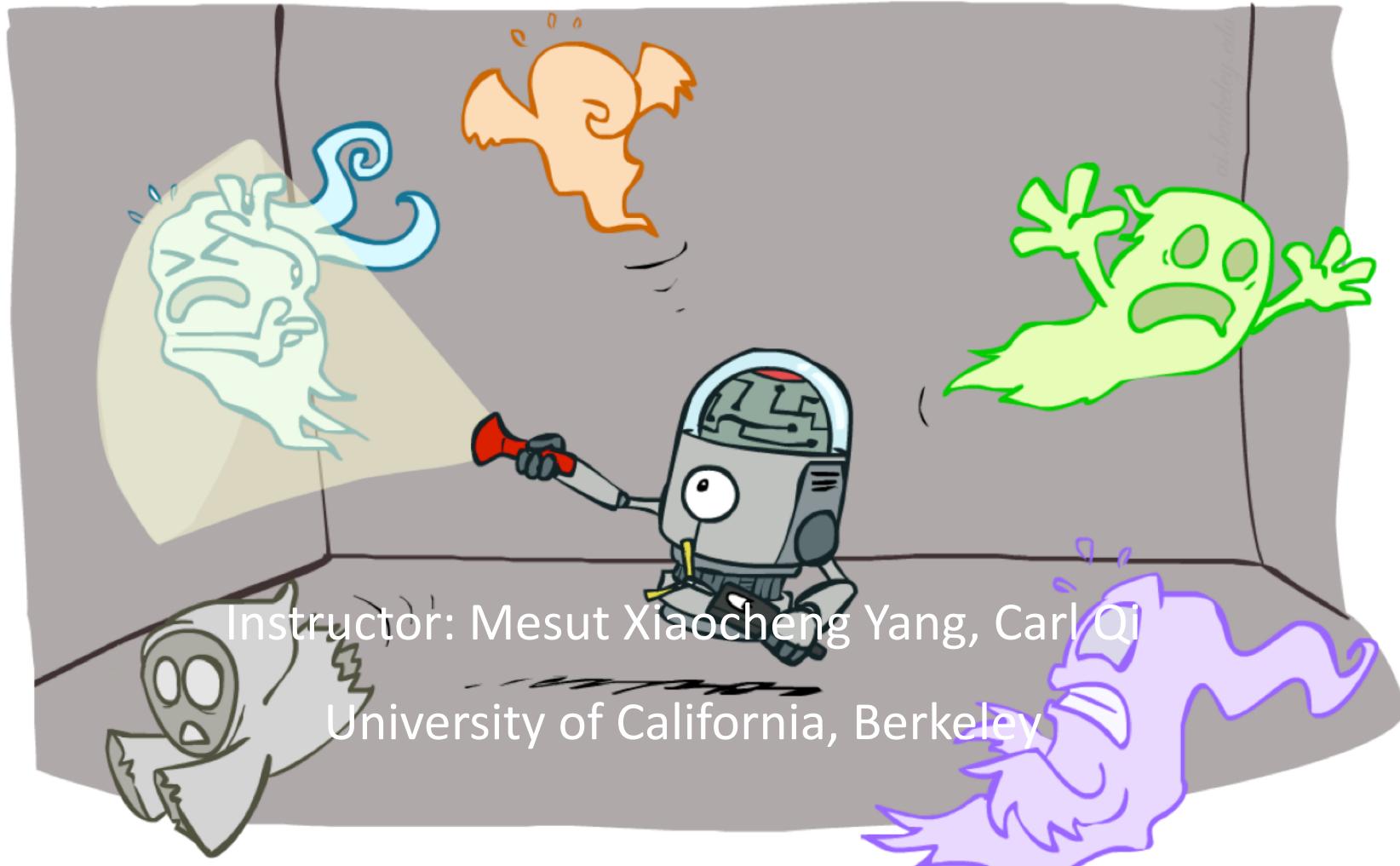


CS 188: Artificial Intelligence

HMMs, Particle Filters, and Applications



Instructor: Mesut Xiaocheng Yang, Carl Qi

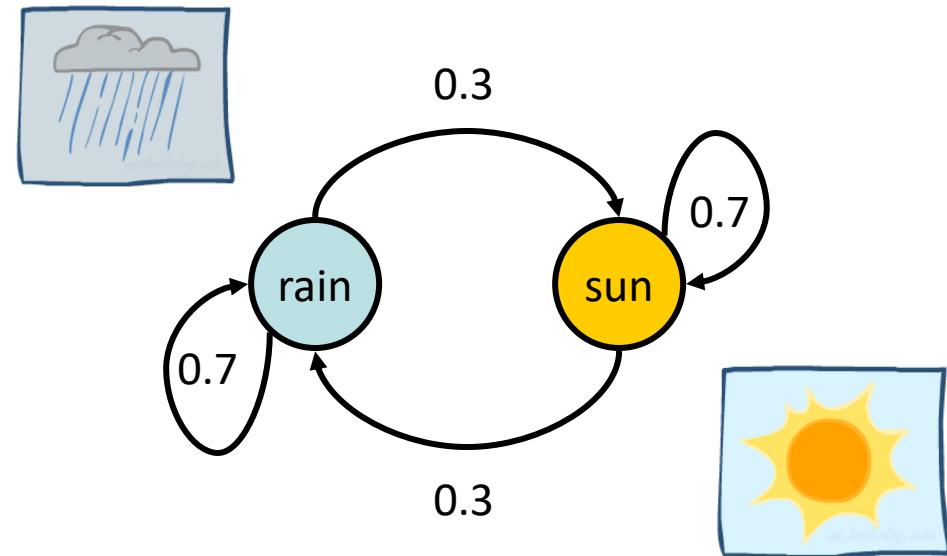
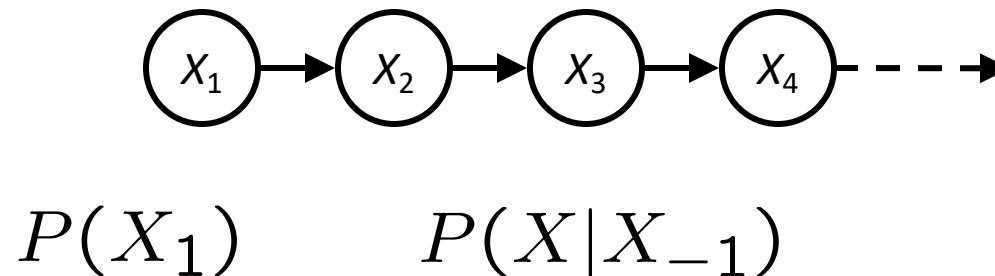
University of California, Berkeley

