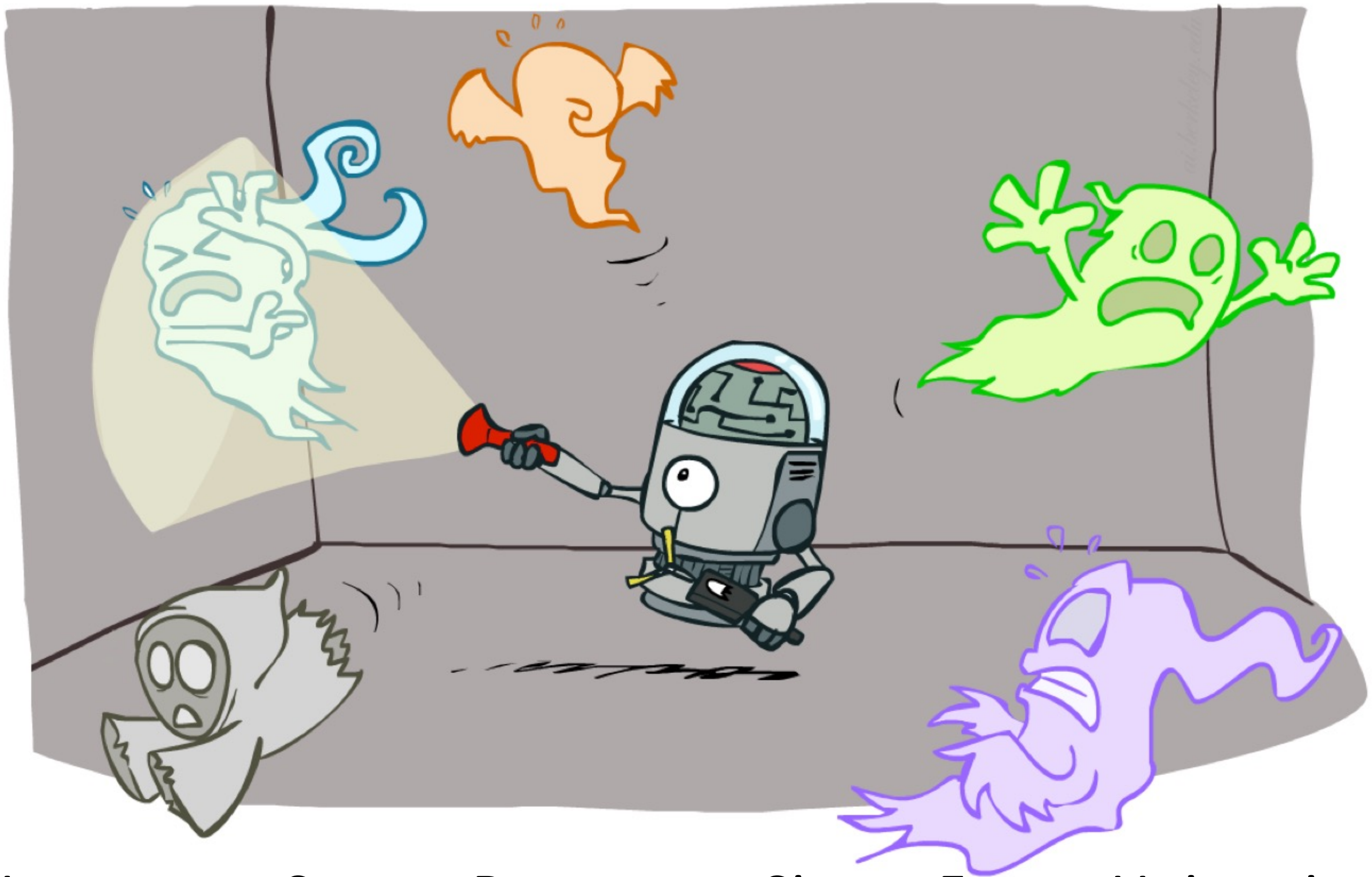
CMPT 310: Artificial Intelligence

Particle Filters and Applications of HMMs

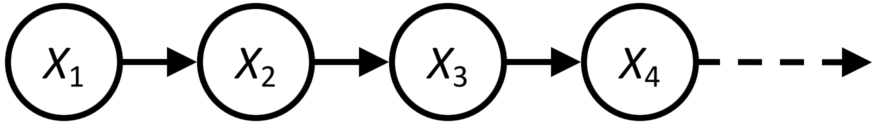


Instructor: Steven Bergner --- Simon Fraser University

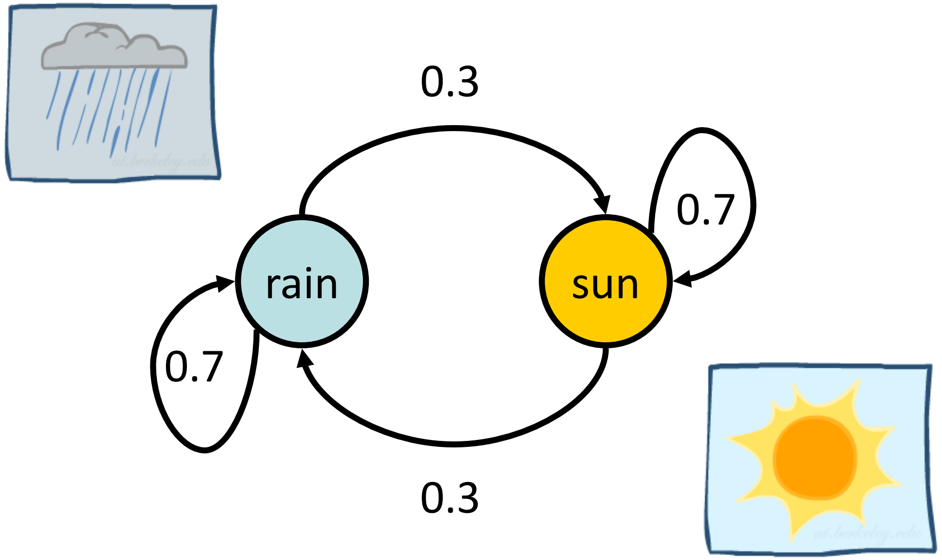
[These slides were created by Dan Klein and Pieter Abbeel for CS188 Intro to AI at UC Berkeley. All CS188 materials are available at http://ai.berkeley.edu.]

Recap: Reasoning Over Time

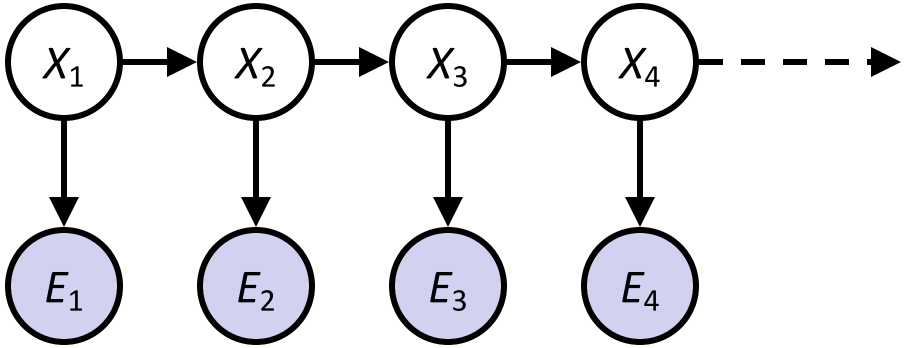
- Markov models



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



- 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

Video of Demo Ghostbusters Markov Model (Reminder)

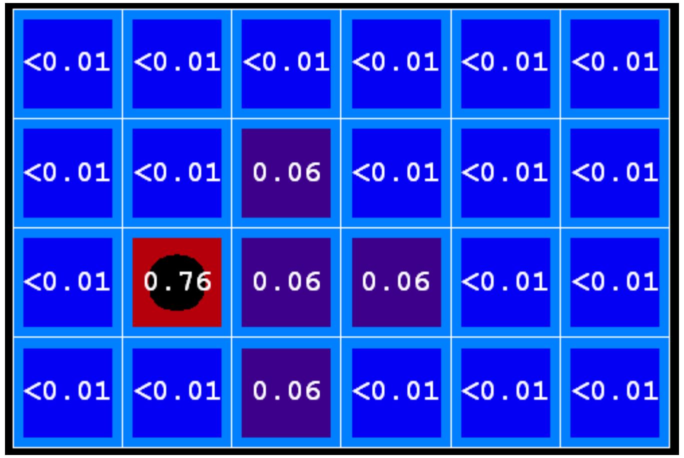
Recap: Filtering

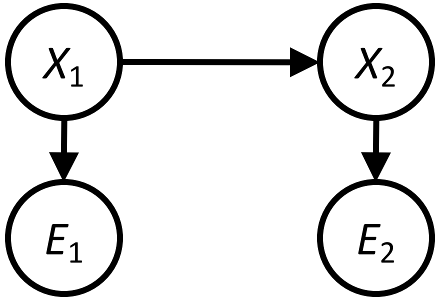
Elapse time: compute P(X_t | $e_{1:t-1}$)

