CSE 573: Artificial Intelligence
Winter 2019

Hidden Markov Models

Hanna Hajishirzi

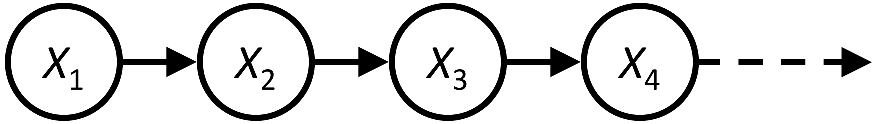
Many slides adapted from Pieter Abbeel, Dan Klein, Dan Weld,Stuart Russell, Andrew Moore & Luke Zettlemoyer

Today

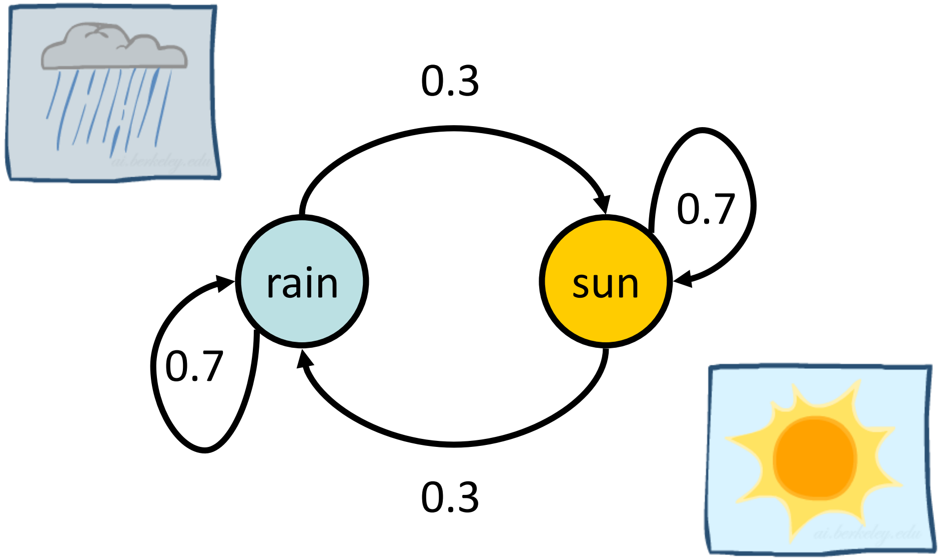
- HMMs
  - Particle filters
  - Demos!
- Applications:
  - Robot localization / mapping
- Bayes Nets

Recap: Reasoning Over Time

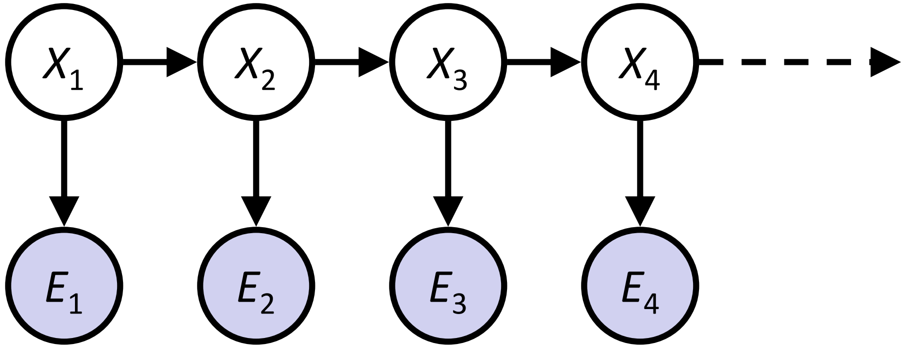
- Markov models



$P(X_1)$ $\qquad$ $P(X|X_{-1})$



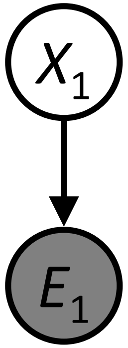
- Hidden Markov models

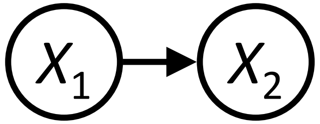


$$P(E|X)$$

X	E	P
rain	umbrella	0.9
rain	no umbrella	0.1
sun	umbrella	0.2
sun	no umbrella	0.8

Inference: Base Cases





$$P(X_1|e_1)$$

$$\begin{aligned} P(x_1|e_1) &= P(x_1, e_1)/P(e_1) \\ &\propto_{X_1} P(x_1, e_1) \\ &= P(x_1)P(e_1|x_1) \end{aligned}$$

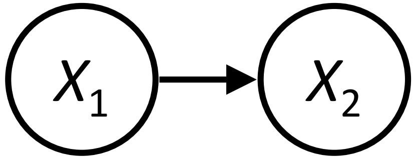
$$P(X_2)$$

$$\begin{aligned} P(x_2) &= \sum_{x_1} P(x_1, x_2) \\ &= \sum_{x_1} P(x_1)P(x_2|x_1) \end{aligned}$$

Passage of Time

- Assume we have current belief P(X | evidence to date)

$$B(X_t) = P(X_t|e_{1:t})$$



- Then, after one time step passes:

$$P(X_{t+1}|e_{1:t}) = \sum_{x_t} P(X_{t+1}, x_t|e_{1:t})$$

$$= \sum_{x_t} P(X_{t+1}|x_t, e_{1:t})P(x_t|e_{1:t})$$

$$= \sum_{x_t} P(X_{t+1}|x_t)P(x_t|e_{1:t})$$

- Or compactly:

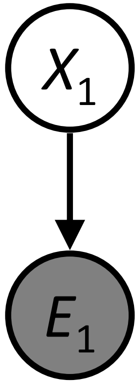
$$B'(X_{t+1}) = \sum_{x_t} P(X'|x_t)B(x_t)$$

- Basic idea: beliefs get “pushed” through the transitions
  - With the “B” notation, we have to be careful about what time step t the belief is about, and what evidence it includes

