it can incorporate this measurement into its state estimate
  - it now has a pretty good idea where it is in the hallway



Grid Localization

- as the robot moves forward, its uncertainty in its location shifts and grows according to its motion model



Grid Localization Algorithm

1. algorithm_grid_localization($\{p_{k,t-1}\}, u_t, z_t, m$)
2. for all k do
3. $\bar{p}_{k,t} = \sum_i p_{i,t-1}$ motion_model($\text{mean}(\mathbf{x}_k), u_t, \text{mean}(\mathbf{x}_i)$)
4. $p_{k,t} = \eta \; \bar{p}_{k,t}$ measurement_model($z_t, \text{mean}(\mathbf{x}_k), m$)
5. endfor
6. return $\{p_{k,t}\}$

$\{p_{k,t}\}$	histogram
u_t	control input
z_t	measurement
m	map
$\text{mean}(\mathbf{x}_i)$	center of mass of grid cell $\mathbf{x}_i$