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

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

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





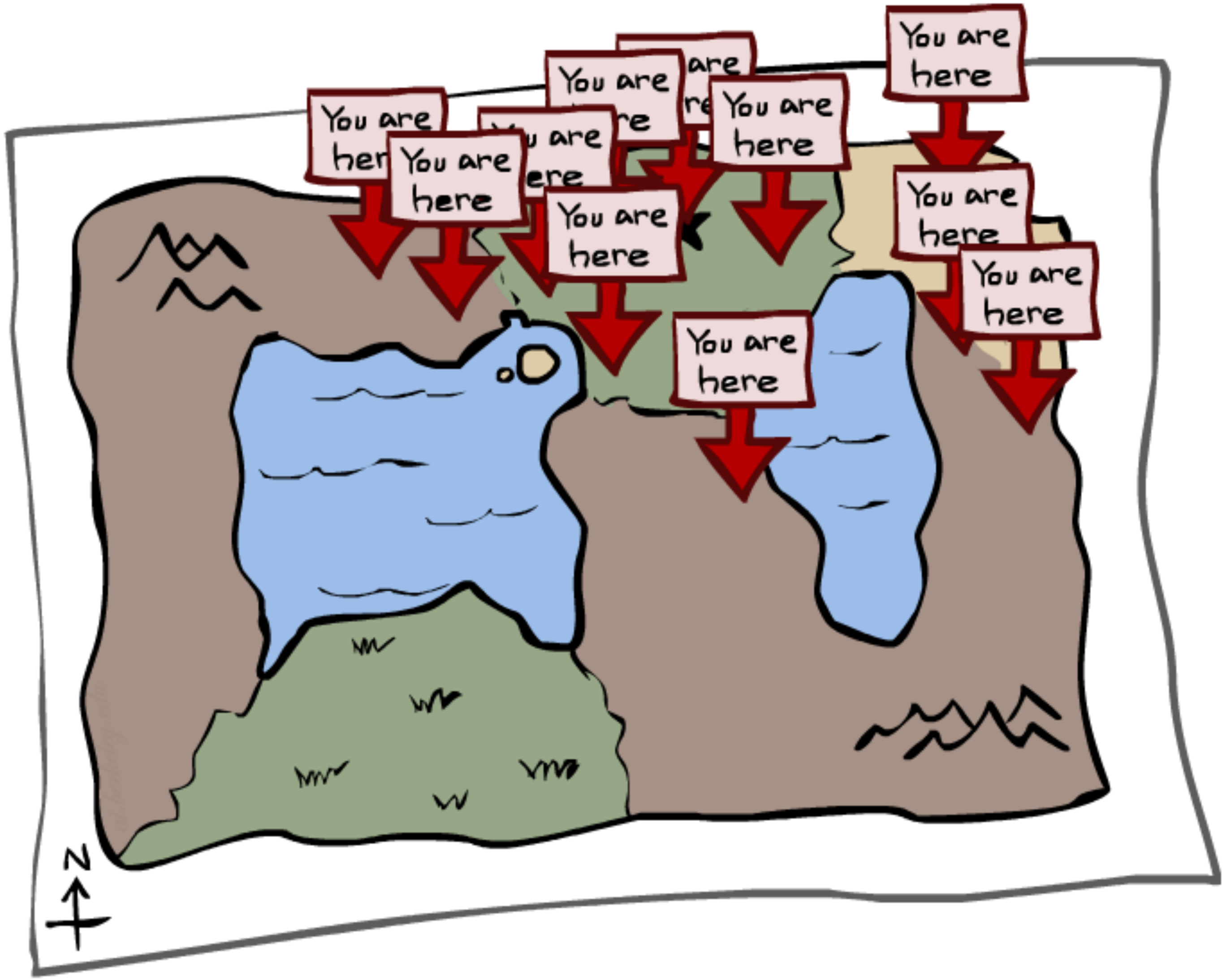
Belief: <P(rain), P(sun)>

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

[Demo: Ghostbusters Exact Filtering (L15D2)]

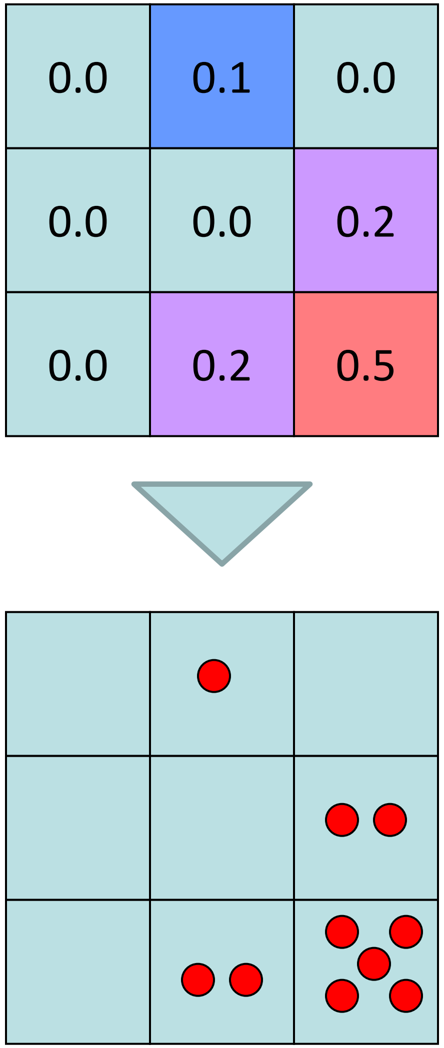
Video of Ghostbusters Exact Filtering (Reminder)

Particle Filtering



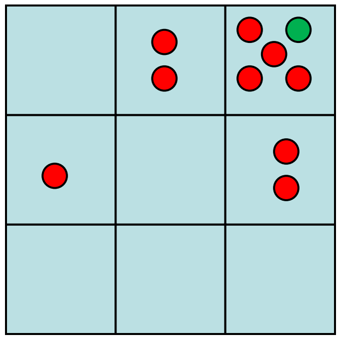
Particle Filtering

- Filtering: approximate solution
- Sometimes |X| is too big to use exact inference
  - |X| may be too big to even store P(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 << |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

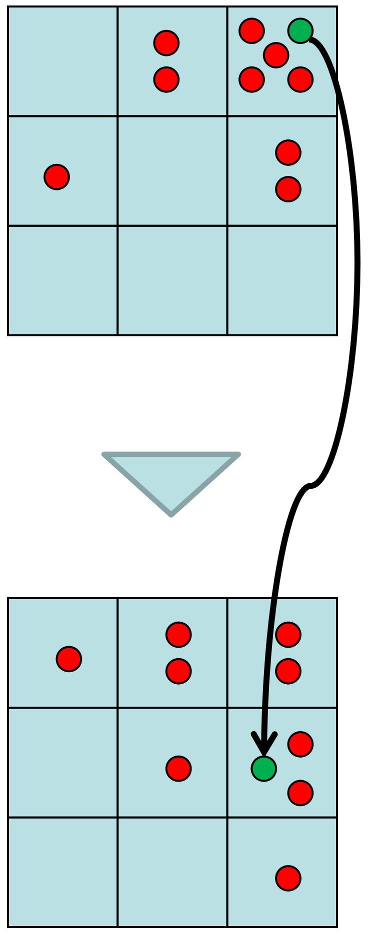
$$x' = \text{sample}(P(X'|x))$$

  - 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

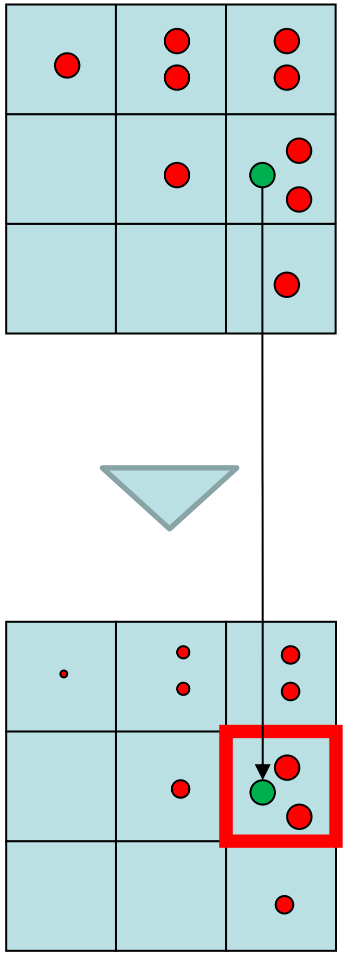
$$w(x) = P(e|x)$$

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

  - 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

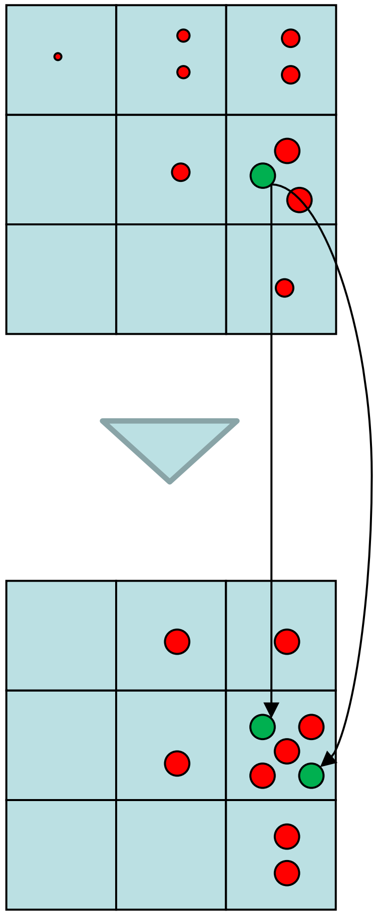
- Rather than tracking weighted samples, we resample
- N times, we choose from our weighted sample distribution (i.e. draw with replacement)
- This is equivalent to renormalizing the distribution
- Now the update is complete for this time step, continue with the next one