Today

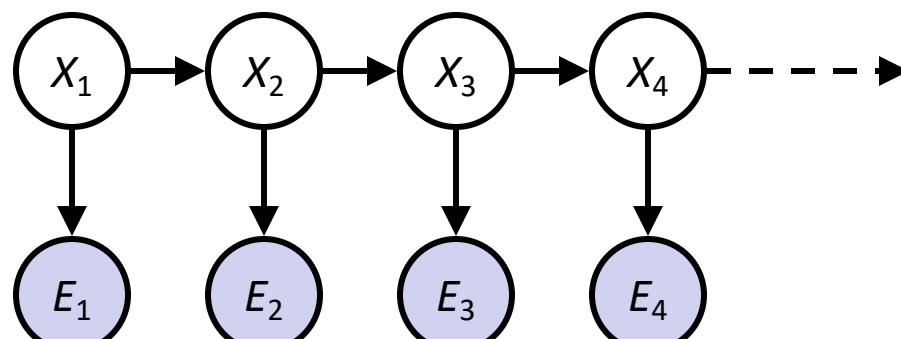
- HMMs
 - Particle filters
 - Demos!
- Applications:
 - Robot localization / mapping

Recap: Reasoning Over Time

- Markov models



- Hidden Markov models



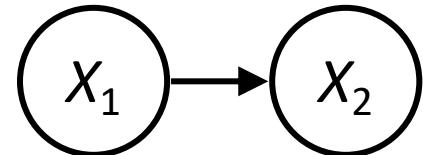
$$P(E|X)$$

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

Passage of Time

- Assume we have current belief $P(X \mid \text{evidence to date})$

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



- Then, after one time step passes:

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

- 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 $P(X \mid \text{previous evidence})$:

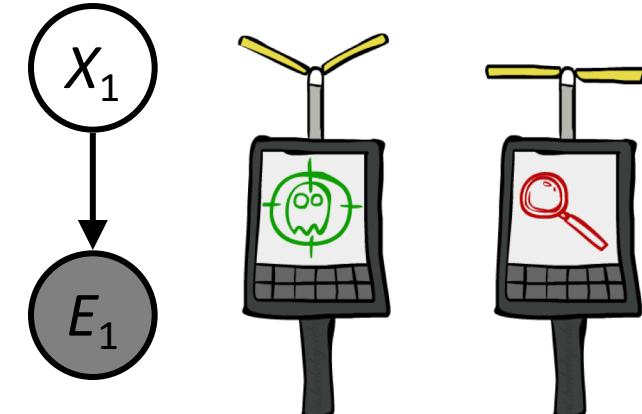
$$B'(X_{t+1}) = P(X_{t+1} | e_{1:t})$$

- Then, after evidence comes in:

$$\begin{aligned} P(X_{t+1} | e_{1:t+1}) &= P(X_{t+1}, e_{t+1} | e_{1:t}) / P(e_{t+1} | e_{1:t}) \\ &\propto_{X_{t+1}} P(X_{t+1}, e_{t+1} | e_{1:t}) \\ &= P(e_{t+1} | e_{1:t}, X_{t+1}) P(X_{t+1} | e_{1:t}) \\ &= P(e_{t+1} | X_{t+1}) B'(X_{t+1}) \end{aligned}$$

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

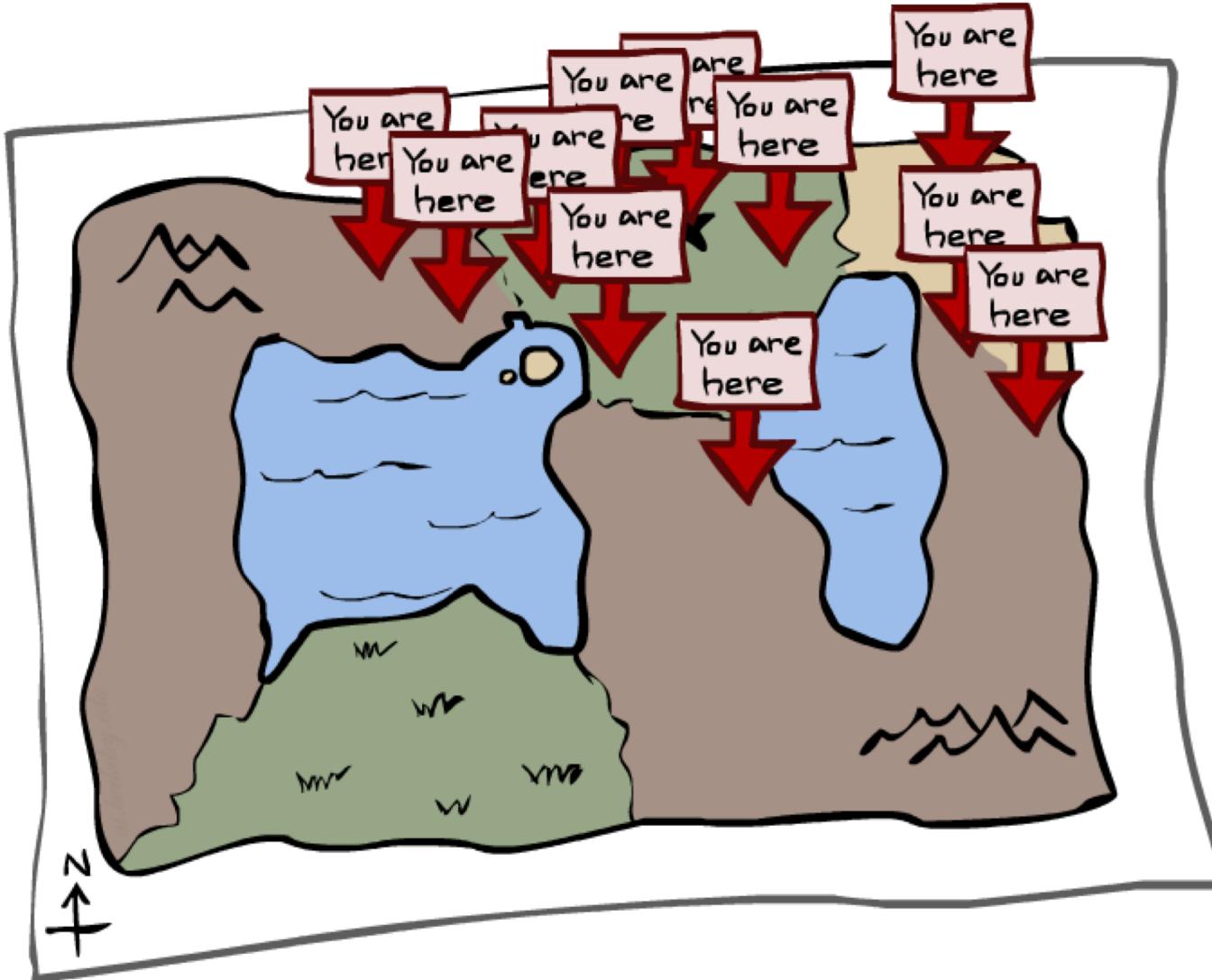
$$P(x_t | e_{1:t-1}) = \sum_{x_{t-1}} P(x_{t-1} | e_{1:t-1}) \cdot P(x_t | x_{t-1})$$