Observation

- Assume we have current belief P(X | previous evidence):

$$B'(X_{t+1}) = P(X_{t+1}|e_{1:t})$$



- Then, after evidence comes in:

$$P(X_{t+1}|e_{1:t+1}) = P(X_{t+1}, e_{t+1}|e_{1:t})/P(e_{t+1}|e_{1:t})$$

$$\propto_{X_{t+1}} P(X_{t+1}, e_{t+1}|e_{1:t})$$

$$= P(e_{t+1}|e_{1:t}, X_{t+1})P(X_{t+1}|e_{1:t})$$

$$= P(e_{t+1}|X_{t+1})P(X_{t+1}|e_{1:t})$$

- Or, compactly:

$$B(X_{t+1}) \propto_{X_{t+1}} P(e_{t+1}|X_{t+1})B'(X_{t+1})$$

- Basic idea: beliefs "reweighted" by likelihood of evidence
- Unlike passage of time, we have to renormalize

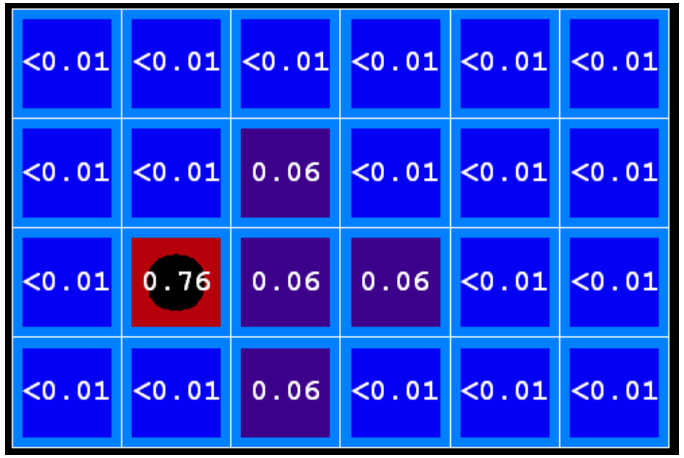
Filtering

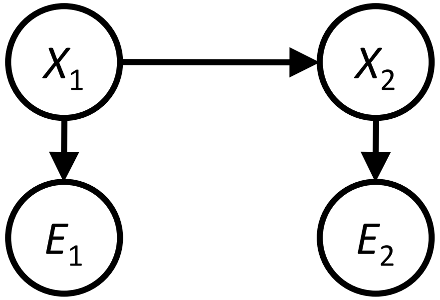
Elapse time: compute P(X_t | $e_{1:t-1}$)

$$\color{green}{P(x_t|e_{1:t-1})} = \sum_{x_{t-1}} \color{red}{P(x_{t-1}|e_{1:t-1})} \cdot P(x_t|x_{t-1})$$

Observe: compute P(X_t | $e_{1:t}$)

$$\color{red}{P(x_t|e_{1:t})} \propto \color{green}{P(x_t|e_{1:t-1})} \cdot P(e_t|x_t)$$





Belief: <P(rain), P(sun)>

$P(X_1)$	<0.5, 0.5>	*Prior on X_1*
$\color{red}{P(X_1 \mid E_1 = umbrella)}$	<0.82, 0.18>	*Observe*
$\color{green}{P(X_2 \mid E_1 = umbrella)}$	<0.63, 0.37>	*Elapse time*
$\color{red}{P(X_2 \mid E_1 = umb, E_2 = umb)}$	<0.88, 0.12>	*Observe*