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

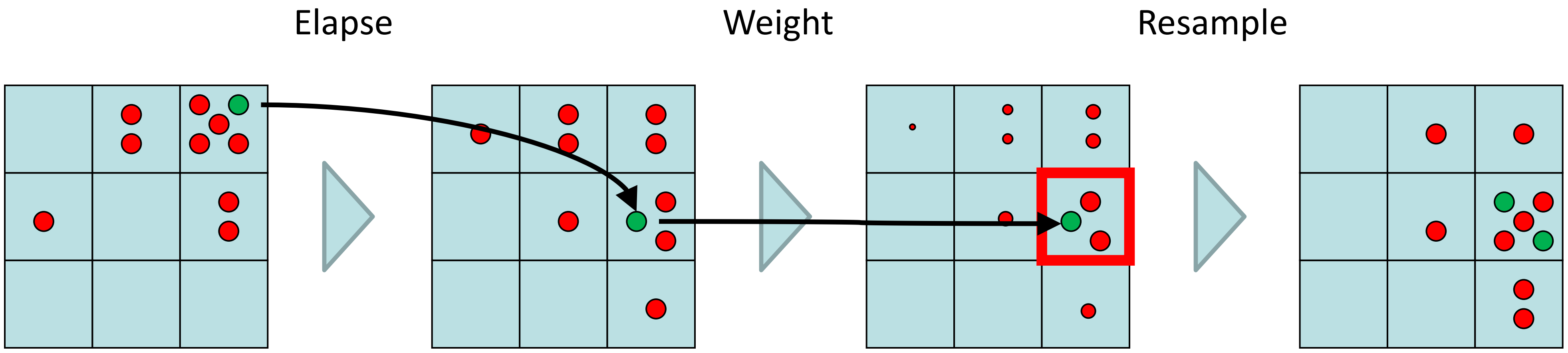
(New) Particles:

(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)

[Demos: ghostbusters particle filtering (L15D3,4,5)]

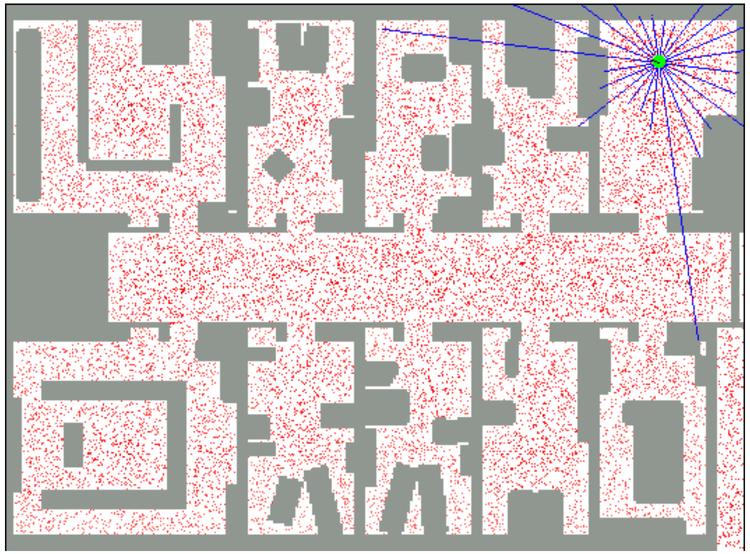
Video of Demo – Moderate Number of Particles

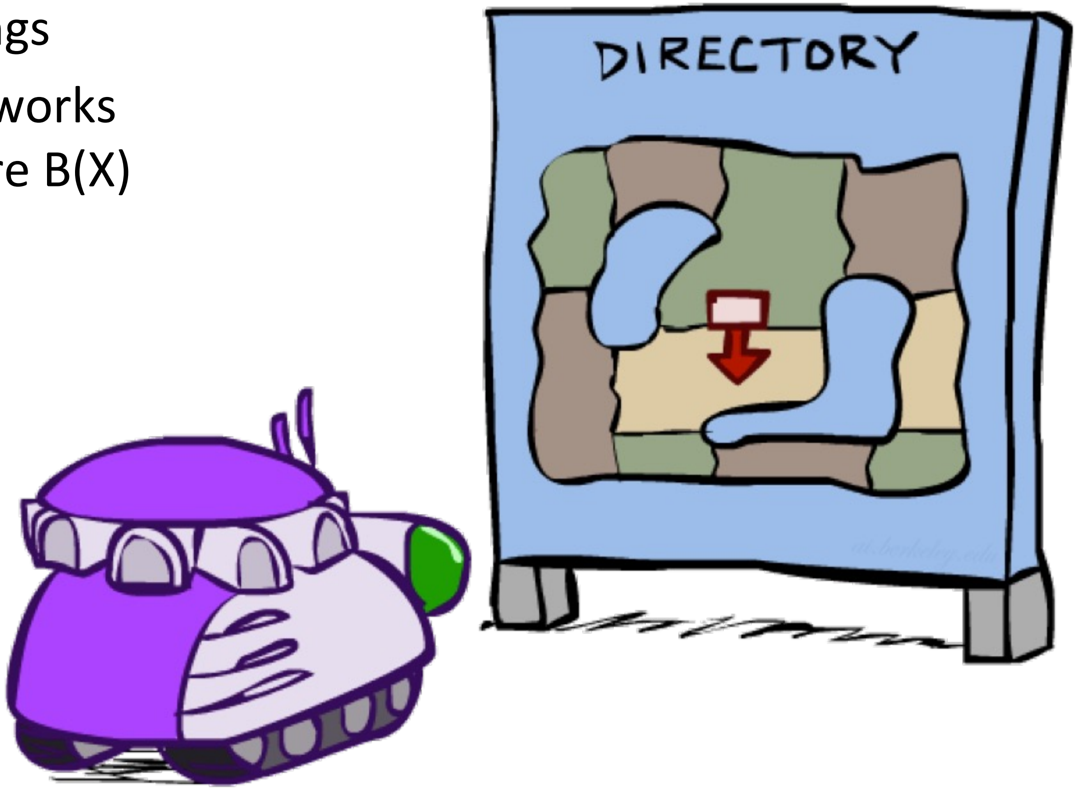
Video of Demo – One Particle

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

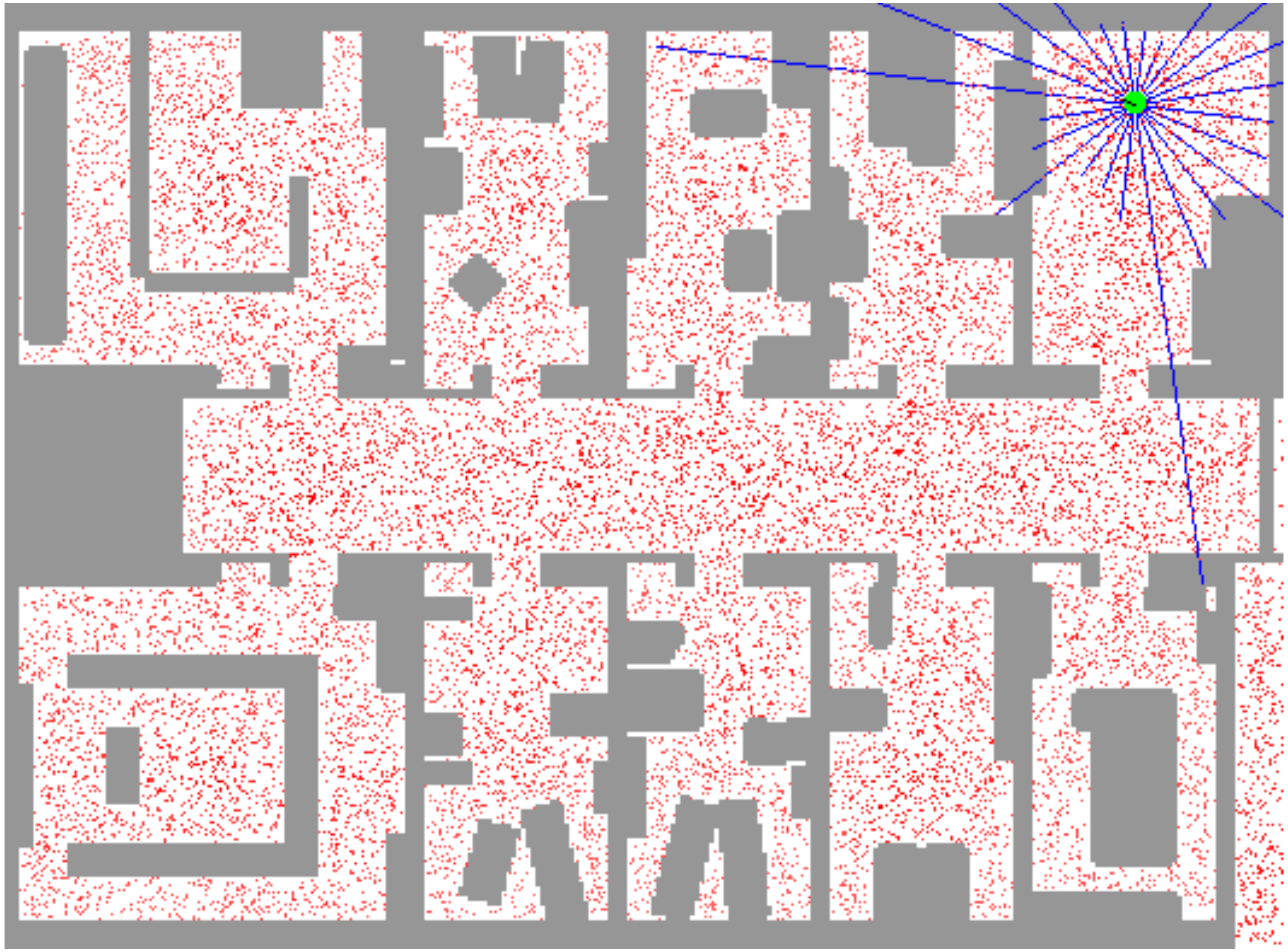




Particle Filter Localization (Sonar)

