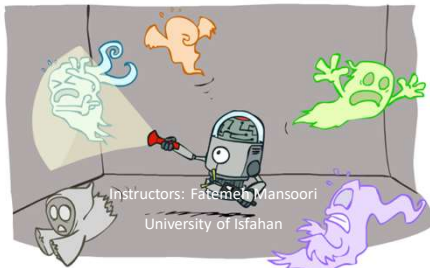


## Advanced Artificial Intelligence

### HMMs, Particle Filters



(These slides were created by Dan Klein and Pieter Abbeel for CS188 Intro to AI at UC Berkeley. All CS188 materials are available at <http://ai.berkeley.edu>.)

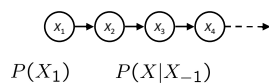
## Today

- HMMs
  - Particle filters
  - Demos!
  - Most-likely-explanation queries
- Applications:
  - Robot localization / mapping
  - Speech recognition (later)

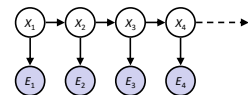
## Recap: Reasoning Over Time

[Demo: Ghostbusters Markov Model (L15D1)]

### 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

## Inference: Base Cases



$P(X_1|e_1)$

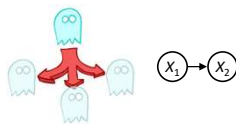
$$P(x_1|e_1) = P(x_1, e_1) / P(e_1) \\ \propto_{X_1} P(x_1, e_1) \\ = P(x_1)P(e_1|x_1)$$



$P(X_2)$

$$P(x_2) = \sum_{x_1} P(x_1, x_2) \\ \propto_{X_1} P(x_1, e_1) \\ = \sum_{x_1} P(x_1)P(x_2|x_1)$$

## Inference: Base Cases



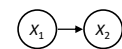
$P(X_2)$

$$P(x_2) = \sum_{x_1} P(x_1, x_2) \\ = \sum_{x_1} P(x_1)P(x_2|x_1)$$

## Passage of Time

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

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



- 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:

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

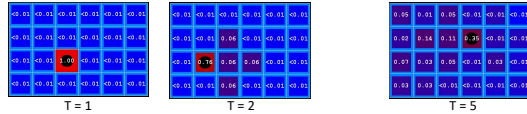
- Basic idea: beliefs get "pushed" through the transitions

- With the "B" notation, we have to be careful about what time step t the belief is about, and what evidence it includes

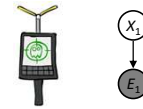
## Example: Passage of Time

- As time passes, uncertainty “accumulates”

(Transition model: ghosts usually go clockwise)



## Inference: Base Cases



$$P(X_1|e_1)$$

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

## Observation

- Assume we have current belief  $P(X | \text{previous evidence})$ :

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

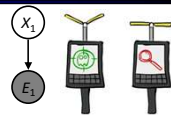
- Then, after evidence comes in:

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

- Or, compactly:

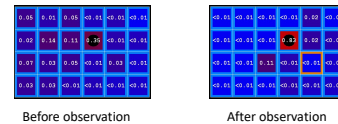
$$B(X_{t+1}) \propto_{X_{t+1}} P(e_{t+1}|X_{t+1})B'(X_{t+1})$$

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



## Example: Observation

- As we get observations, beliefs get reweighted, uncertainty “decreases”



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

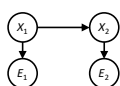
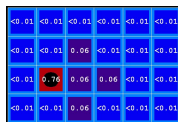
## Filtering

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

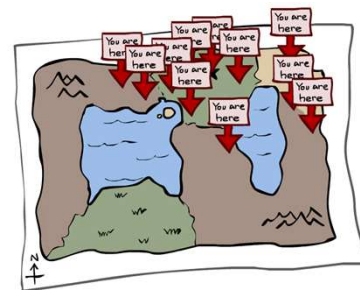
$$P(x_t|e_{1:t}) \propto P(x_t|e_{1:t-1}) \cdot P(e_t|x_t)$$



$$\begin{aligned} \text{Belief: } \langle P(\text{rain}), P(\text{sun}) \rangle \\ P(X_1) &\langle 0.5, 0.5 \rangle && \text{Prior on } X_1 \\ P(X_1 | E_1 = \text{umbrella}) &\langle 0.82, 0.18 \rangle && \text{Observe} \\ P(X_2 | E_1 = \text{umbrella}) &\langle 0.63, 0.37 \rangle && \text{Elapse time} \\ P(X_2 | E_1 = \text{umb}, E_2 = \text{umb}) &\langle 0.88, 0.12 \rangle && \text{Observe} \end{aligned}$$