Example: Weather HMM

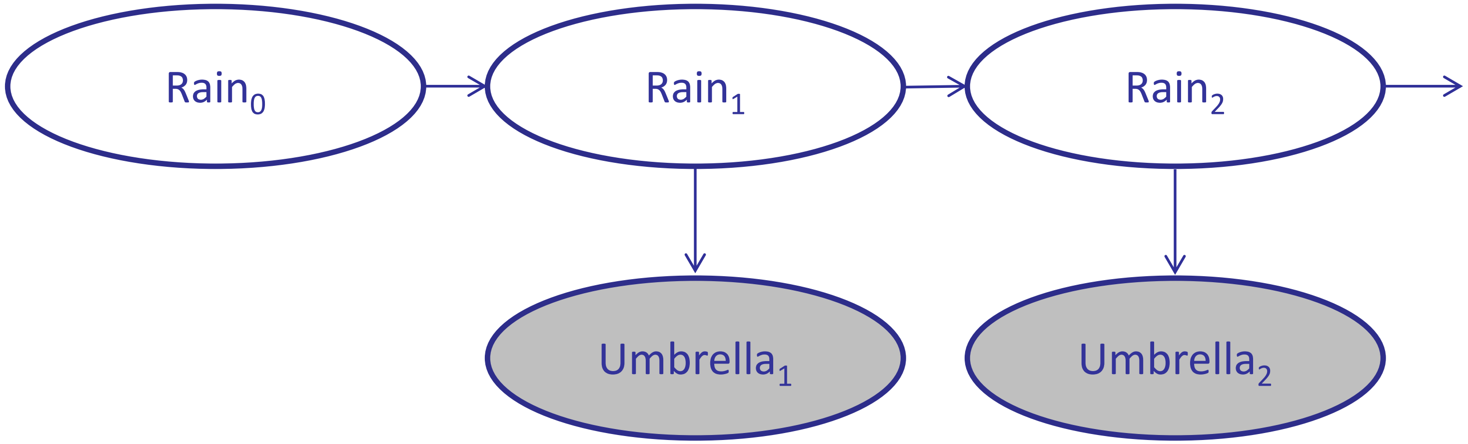
$B(+r) = 0.5$
$B(-r) = 0.5$

$B'(+r) = 0.5$
$B'(-r) = 0.5$

$B(+r) = 0.818$
$B(-r) = 0.182$

$B'(+r) = 0.627$
$B'(-r) = 0.373$

$B(+r) = 0.883$
$B(-r) = 0.117$



R_t	R_{t+1}	$P(R_{t+1} \mid R_t)$
+r	+r	0.7
+r	-r	0.3
-r	+r	0.3
-r	-r	0.7

R_t	U_t	$P(U_t \mid R_t)$
+r	+u	0.9
+r	-u	0.1
-r	+u	0.2
-r	-u	0.8

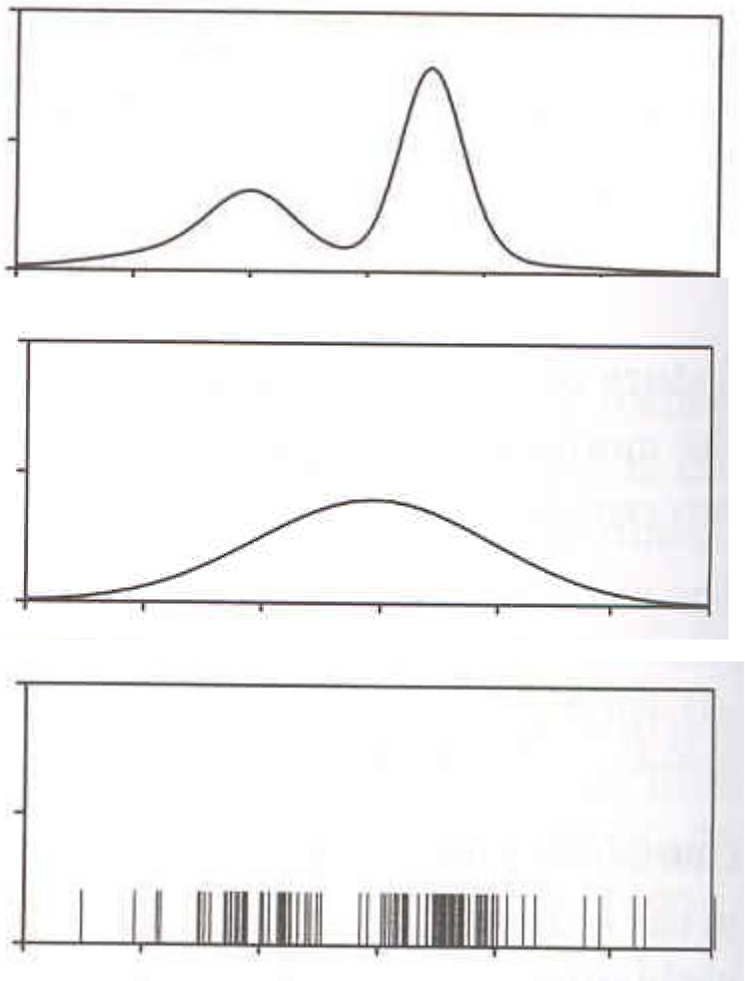
Approximate Inference

- Sometimes |X| is too big for exact inference
  - |X| may be too big to even store B(X)
  - E.g. when X is continuous
  - $|X|^2$ may be too big to do updates

- Solution: approximate inference by sampling
- How robot localization works in practice

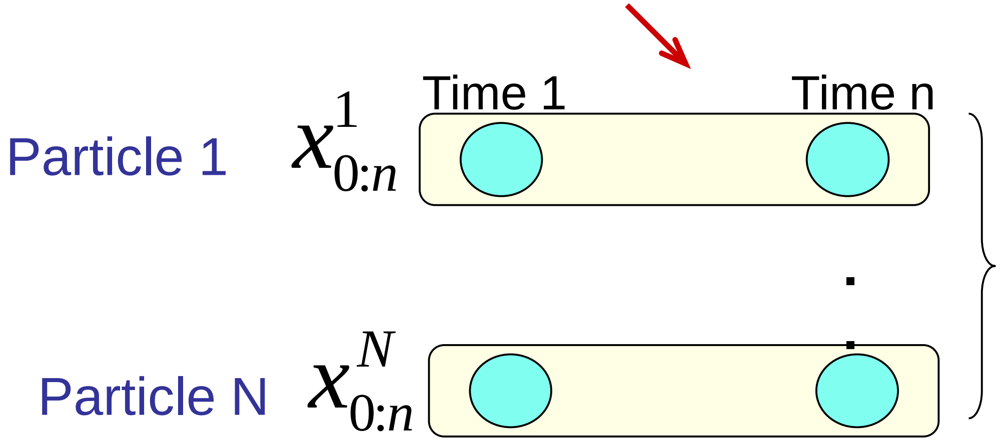
What is Sampling?

- Goal: Approximate the original distribution:
- Approximate with Gaussian distribution
- Draw samples from a distribution close enough to the original distribution



Approximate Solution: Perfect Sampling

Robot path till time n



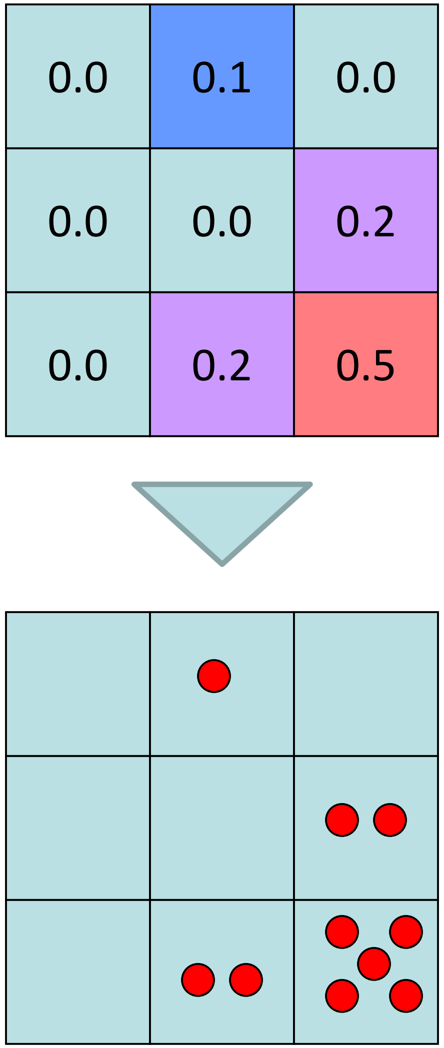
Assume we can sample from the original distribution $p(x_{0:n} \mid y_{0:n})$

$$P(x_{0:n} \mid y_{0:n}) = \frac{1}{N} \left(\text{Number of samples that match with query} \right)$$