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

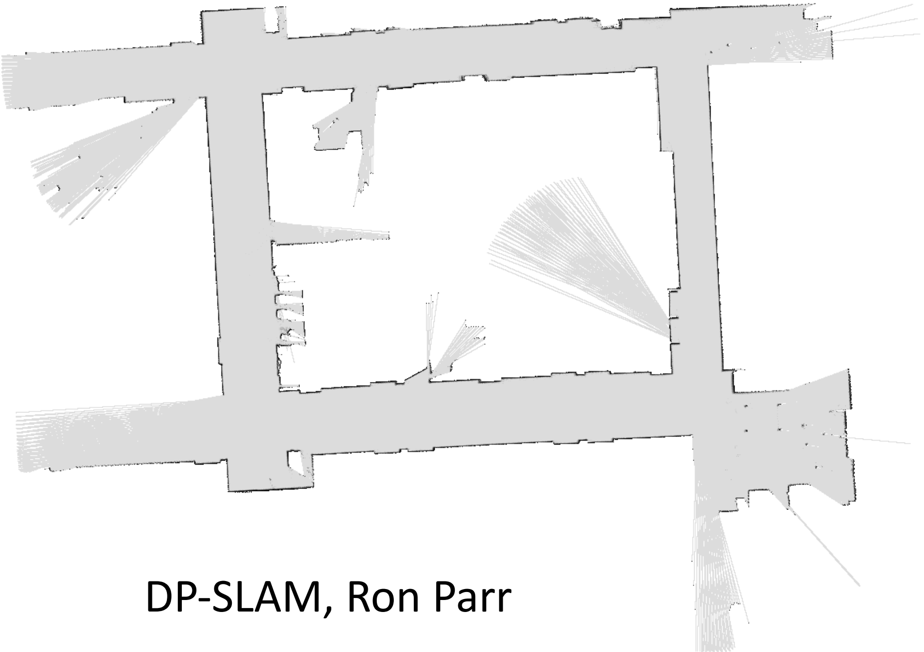
Particle Filter Localization (Laser)



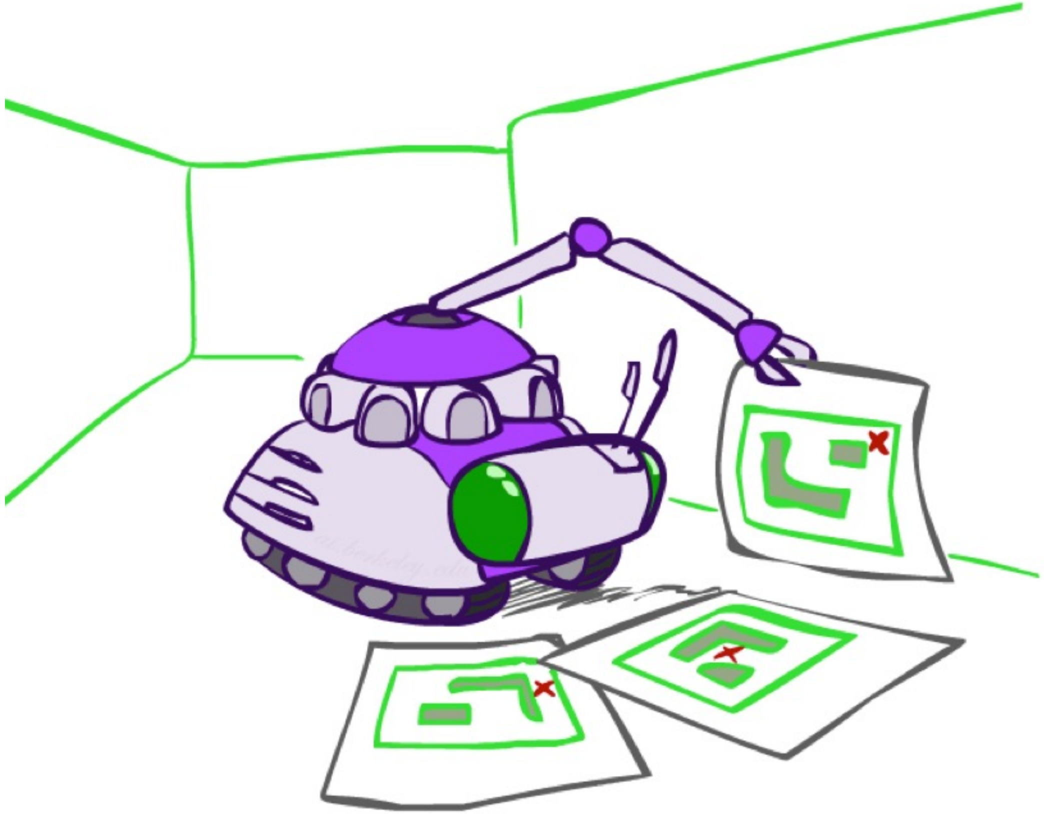
[Video: global-floor.gif]