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

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

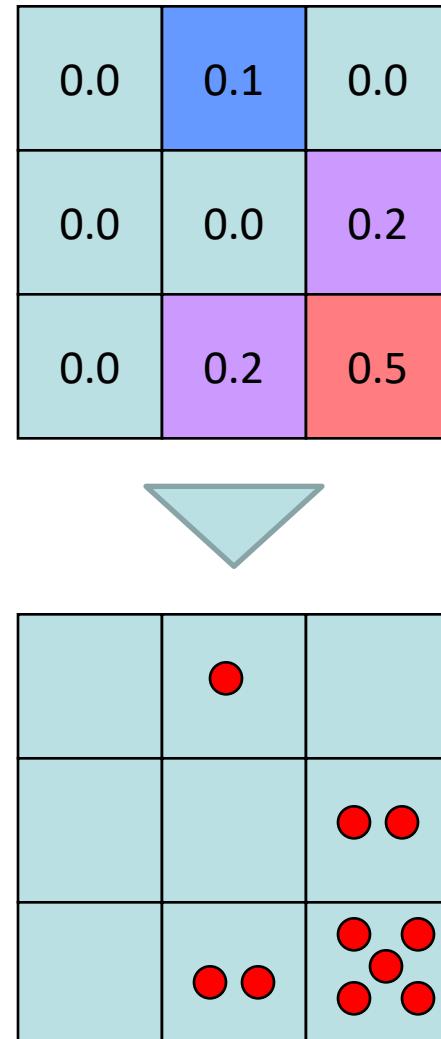
<0.01	<0.01	<0.01	<0.01	<0.01	<0.01
<0.01	<0.01	0.06	<0.01	<0.01	<0.01
<0.01	0.76	0.06	0.06	<0.01	<0.01
<0.01	<0.01	0.06	<0.01	<0.01	<0.01

Particle Filtering



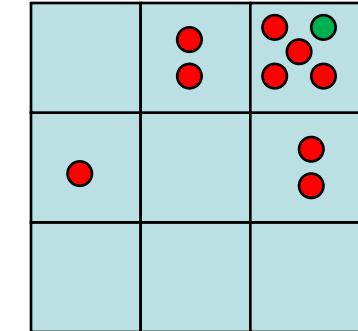
Particle Filtering

- 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
- This is how robot localization works in practice
- Particle is just new name for sample



Representation: Particles

- Our representation of $P(X)$ is now a list of N particles (samples)
 - Generally, $N \ll |X|$
 - Storing map from X to counts would defeat the point
- $P(x)$ 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

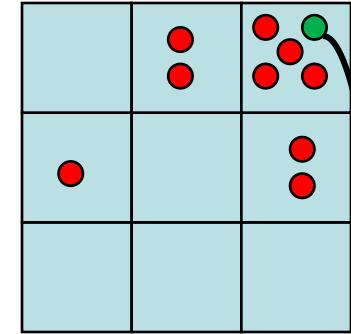
- Each particle is moved by sampling its next position from the transition model

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

- 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)

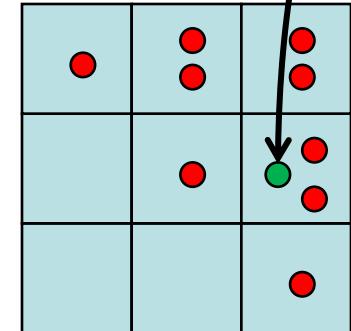
Particles:

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



Particles:

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



Particle Filtering: Observe

- Slightly trickier:

- Don't sample observation, fix it
- Similar to likelihood weighting, downweight samples based on the evidence

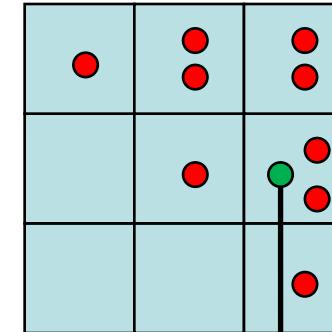
$$w(x) = P(e|x)$$

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

- 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 $P(e)$)

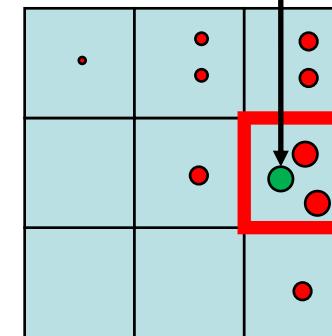
Particles:

(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

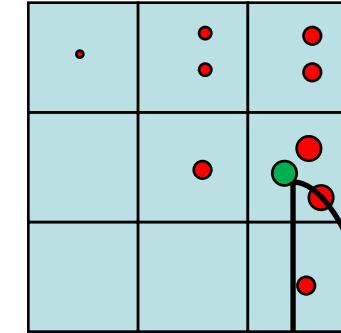


Particle Filtering: Resample

- 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

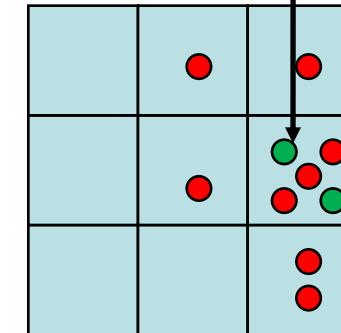
Particles:

(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



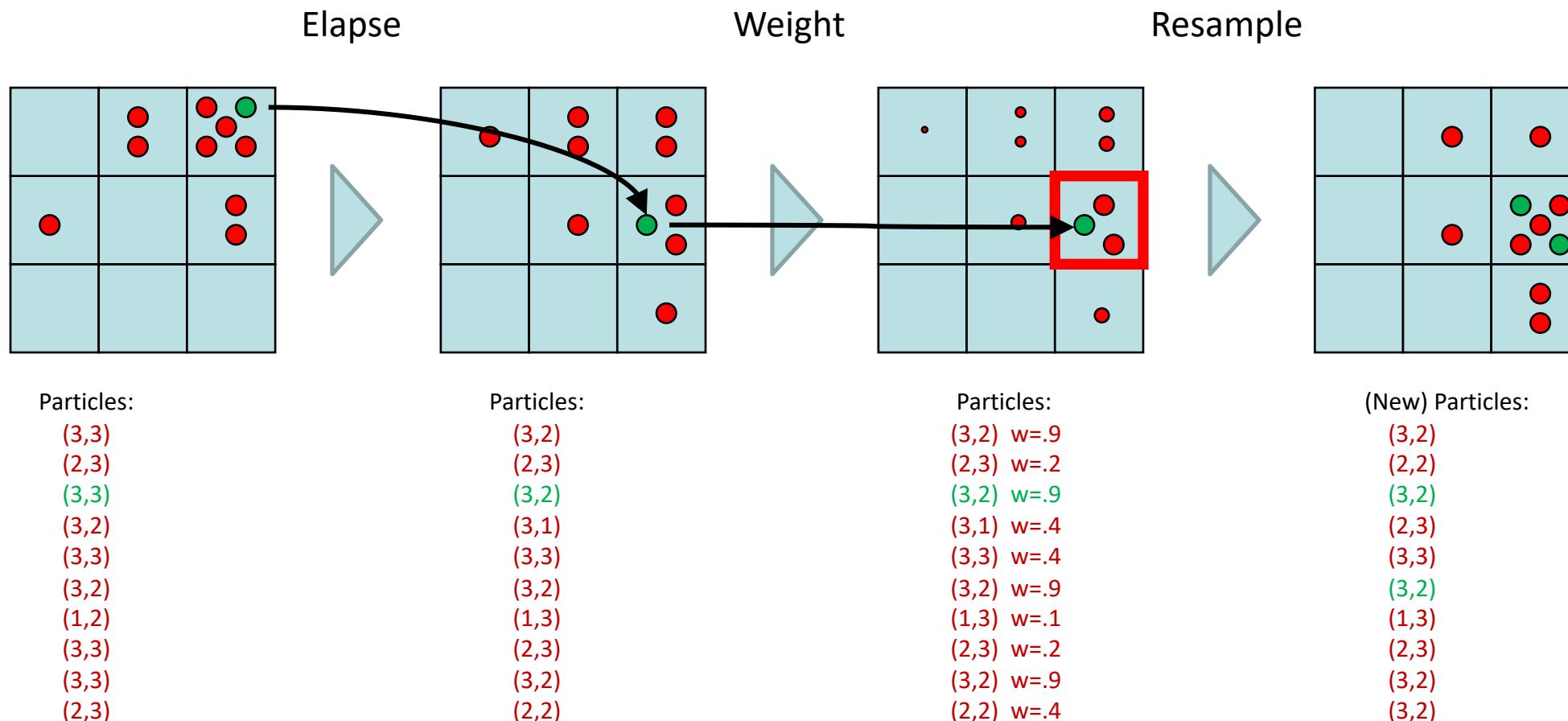
(New) Particles:

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

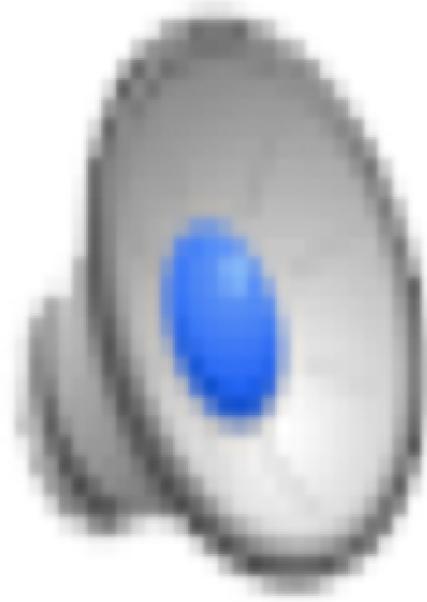


Recap: Particle Filtering

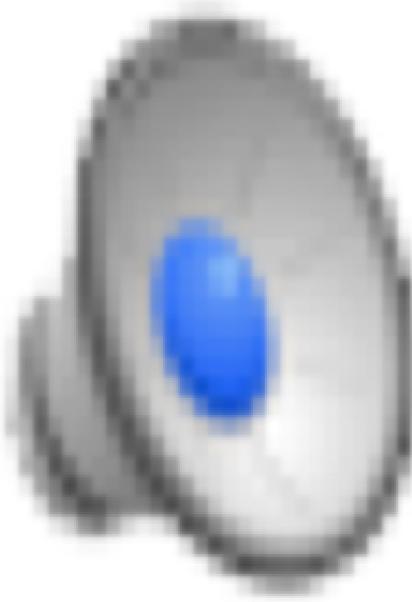
- Particles: track samples of states rather than an explicit distribution



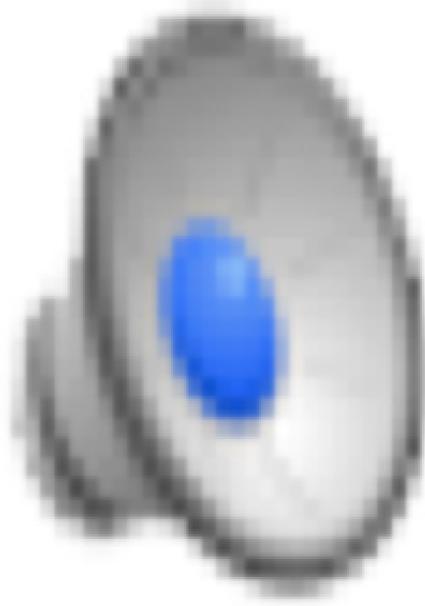
Video of Demo – Moderate Number of Particles