Converges to the exact value for large N

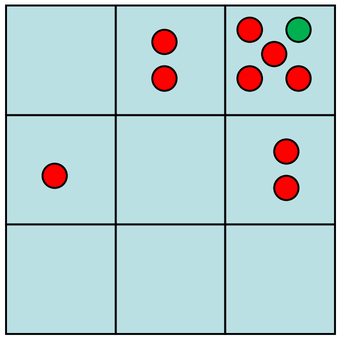
Particle Filtering

- Filtering: approximate solution
- Sometimes |X| is too big to use exact inference
  - |X| may be too big to even store B(X)
  - E.g. X is continuous
- Solution: approximate inference
  - Track samples of X, not all values
  - Samples are called particles
  - Time per step is linear in the number of samples
  - But: number needed may be large
  - In memory: list of particles, not states
- Particle is just new name for sample



Representation: Particles

- Our representation of P(X) is now a list of N particles (samples)
  - Generally, N << |X|
- P(x) approximated by number of particles with value x
  - So, many x may have P(x) = 0!
  - More particles, more accuracy
- For now, all particles have a weight of 1



Particles:
(3,3)
(2,3)
(3,3)
(3,2)
(3,3)
(3,2)
(1,2)
(3,3)
(3,3)
(2,3)

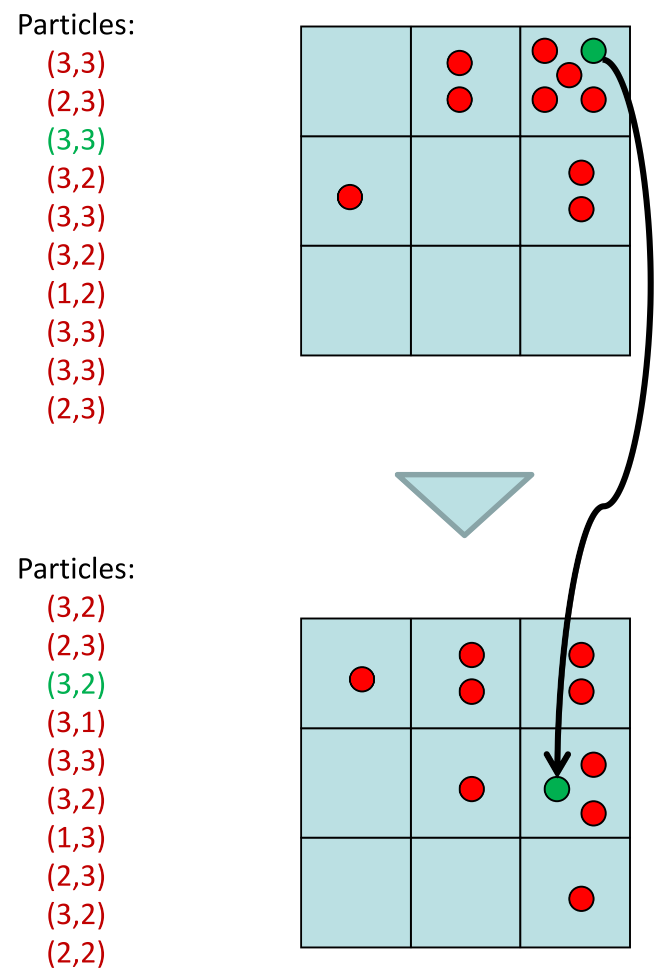
Particle Filtering: Elapse Time

- Each particle is moved by sampling its next position from the transition model

$$x' = \text{sample}(P(X'|x))$$

  - This is like prior sampling – samples' frequencies reflect the transition probabilities
  - Here, most samples move clockwise, but some move in another direction or stay in place

- This captures the passage of time
  - If enough samples, close to exact values before and after (consistent)



Particle Filtering: Observe

- Slightly trickier:
  - Don't sample observation, fix it
  - Downweight samples based on the evidence

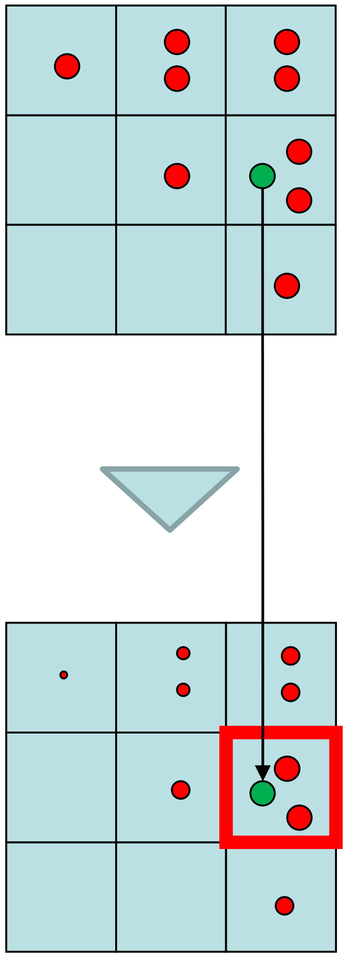
$$w(x) = P(e|x)$$

$$B(X) \propto P(e|X)B'(X)$$

  - As before, the probabilities don't sum to one, since all have been downweighted (in fact they now sum to (N times) an approximation of P(e))

Particles:
(3,2)
(2,3)
(3,2)
(3,1)
(3,3)
(3,2)
(1,3)
(2,3)
(3,2)
(2,2)

Particles:
(3,2) w=.9
(2,3) w=.2
(3,2) w=.9
(3,1) w=.4
(3,3) w=.4
(3,2) w=.9
(1,3) w=.1
(2,3) w=.2
(3,2) w=.9
(2,2) w=.4