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

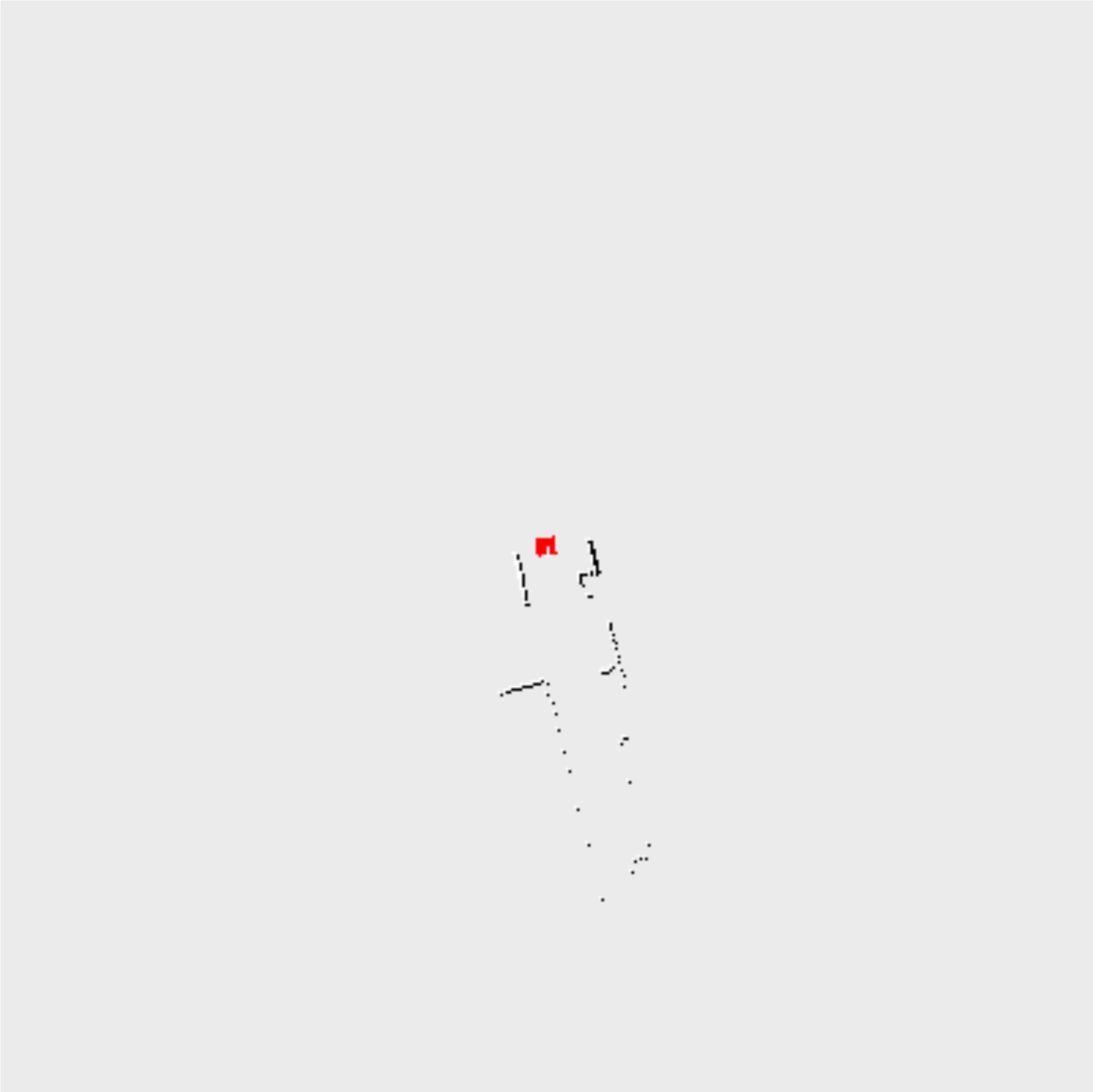


DP-SLAM, Ron Parr



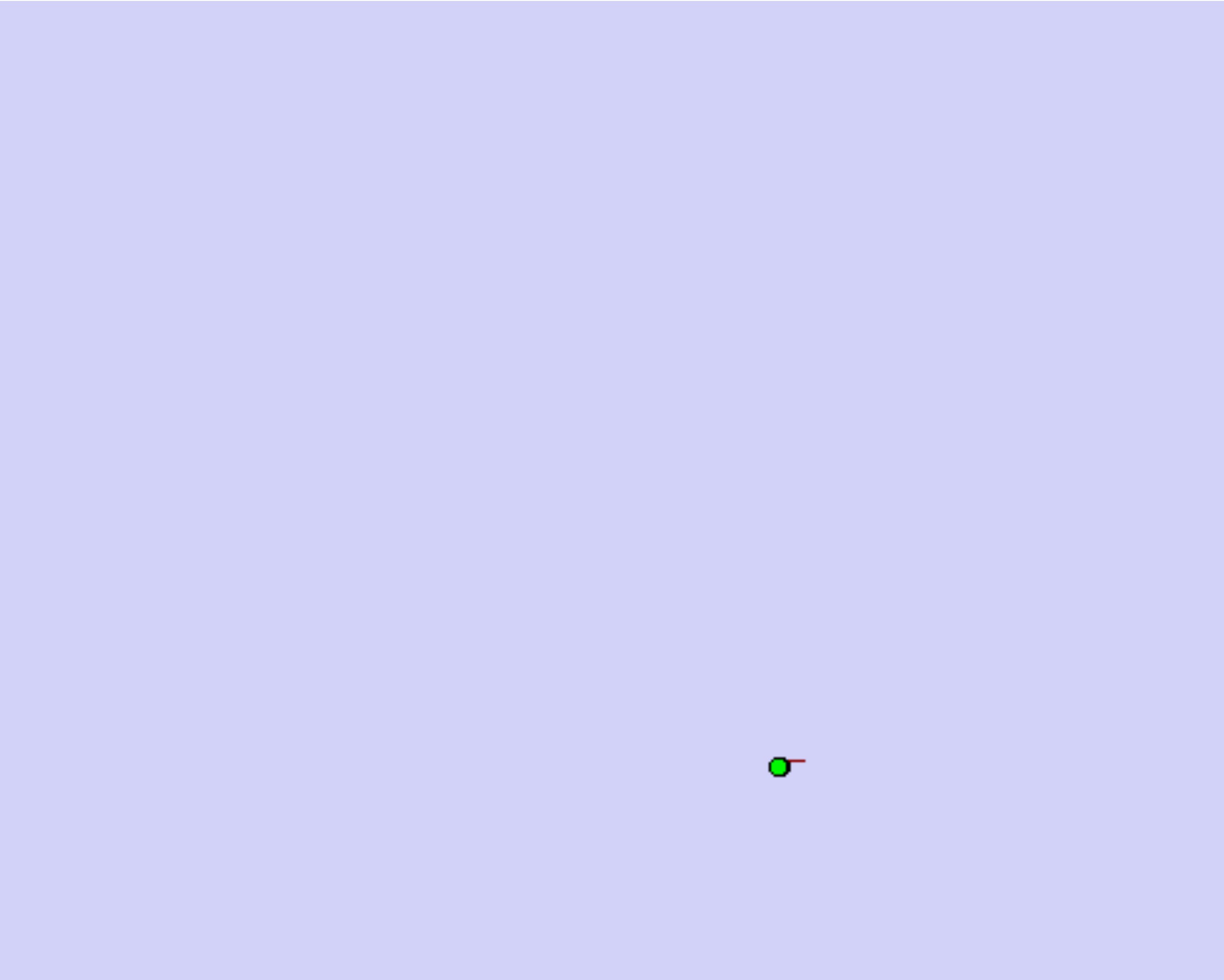
[Demo: PARTICLES-SLAM-mapping1-new.avi]

Particle Filter SLAM – Video 1



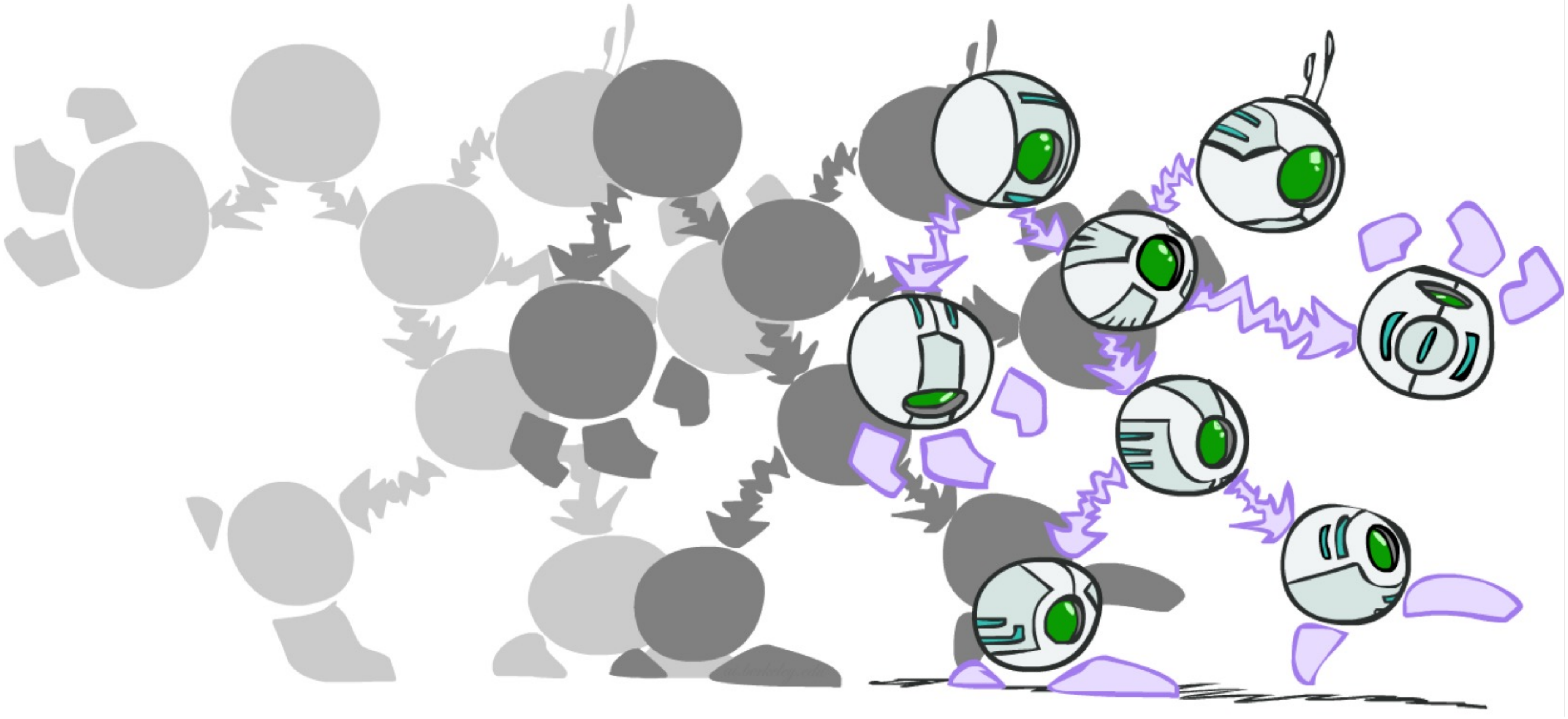
[Demo: PARTICLES-SLAM-mapping1-new.avi]

Particle Filter SLAM – Video 2



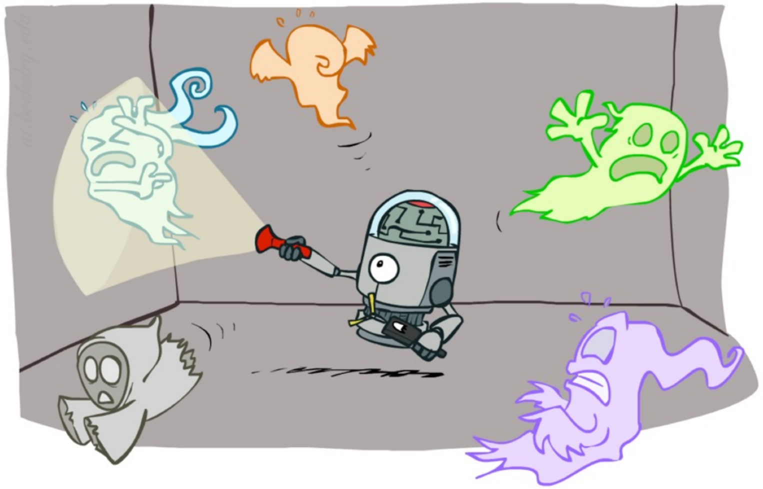
[Demo: PARTICLES-SLAM-fastslam.avi]

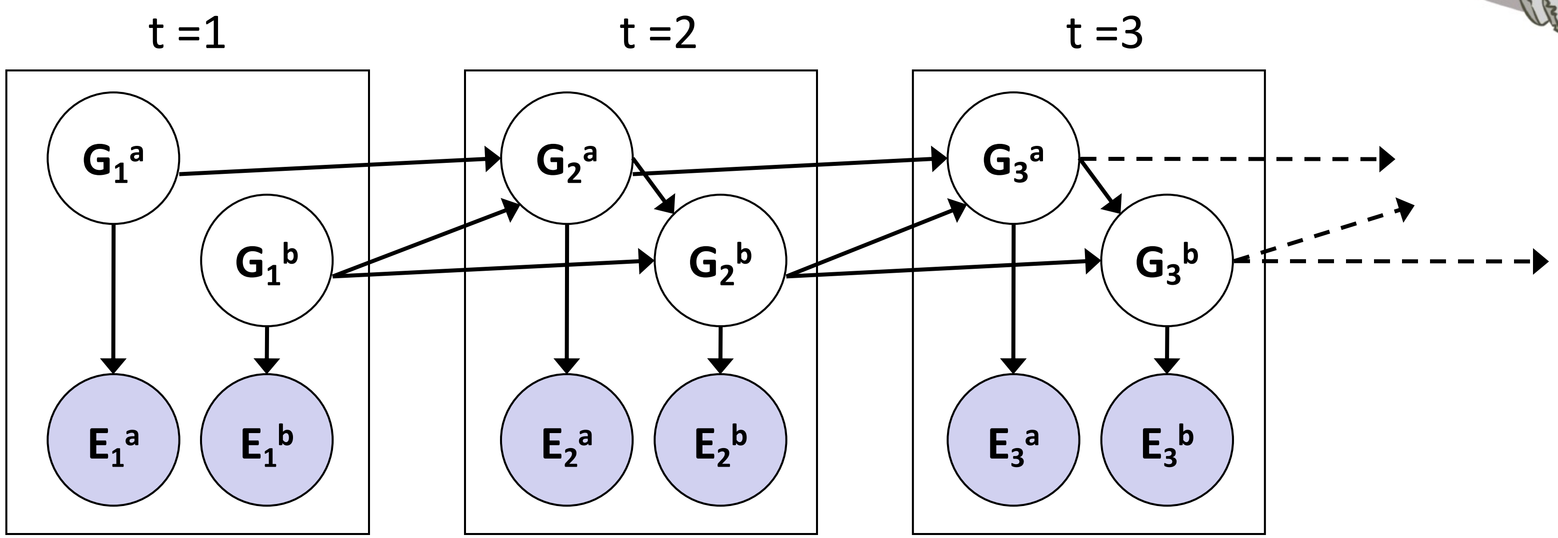
Dynamic Bayes Nets



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)]

