

CS 188: Artificial Intelligence Spring 2010

Lecture 20: HMMs and Particle Filtering 4/5/2010

Pieter Abbeel --- UC Berkeley
Many slides over this course adapted from Dan Klein, Stuart Russell,
Andrew Moore

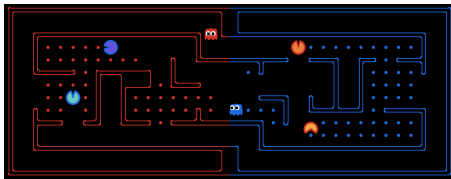
Mid-Semester Evals

- Generally, things seem good!
- General
 - Examples are appreciated in lecture
 - Favorite aspect: projects (almost all) --- writings significantly less preferred
- Office hours:
 - Most common answers: "Helpful." and "Haven't gone."
 - Some: too crowded. → perhaps try a different office hour slot
- Section:
 - Split between basically positive and don't go
- Assignments
 - Written: median time 6hrs
 - Programming: median time 10hrs
 - Some people spend a lot more time though → come talk to us if you are stuck
- Exams:
 - Midterm: evening (13) vs in-class (11) or indifferent (8)
- Want to do the contest

2

Contest

- Course contest



- Fun! (And extra credit.)
- Regular tournaments
- Instructions posted soon!

3

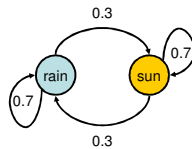
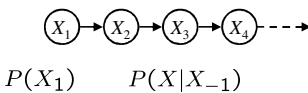
Outline

- HMMs: representation
- HMMs: inference
 - Forward algorithm
 - Particle filtering

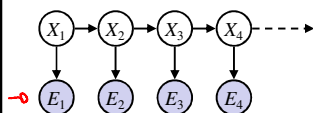
8

Recap: Reasoning Over Time

- Stationary Markov models



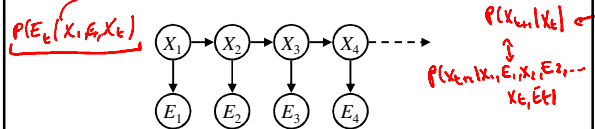
- Hidden Markov models



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

Conditional Independence

- HMMs have two important independence properties:
 - Markov hidden process, future depends on past via the present
 - Current observation independent of all else given current state



- Quiz: does this mean that observations are independent given no evidence?
 - [No, correlated by the hidden state]

Real HMM Examples

- **Speech recognition HMMs:**
 - Observations are acoustic signals (continuous valued)
 - States are specific positions in specific words (so, tens of thousands)
- **Machine translation HMMs:**
 - Observations are words (tens of thousands)
 - States are translation options
- **Robot tracking:**
 - Observations are range readings (continuous)
 - States are positions on a map (continuous)

Outline

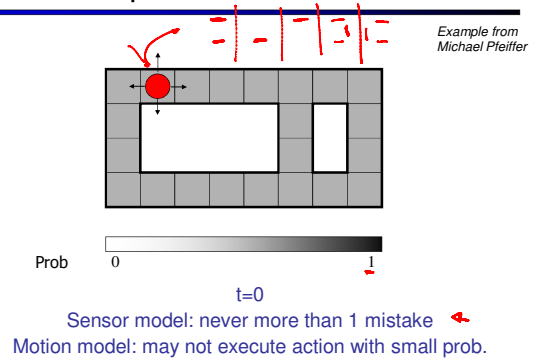
- HMMs: representation
- HMMs: inference
 - Forward algorithm
 - Particle filtering

13

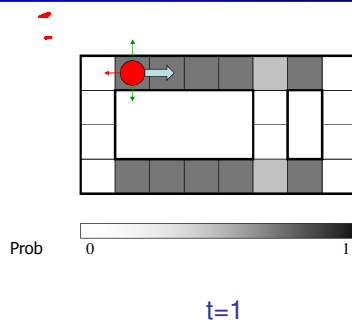
Filtering / Monitoring

- Filtering, or monitoring, is the task of tracking the distribution $B(X)$ (the belief state) over time
- We start with $B(X)$ in an initial setting, usually uniform
- As time passes, or we get observations, we update $B(X)$
- The Kalman filter was invented in the 60's and first implemented as a method of trajectory estimation for the Apollo program