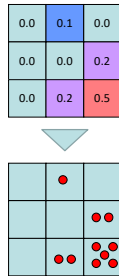
[Demo: Ghostbusters Exact Filtering (L15D2)]

## Particle Filtering



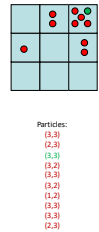
## Particle Filtering

- 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



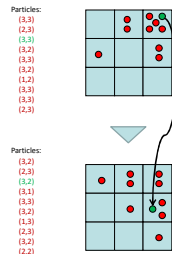
## Representation: Particles

- 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



## 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)



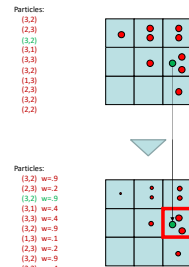
## Particle Filtering: Observe

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

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

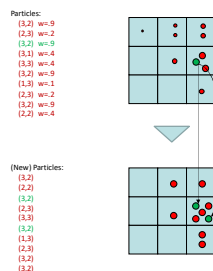
$$B(X) \propto P(e|X)B'(X)$$

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



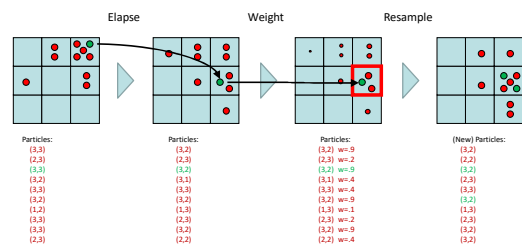
## Particle Filtering: Resample

- 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



## Recap: Particle Filtering

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

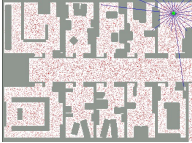


[Demos: ghostbusters particle filtering (L15D3.4.5)]

## 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



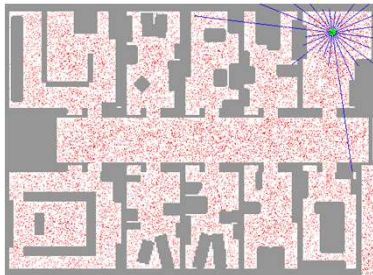
## Particle Filter Localization (Sonar)

### Global localization with sonar sensors



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

## Particle Filter Localization (Laser)

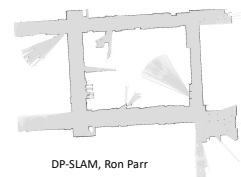


[Video: global-floor.gif]

## 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



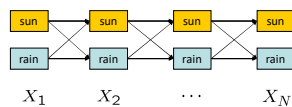
DP-SLAM, Ron Parr



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

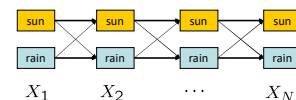
## State Trellis

- State trellis: graph of states and transitions over time



- Each arc represents some transition  $x_{t-1} \rightarrow x_t$
- Each arc has weight  $P(x_t|x_{t-1})P(e_t|x_t)$
- Each path is a sequence of states
- The product of weights on a path is that sequence's probability along with the evidence
- Forward algorithm computes sums of paths, Viterbi computes best paths

## Forward / Viterbi Algorithms



### Forward Algorithm (Sum)

$$f_t[x_t] = P(x_t, e_{1:t})$$

$$= P(e_t|x_t) \sum_{x_{t-1}} P(x_t|x_{t-1}) f_{t-1}[x_{t-1}]$$

### Viterbi Algorithm (Max)

$$m_t[x_t] = \max_{x_{1:t-1}} P(x_{1:t-1}, x_t, e_{1:t})$$

$$= P(e_t|x_t) \max_{x_{t-1}} P(x_t|x_{t-1}) m_{t-1}[x_{t-1}]$$