Video of Demo Pacman Sonar Ghost DBN Model

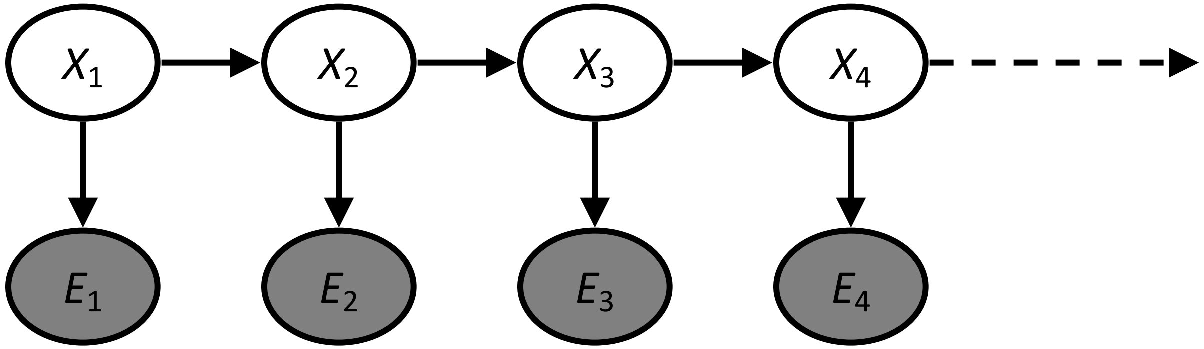
DBN Particle Filters

- 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: $P(\mathbf{E_1}^a \mid \mathbf{G_1}^a) * P(\mathbf{E_1}^b \mid \mathbf{G_1}^b)$
- **Resample:** Select prior samples (tuples of values) in proportion to their likelihood

Most Likely Explanation

HMMs: MLE Queries

- HMMs defined by
  - States X
  - Observations E
  - Initial distribution: $P(X_1)$
  - Transitions: $P(X|X_{-1})$
  - Emissions: $P(E|X)$

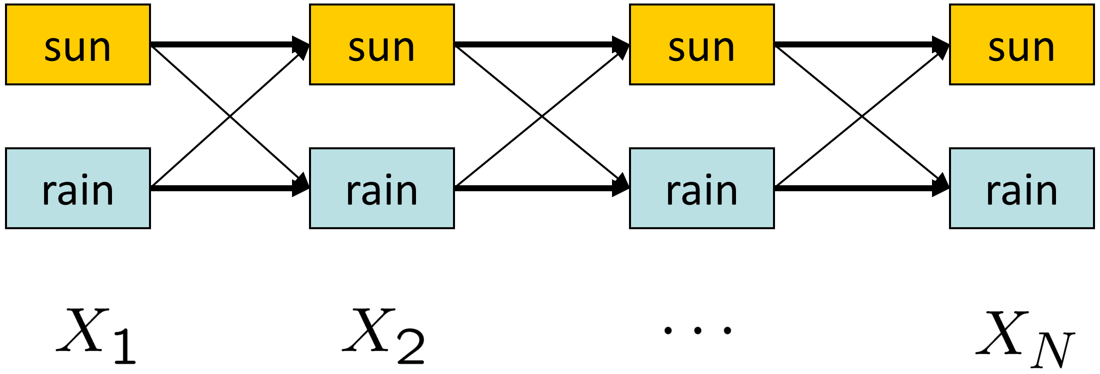


- New query: most likely explanation: $\arg\max_{x_{1:t}} P(x_{1:t}|e_{1:t})$

- New method: the Viterbi algorithm

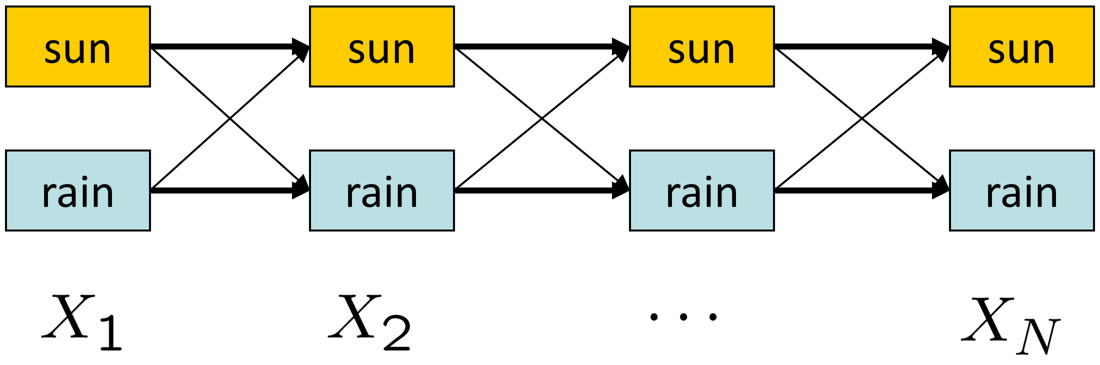
State Trellis

- 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)

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

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

AI in the News

I Know Why You Went to the Clinic: Risks and Realization of HTTPS Traffic Analysis
Brad Miller, Ling Huang, A. D. Joseph, J. D. Tygar (UC Berkeley)

Challenge

- Setting
  - User we want to spy on use HTTPS to browse the internet
- Measurements
  - IP address
  - Sizes of packets coming in
- Goal
  - Infer browsing sequence of that user
- E.g.: medical, financial, legal, …

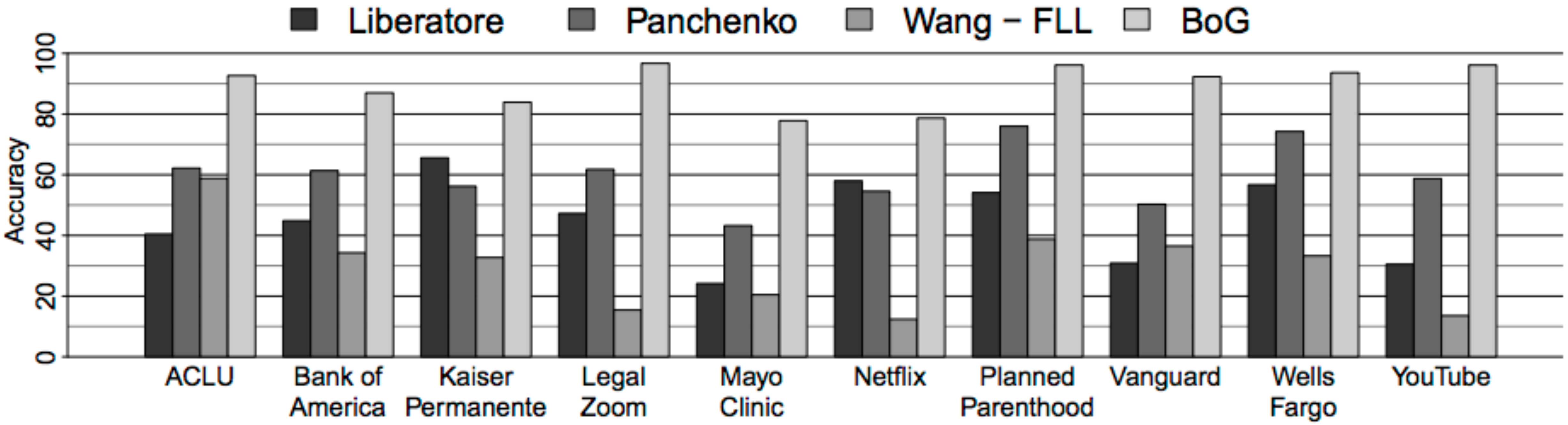
HMM

- Transition model
  - Probability distribution over links on the current page + some probability to navigate to any other page on the site

- Noisy observation model due to traffic variations
  - Caching
  - Dynamically generated content
  - User-specific content, including cookies

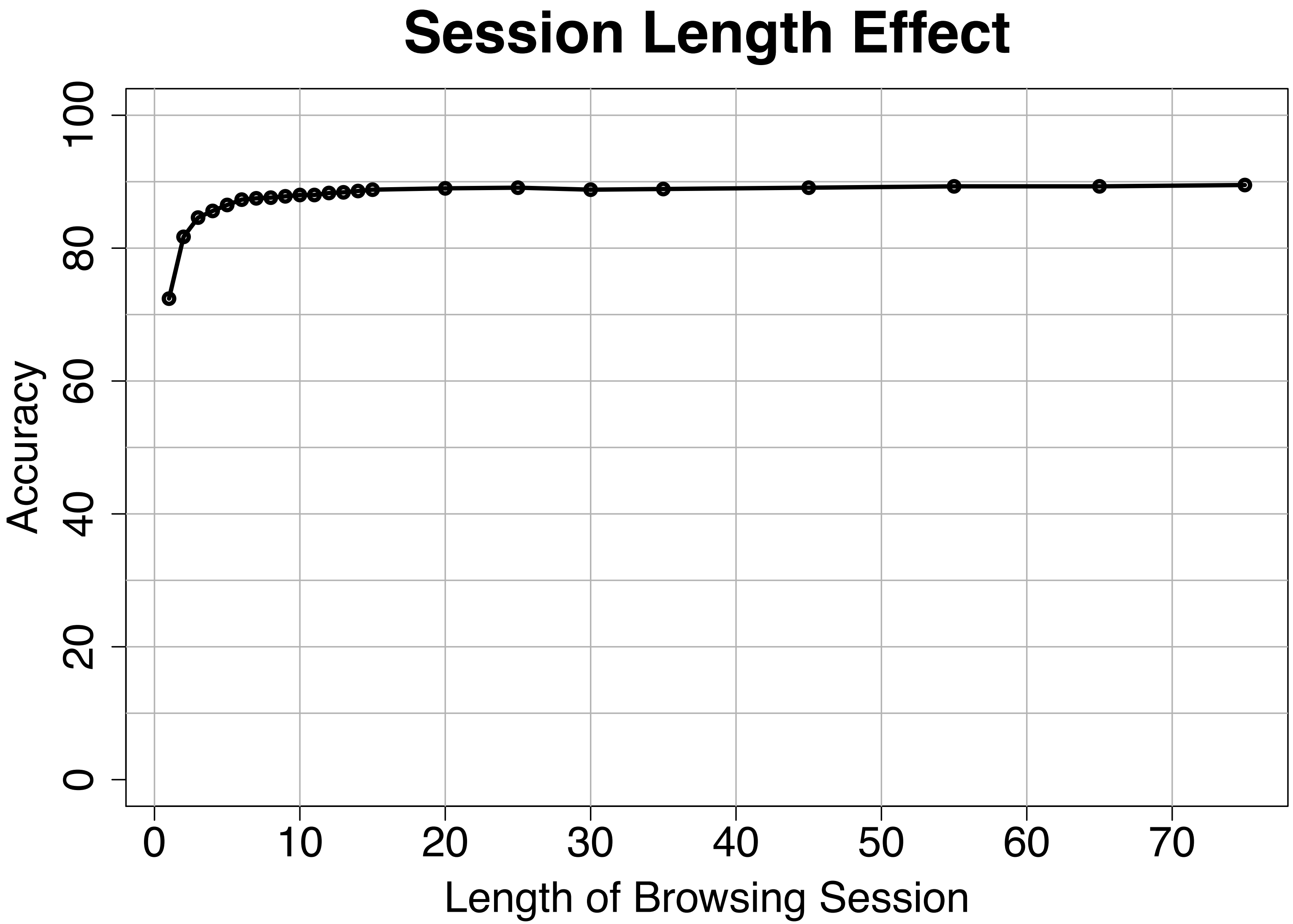
  → Probability distribution P(packet size | page)