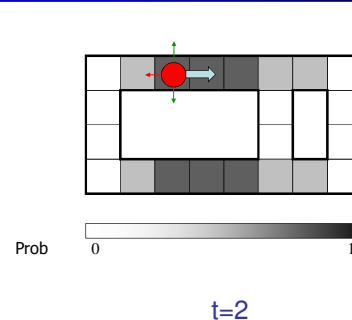
Example: Robot Localization

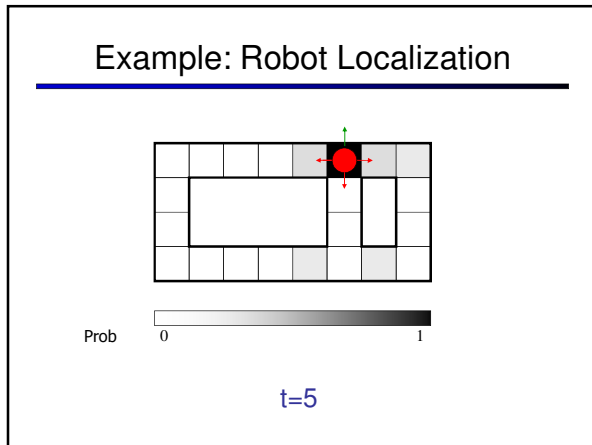
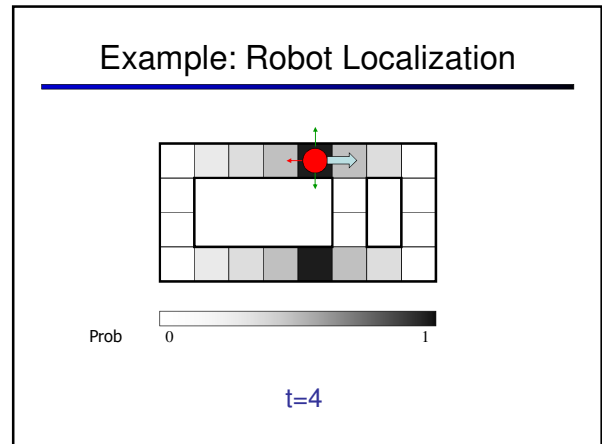
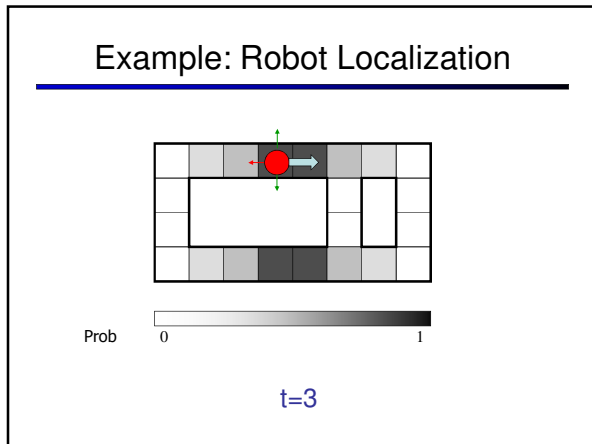


Example: Robot Localization



Example: Robot Localization





Inference Recap: Simple Cases

$$P(X_1|e_1)$$

$$P(x_1|e_1) = P(x_1, e_1) / P(e_1)$$

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

$$= P(x_1) P(e_1|x_1)$$

sense model

$$P(X_2)$$

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

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

transition model

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:

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

- Or, compactly:

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

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

Example: Passage of Time

- As time passes, uncertainty "accumulates"

T = 1

T = 2

T = 5

$$B'(X') = \sum_x P(X'|x) B(x)$$

Transition model: ghosts usually go clockwise

Observation

$B'(X_t) = P(X_t | e_{1:t-1})$

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

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

- Then:

$$P(X_{t+1} | e_{1:t+1}) \propto P(e_{t+1} | X_{t+1}) P(X_{t+1} | e_{1:t})$$
- Or:

$$B(X_{t+1}) \propto P(e_t | X) B'(X_{t+1})$$

- 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"

Before observation

After observation

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

Example HMM

The Forward Algorithm

- We are given evidence at each time and want to know

$$B_t(X) = P(X_t | e_{1:t})$$
- We can derive the following updates

$$P(x_t | e_{1:t}) \propto_X P(x_t, e_{1:t})$$

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

$$= \sum_{x_{t-1}} \underbrace{P(x_{t-1}, e_{1:t-1})}_{\text{We can normalize as we go if we want to have } P(x_t) \text{ at each time step, or just once at the end...}} \underbrace{P(x_t | x_{t-1})}_{\text{Transition}} \underbrace{P(e_t | x_t)}_{\text{Emission}}$$

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

Online Belief Updates

- Every time step, we start with current $P(X | \text{evidence})$
- We update for time:

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

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

- The forward algorithm does both at once (and doesn't normalize)
- Problem: space is $|X|$ and time is $|X|^2$ per time step

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

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

| Belief: <P(rain), P(sun)> | | |
|---|--------------|----------------|
| $P(X_1)$ | <0.5, 0.5> | Prior on X_1 |
| $P(X_1 E_1 = \text{umbrella})$ | <0.82, 0.18> | Observe |
| $P(X_2 E_1 = \text{umbrella})$ | <0.63, 0.37> | Elapse time |
| $P(X_2 E_1 = \text{umb}, E_2 = \text{umb})$ | <0.88, 0.12> | Observe |

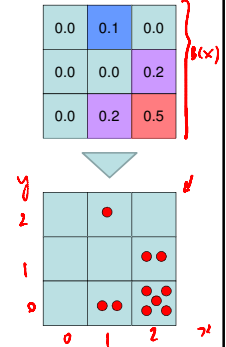
Outline

- HMMs: representation
- HMMs: inference
 - Forward algorithm
 - Particle filtering

31

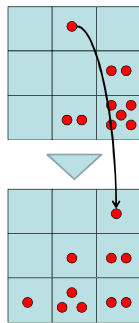
Particle Filtering

- 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 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 probs
 - Here, most samples move clockwise, but some move in another direction or stay in place
- This captures the passage of time
 - If we have enough samples, close to the exact values before and after (consistent)



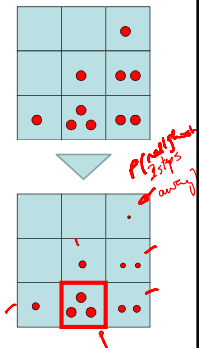
Particle Filtering: Observe

- Slightly trickier:
 - We don't sample the observation, we fix it
 - This is similar to likelihood weighting, so we downweight our samples based on the evidence

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

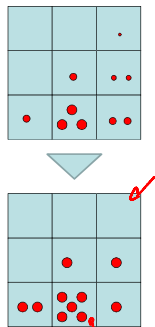
$$\rightarrow B(X) \propto P(e|X)B'(X)$$

- Note that, as before, the probabilities don't sum to one, since most have been downweighted (in fact they sum to an approximation of $P(e)$)



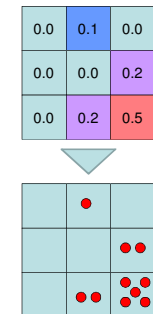
Particle Filtering: Resample

- 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



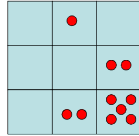
Particle Filtering

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



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

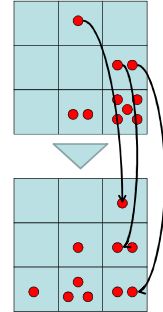
37

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 probs
- Here, most samples move clockwise, but some move in another direction or stay in place
- This captures the passage of time
 - If we have enough samples, close to the exact values before and after (consistent)



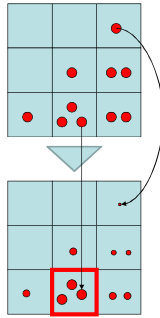
Particle Filtering: Observe

- Slightly trickier:
 - Don't do rejection sampling (why not?)
 - We don't sample the observation, we fix it
 - This is similar to likelihood weighting, so we downweight our samples based on the evidence

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

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

- Note that, as before, the probabilities don't sum to one, since most have been downweighted (in fact they sum to an approximation of $P(e)$)

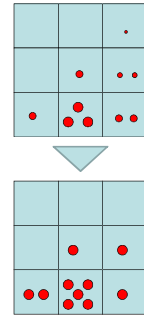


Particle Filtering: Resample

- 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

Old Particles:
 (3,3) $w=0.1$
 (2,1) $w=0.9$
 (2,1) $w=0.9$
 (3,1) $w=0.4$
 (3,2) $w=0.3$
 (2,2) $w=0.4$
 (1,1) $w=0.4$
 (3,1) $w=0.4$
 (2,1) $w=0.9$
 (3,2) $w=0.3$

Old Particles:
 (2,1) $w=1$
 (2,1) $w=1$
 (2,1) $w=1$
 (3,2) $w=1$
 (2,2) $w=1$
 (2,1) $w=1$
 (1,1) $w=1$
 (3,1) $w=1$
 (2,1) $w=1$
 (1,1) $w=1$



Naive: sample according to renormalized weights

Commonly used:



choose random number between $(0, \frac{1}{4})$

41

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

- [Demos]