Video of Demo – One Particle



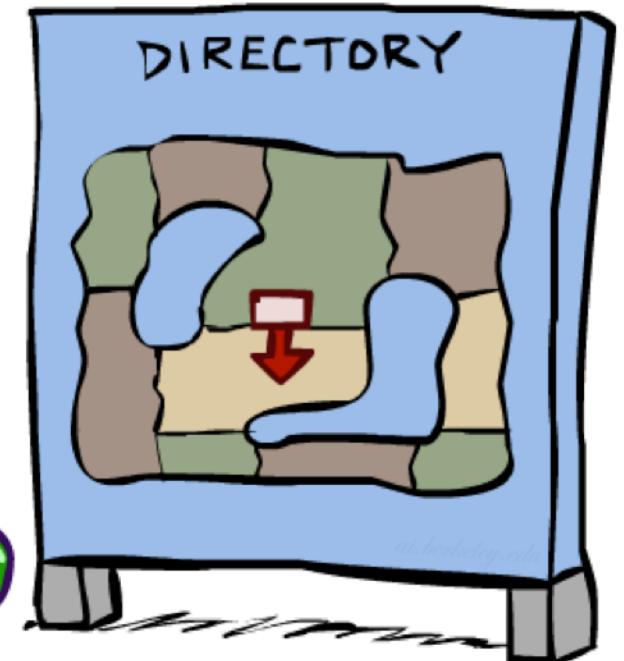
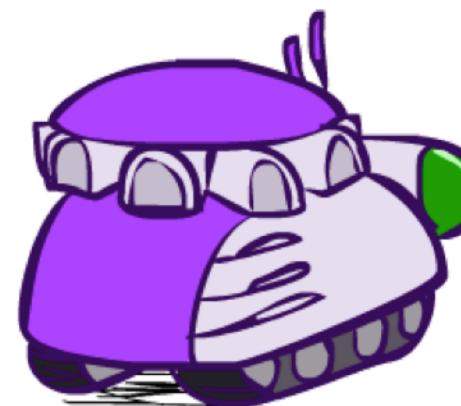
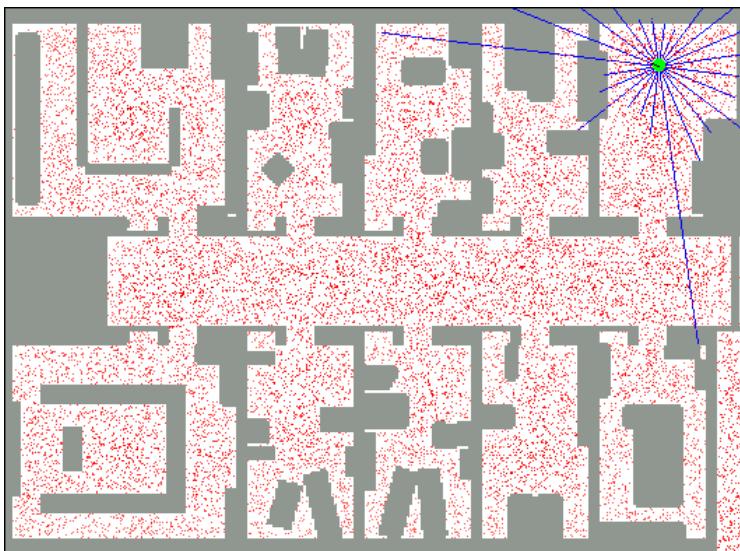
Video of Demo – Huge Number of Particles



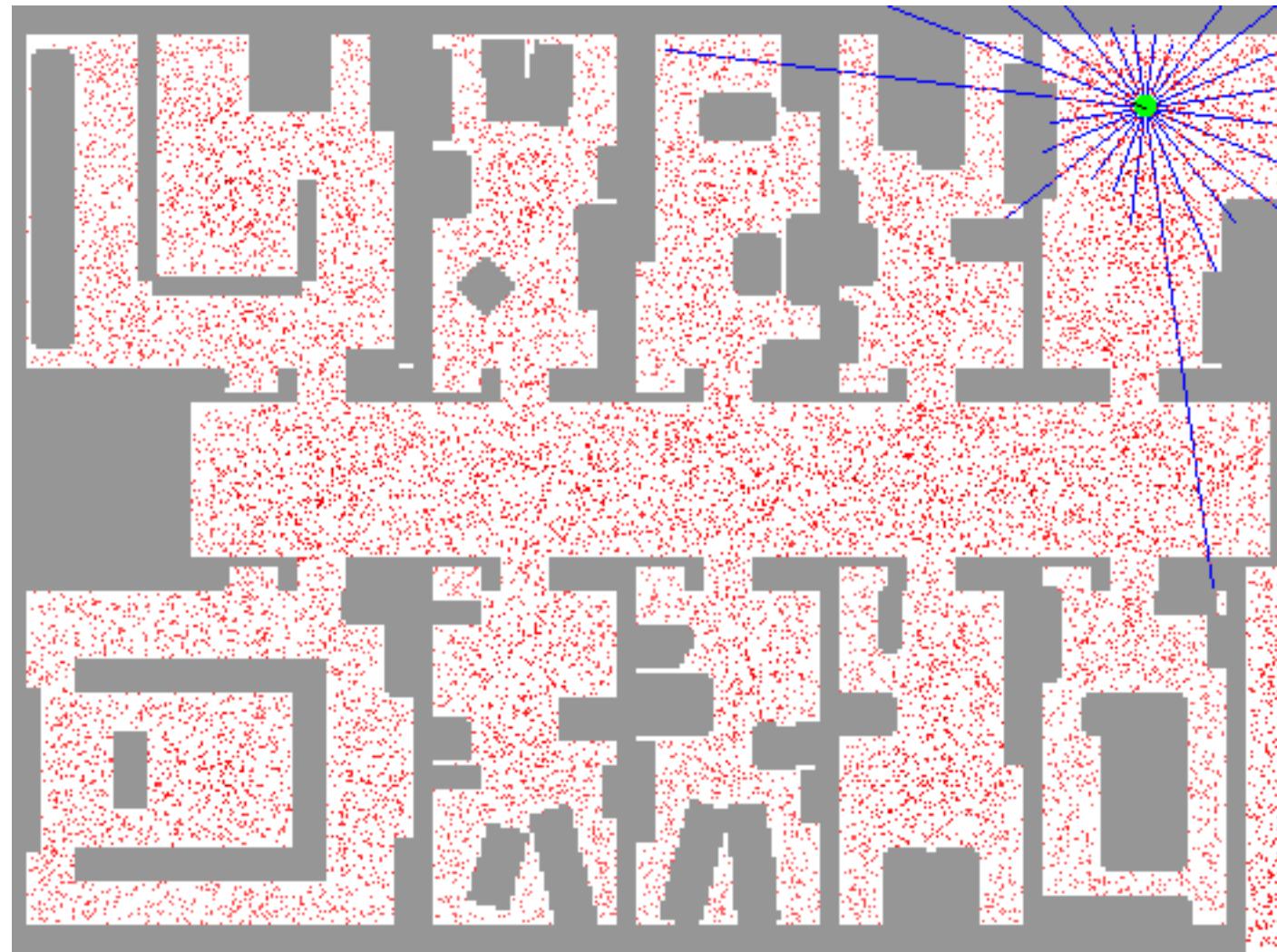
Robot Localization

- 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 (Laser)