Results



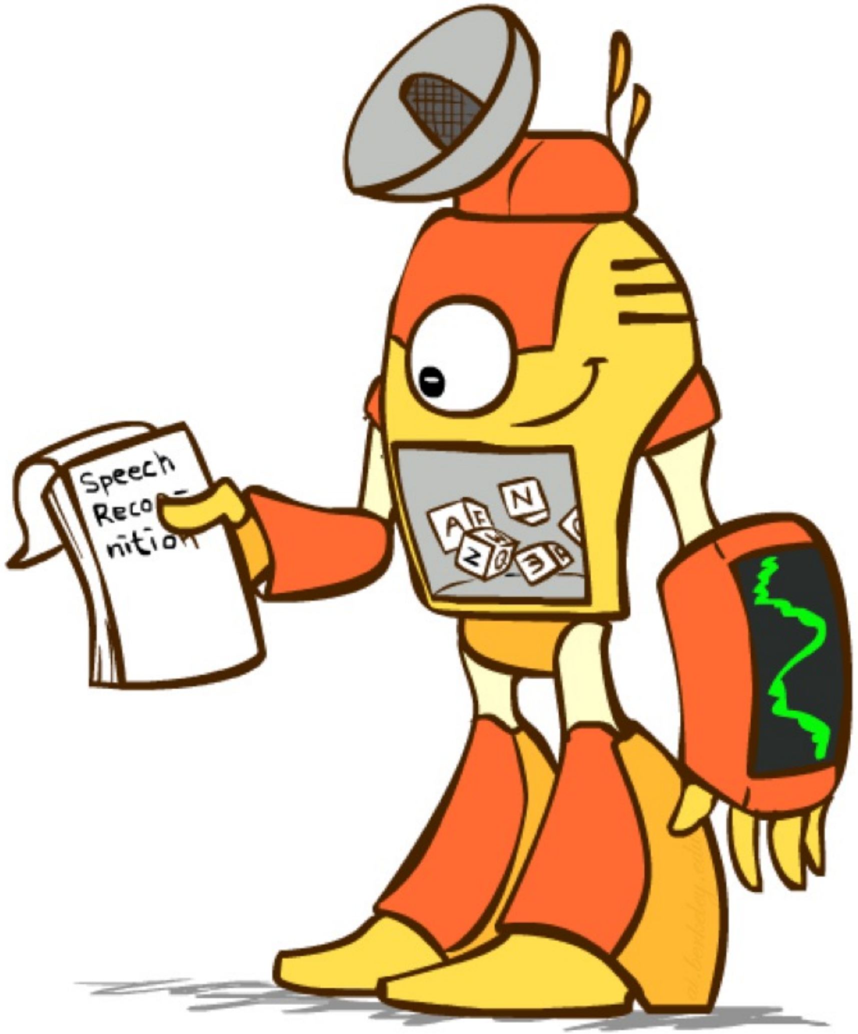
BoG = described approach, others are prior work

Results

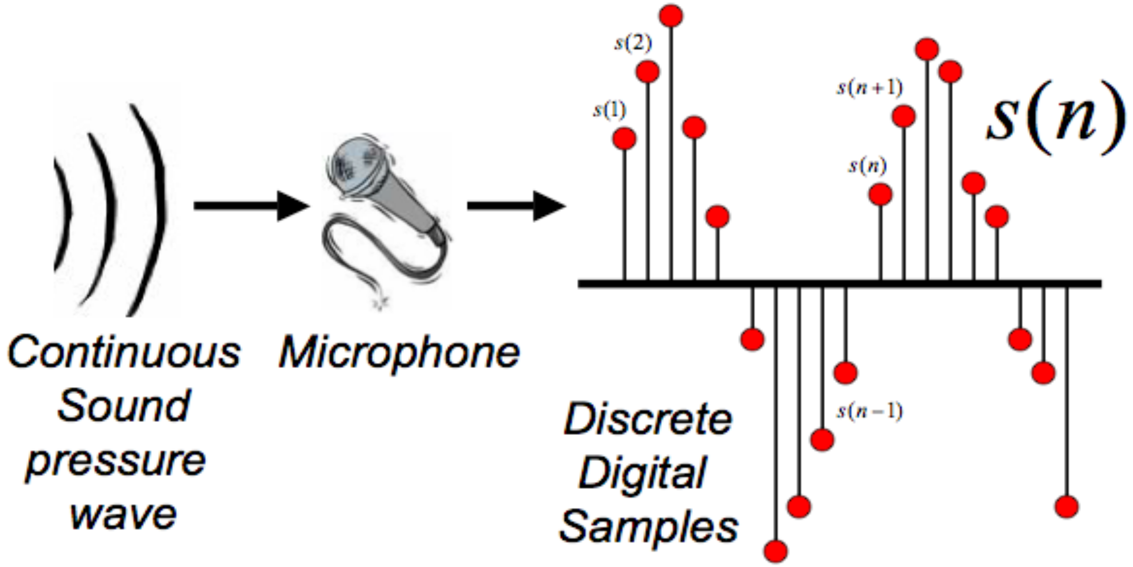


Session Length Effect
Accuracy
0
20
40
60
80
100
0
10
20
30
40
50
60
70
Length of Browsing Session

Speech Recognition



Digitizing Speech

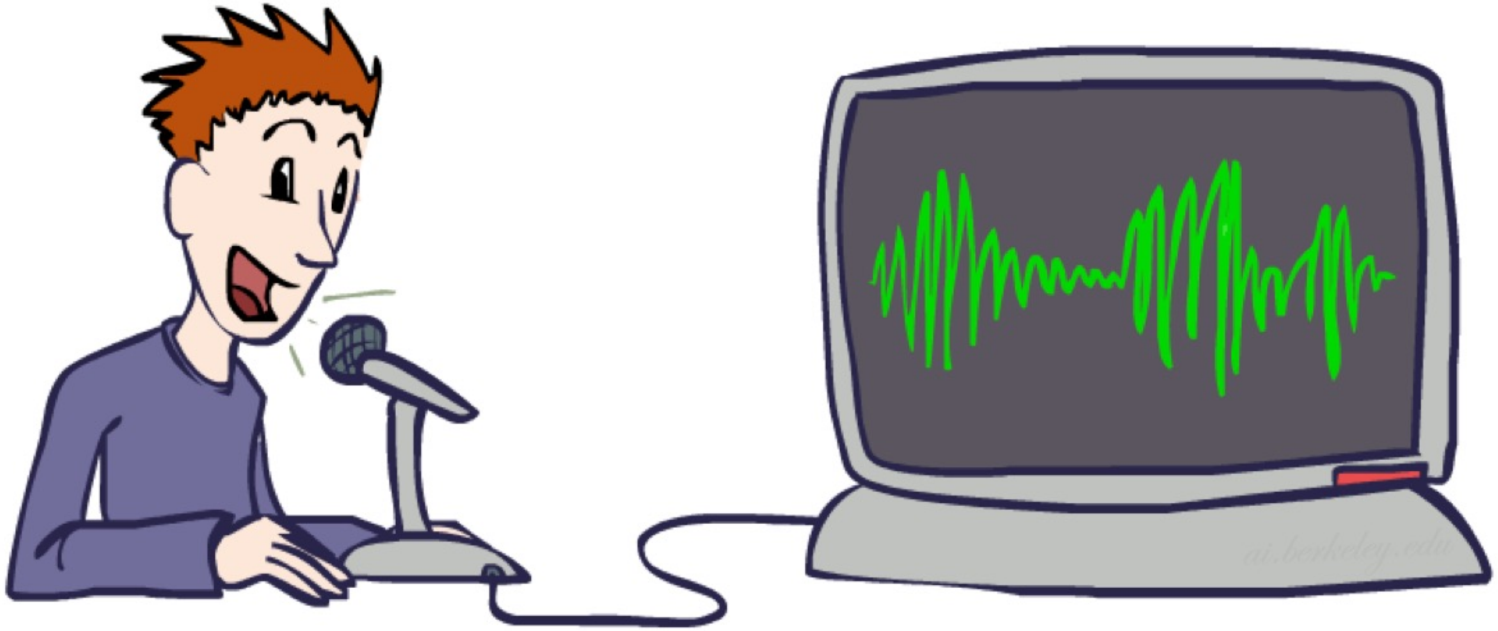


Thanks to Bryan Pellom for this slide!