Particle Filtering: Resample

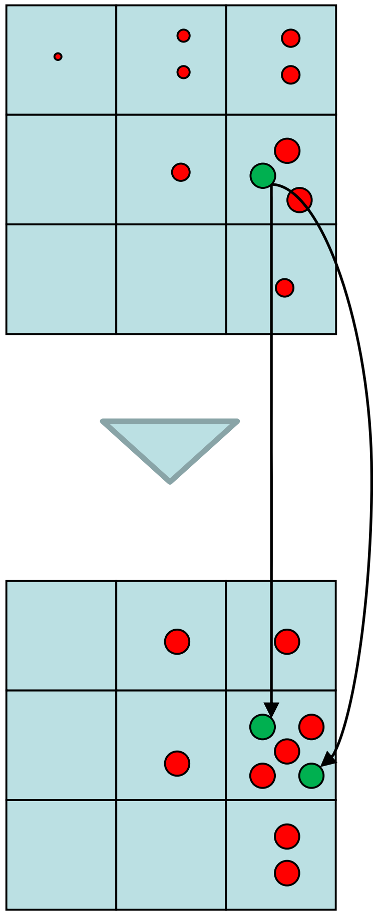
- Rather than tracking weighted samples, we resample
- N times, we choose from our weighted sample distribution (i.e. draw with replacement)
- This is equivalent to renormalizing the distribution
- Now the update is complete for this time step, continue with the next one

Particles:

(3,2) w=.9
(2,3) w=.2
(3,2) w=.9
(3,1) w=.4
(3,3) w=.4
(3,2) w=.9
(1,3) w=.1
(2,3) w=.2
(3,2) w=.9
(2,2) w=.4

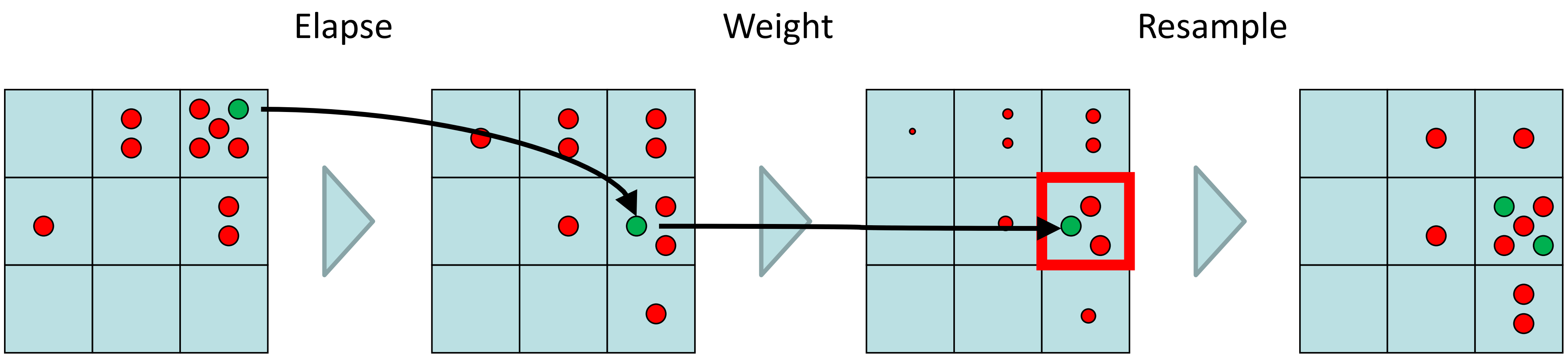
(New) Particles:

(3,2)
(2,2)
(3,2)
(2,3)
(3,3)
(3,2)
(1,3)
(2,3)
(3,2)
(3,2)



Recap: Particle Filtering

- Particles: track samples of states rather than an explicit distribution



Particles:	Particles:	Particles:	(New) Particles:
(3,3)	(3,2)	(3,2) w=.9	(3,2)
(2,3)	(2,3)	(2,3) w=.2	(2,2)
(3,3)	(3,2)	(3,2) w=.9	(3,2)
(3,2)	(3,1)	(3,1) w=.4	(2,3)
(3,3)	(3,3)	(3,3) w=.4	(3,3)
(3,2)	(3,2)	(3,2) w=.9	(3,2)
(1,2)	(1,3)	(1,3) w=.1	(1,3)
(3,3)	(2,3)	(2,3) w=.2	(2,3)
(3,3)	(3,2)	(3,2) w=.9	(3,2)
(2,3)	(2,2)	(2,2) w=.4	(3,2)

Particle Filtering Summary

- Represent current belief P(X | evidence to date) as set of *n* samples (actual assignments X=x)
- For each new observation e:

1. Sample transition, once for each current particle x

$$x' = \text{sample}(P(X'|x))$$

2. For each new sample x', compute importance weights for the new evidence e:

$$w(x') = P(e|x')$$

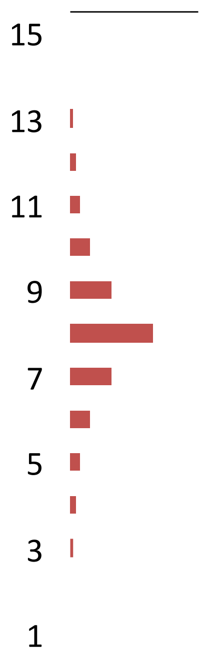
3. Finally, normalize by resampling the importance weights to create N new particles