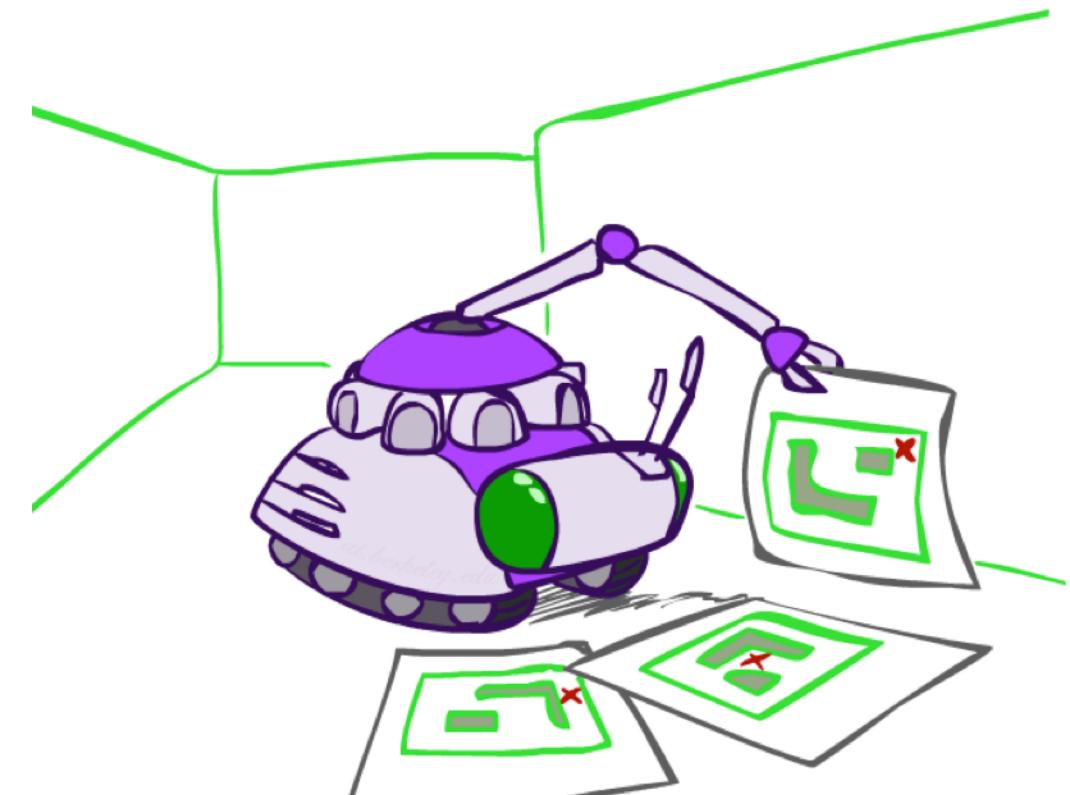
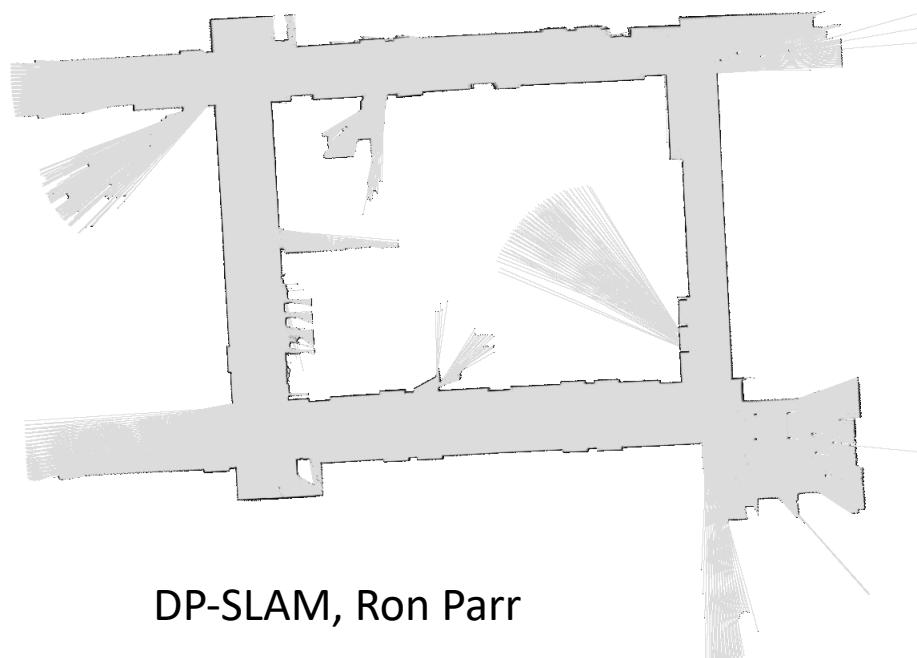


[Video: global-floor.gif]

Robot Mapping

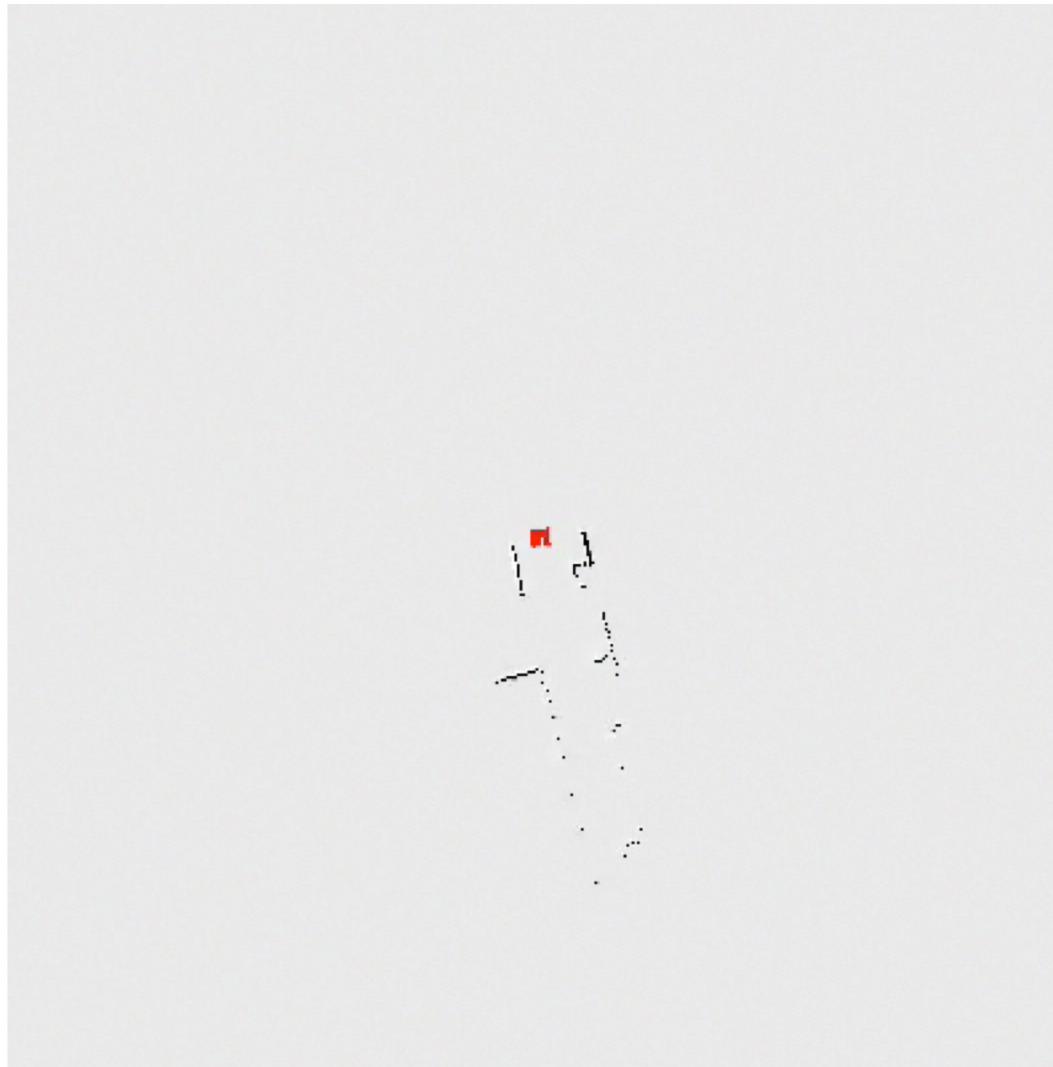
- SLAM: Simultaneous Localization And Mapping

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



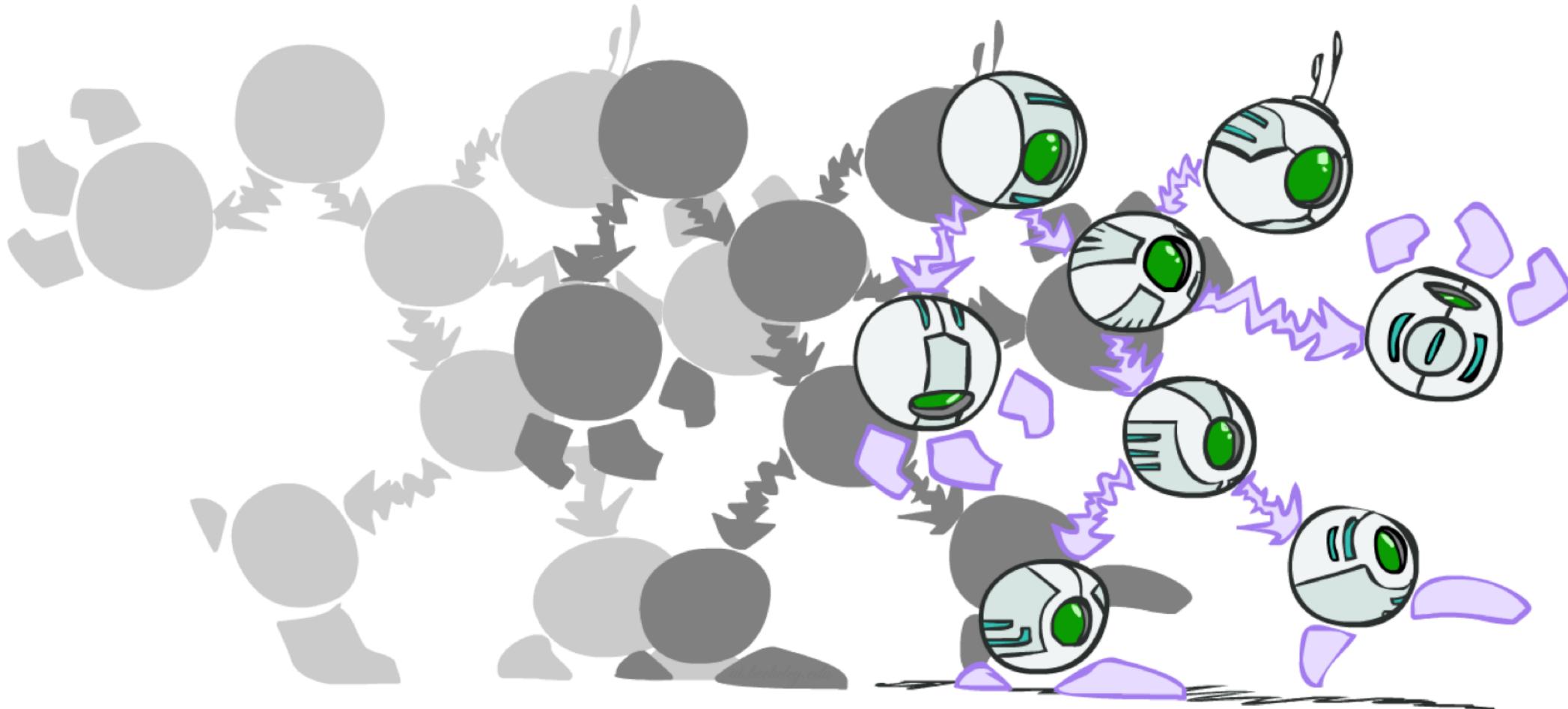
[Demo: PARTICLES-SLAM-mapping1-new.avi]