HMM Examples & Applications

P4: Ghostbusters

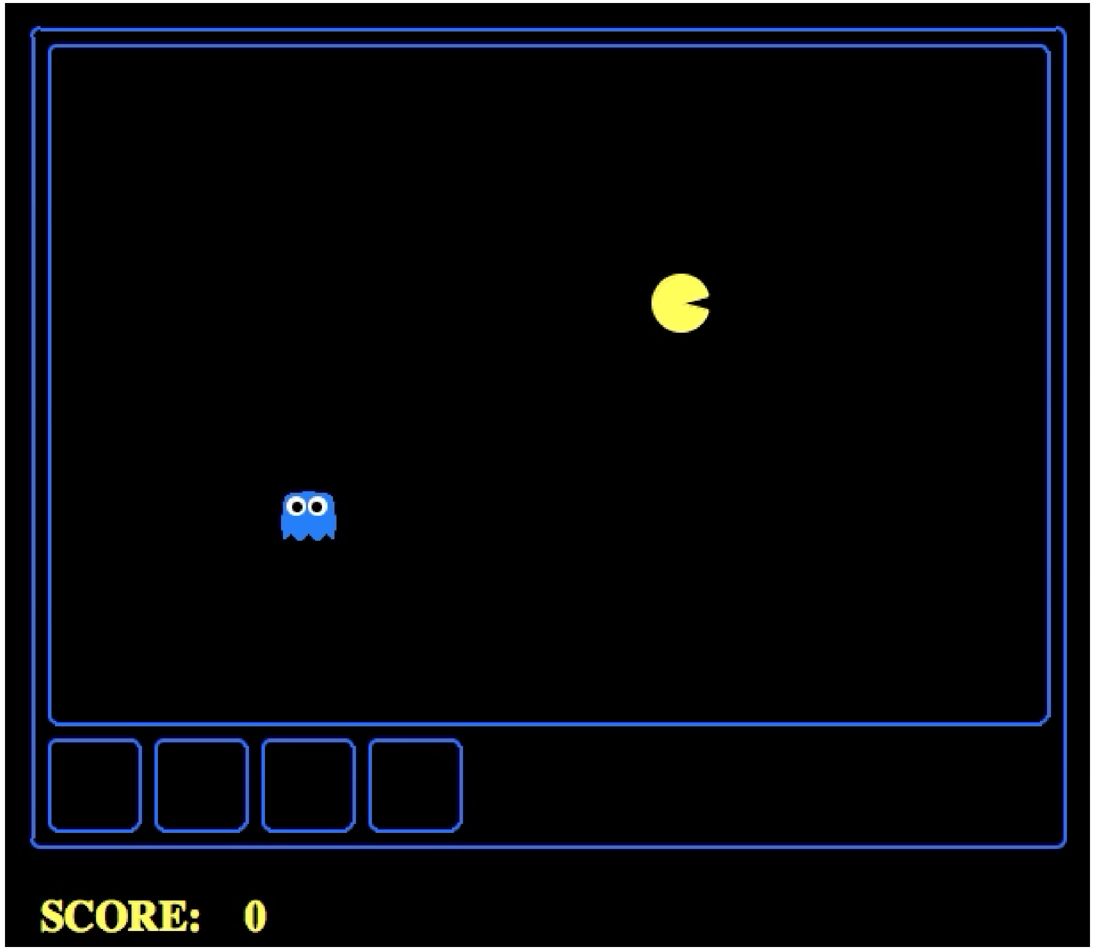
- **Plot:** Pacman's grandfather, Grandpac, learned to hunt ghosts for sport.
- He was blinded by his power, but could hear the ghosts' banging and clanging.
- **Transition Model:** All ghosts move randomly, but are sometimes biased
- **Emission Model:** Pacman knows a "noisy" distance to each ghost

Noisy distance prob
True distance = 8



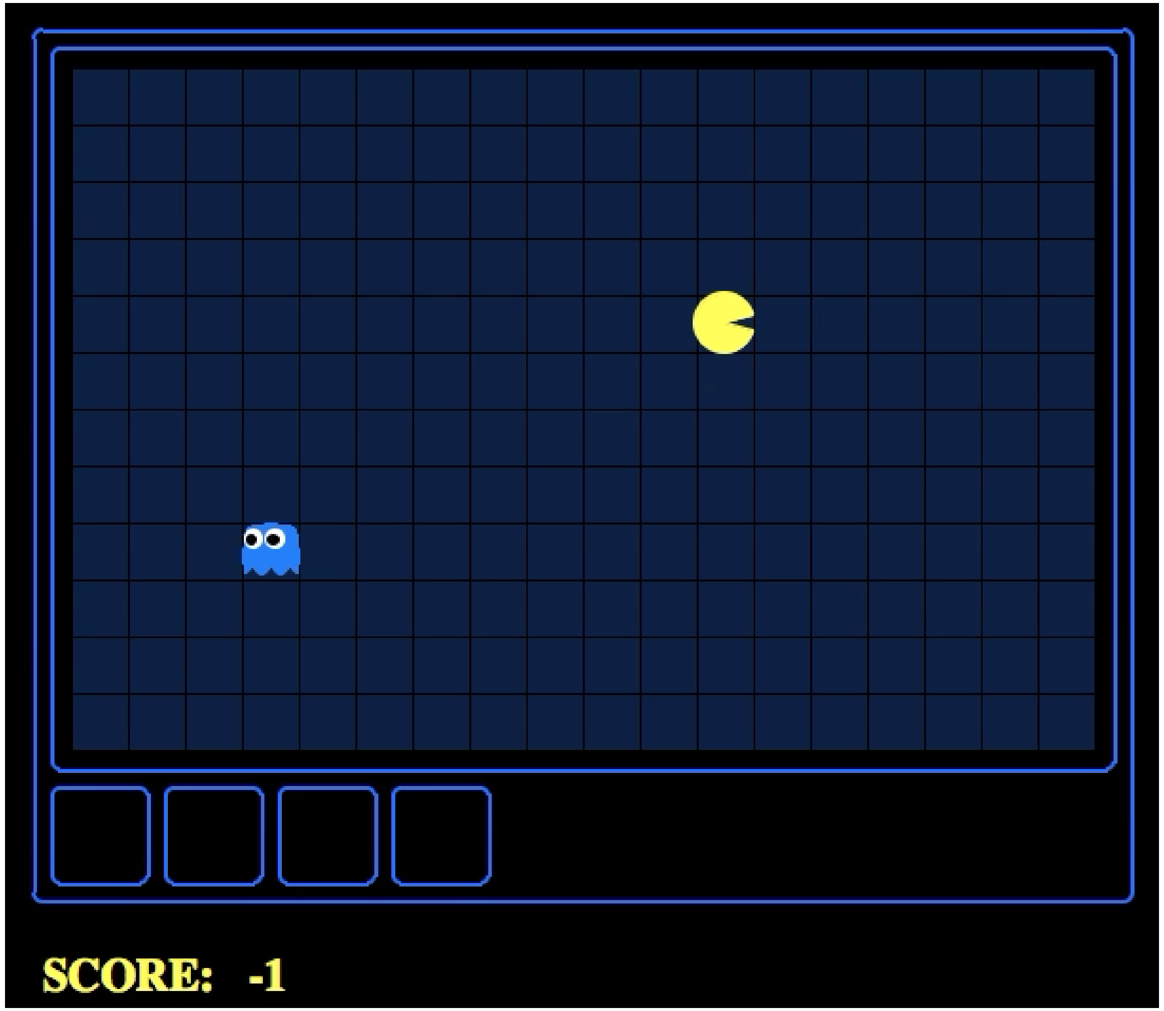
Which Algorithm?