Particle Filter SLAM – Video 1



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

Dynamic Bayes Nets



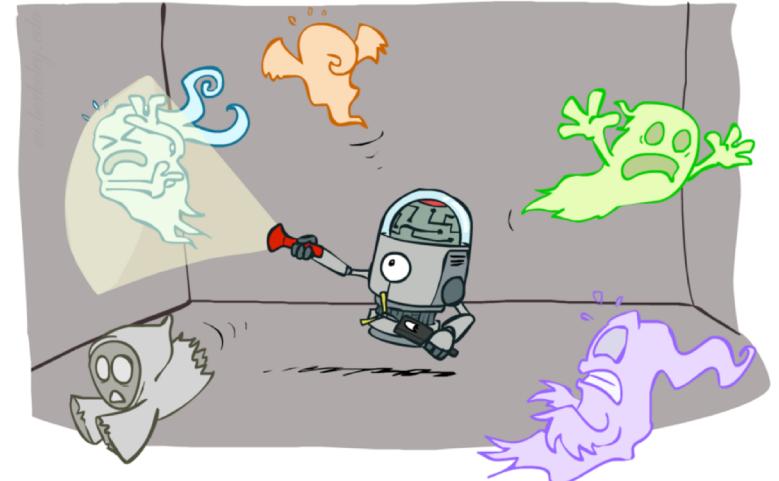
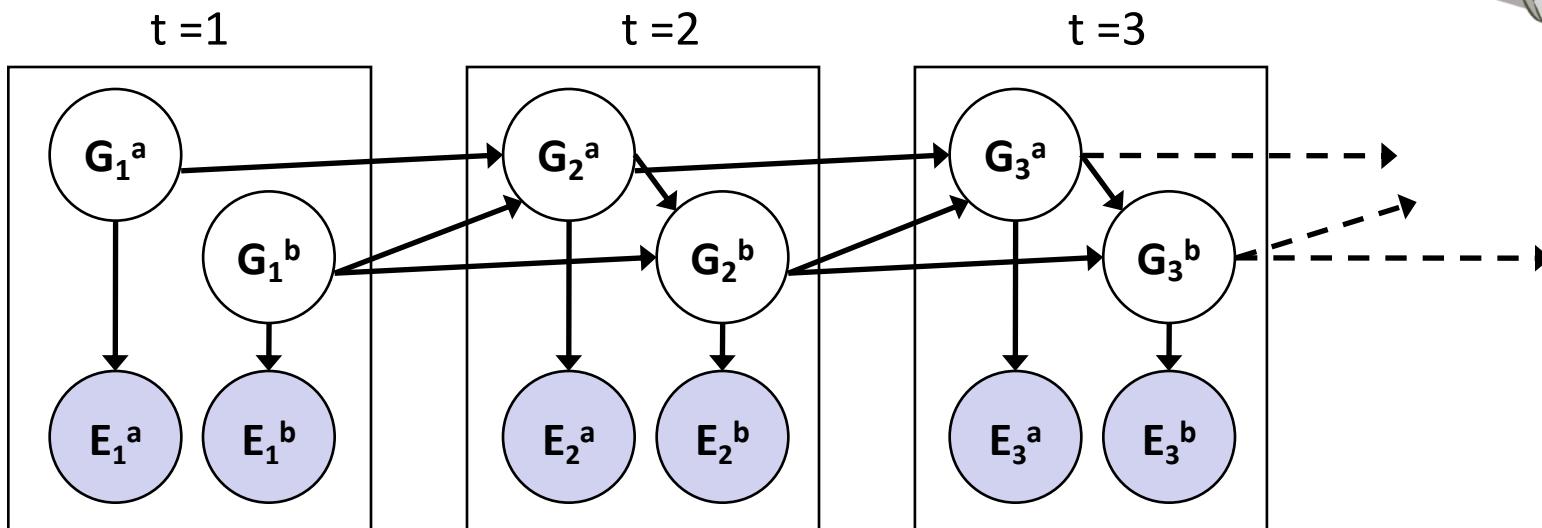
Pacman – Sonar (P4)



[Demo: Pacman – Sonar – No Beliefs(L14D1)]

Dynamic Bayes Nets (DBNs)

- We want to track multiple variables over time, using multiple sources of evidence
- Idea: Repeat a fixed Bayes net structure at each time
- Variables from time t can condition on those from $t-1$

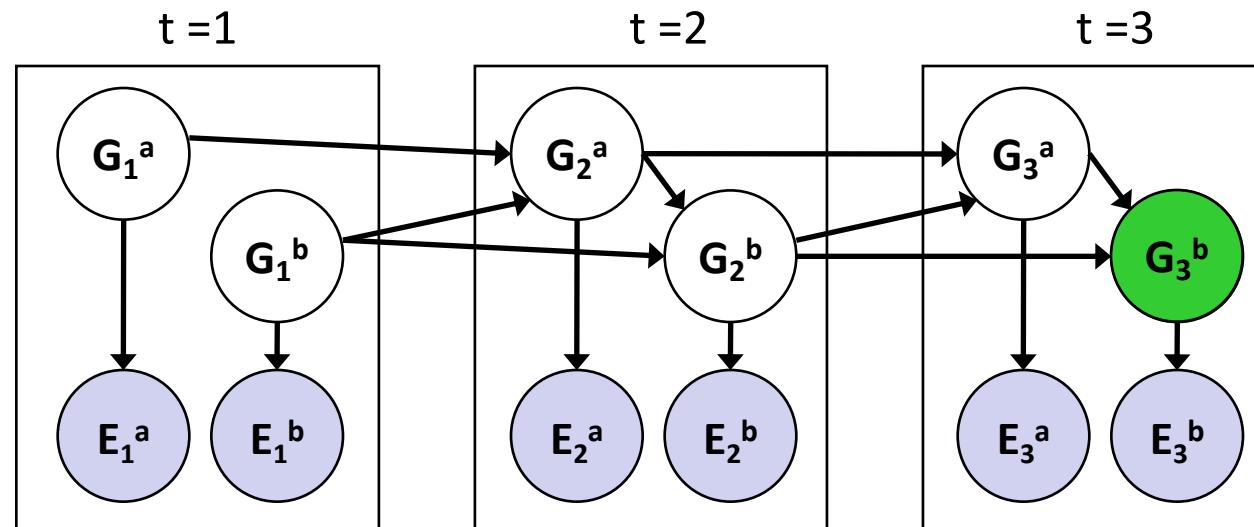


- Dynamic Bayes nets are a generalization of HMMs

[Demo: pacman sonar ghost DBN model (L15D6)]

Exact Inference in DBNs

- Variable elimination applies to dynamic Bayes nets
- Procedure: “unroll” the network for T time steps, then eliminate variables until $P(X_T | e_{1:T})$ is computed



- Online belief updates: Eliminate all variables from the previous time step; store factors for current time only

DBN Particle Filters

- A particle is a complete sample for a time step
- **Initialize:** Generate prior samples for the t=1 Bayes net
 - Example particle: $\mathbf{G}_1^a = (3,3)$ $\mathbf{G}_1^b = (5,3)$
- **Elapse time:** Sample a successor for each particle
 - Example successor: $\mathbf{G}_2^a = (2,3)$ $\mathbf{G}_2^b = (6,3)$
- **Observe:** Weight each entire sample by the likelihood of the evidence conditioned on the sample
 - Likelihood: $P(E_1^a | \mathbf{G}_1^a) * P(E_1^b | \mathbf{G}_1^b)$
- **Resample:** Select prior samples (tuples of values) in proportion to their likelihood

Video of Demo Pacman – Sonar (with beliefs)