Particle filter, uniform initial beliefs, 25 particles



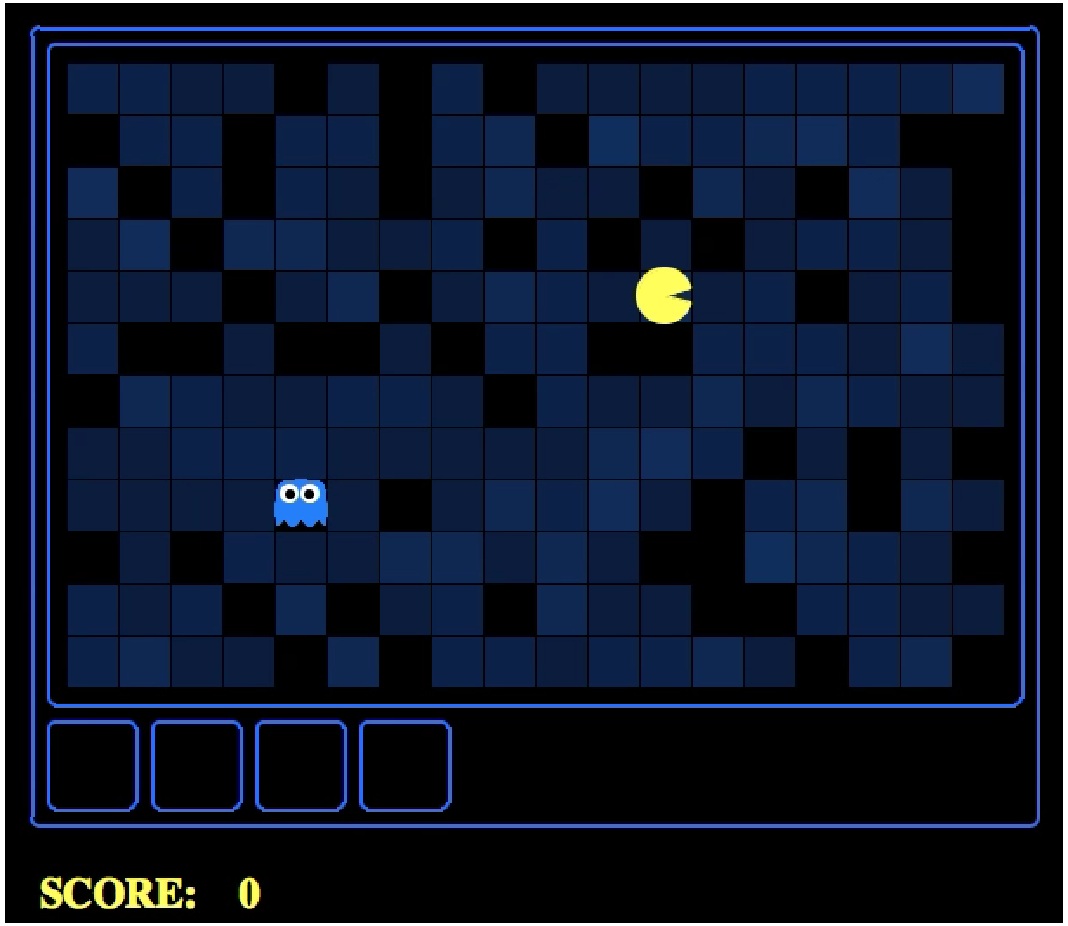
Which Algorithm?

Exact filter, uniform initial beliefs



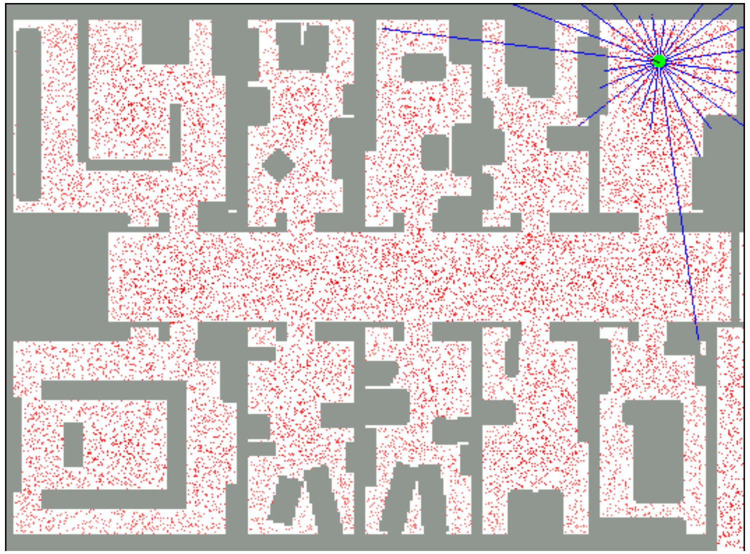
Which Algorithm?

Particle filter, uniform initial beliefs, 300 particles



Robot Localization

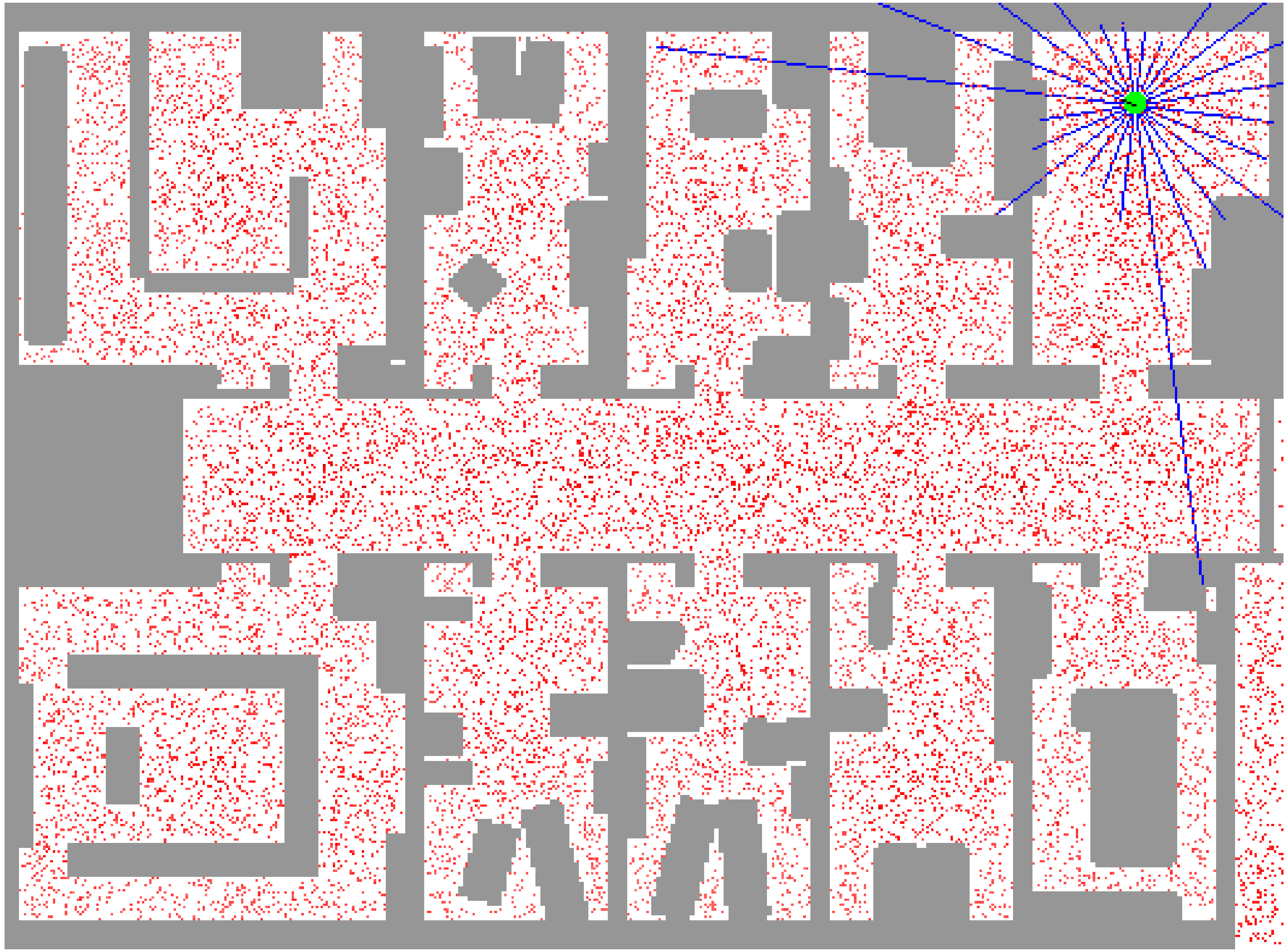
- In robot localization:
  - We know the map, but not the robot's position
  - Observations may be vectors of range finder readings
  - State space and readings are typically continuous (works basically like a very fine grid) and so we cannot store B(X)
  - Particle filtering is a main technique



Particle Filter Localization (Sonar)

[Video: global-sonar-uw-annotated.avi]

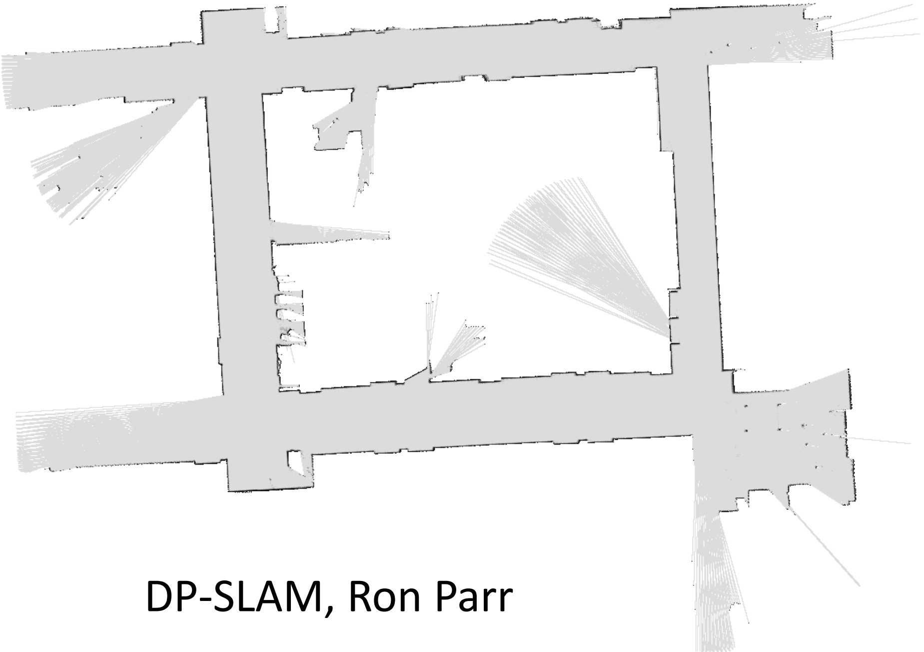
Particle Filter Localization (Laser)



[Video: global-floor.gif]

Robot Mapping

- SLAM: Simultaneous Localization And Mapping
  - We do not know the map or our location
  - State consists of position AND map!
  - Main techniques: Kalman filtering (Gaussian HMMs) and particle methods



DP-SLAM, Ron Parr

[Demo: PARTICLES-SLAM-mapping1-new.avi]