Speech Recognition in Action

[Video: NLP – ASR tvsample.avi (from Lecture 1)]

Digitizing Speech



Speech in an Hour

- Speech input is an acoustic waveform

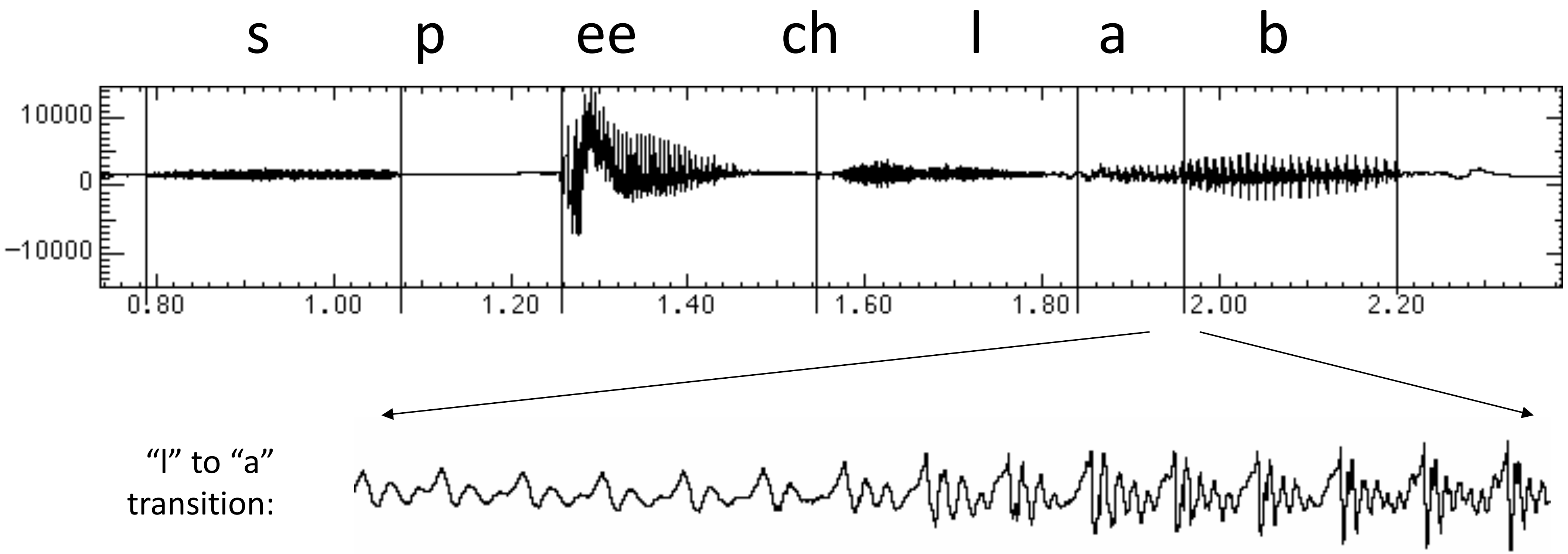
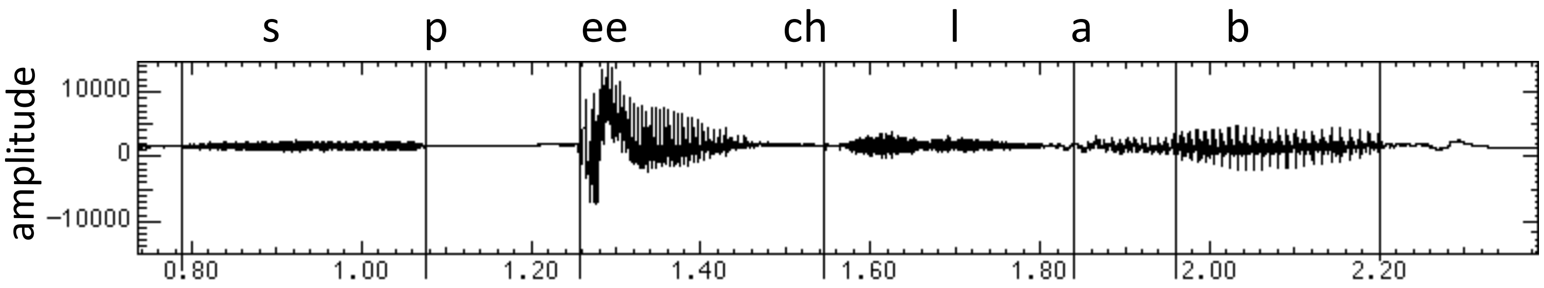
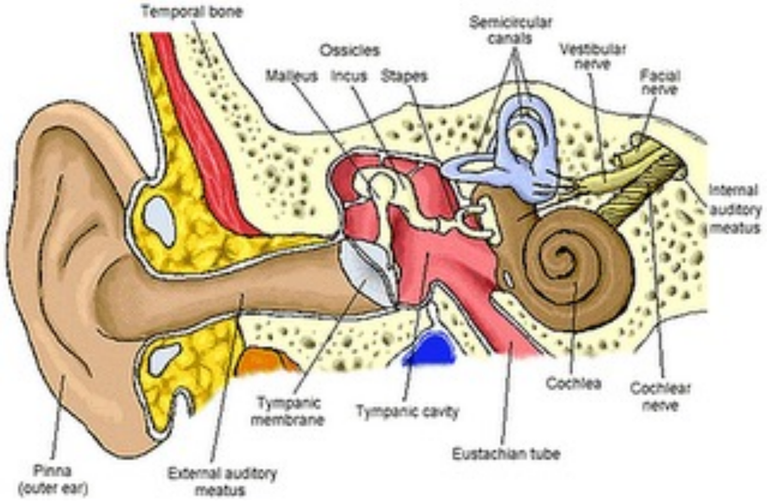


Figure: Simon Arnfield, http://www.psyc.leeds.ac.uk/research/cogn/speech/tutorial/

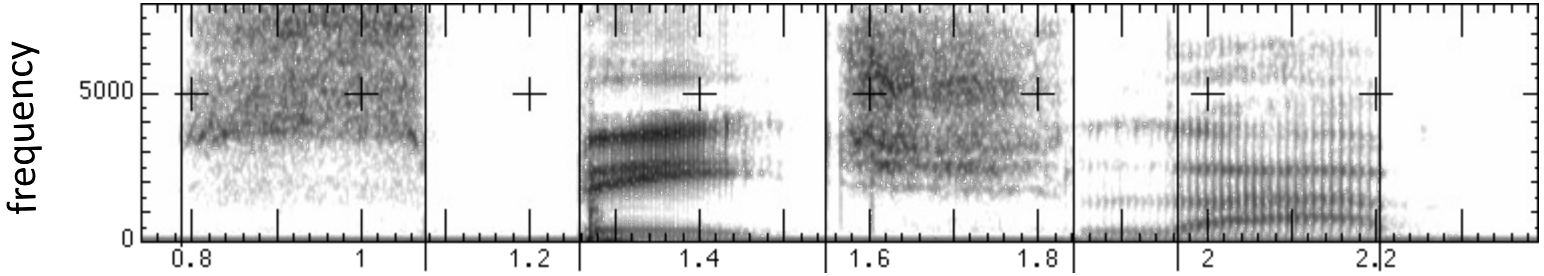
Spectral Analysis

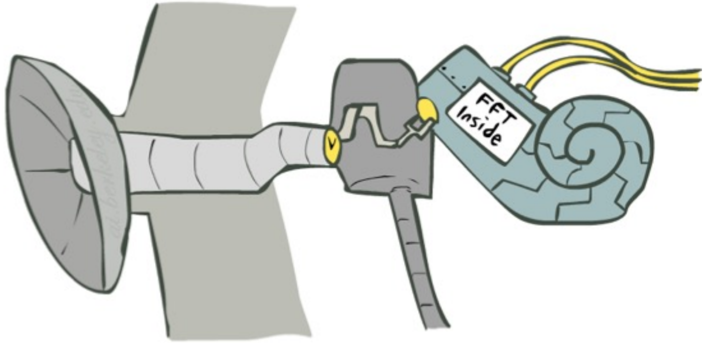
- Frequency gives pitch; amplitude gives volume
  - Sampling at ~8 kHz (phone), ~16 kHz (mic) (kHz=1000 cycles/sec)





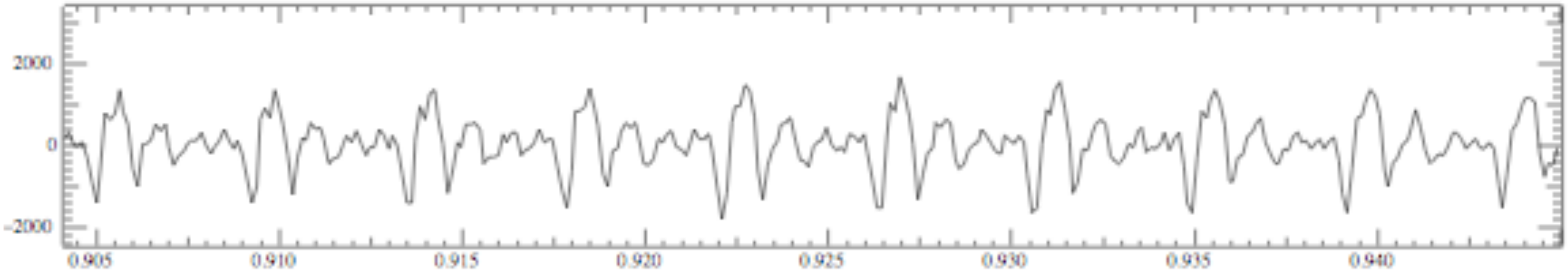
- Fourier transform of wave displayed as a spectrogram
  - Darkness indicates energy at each frequency



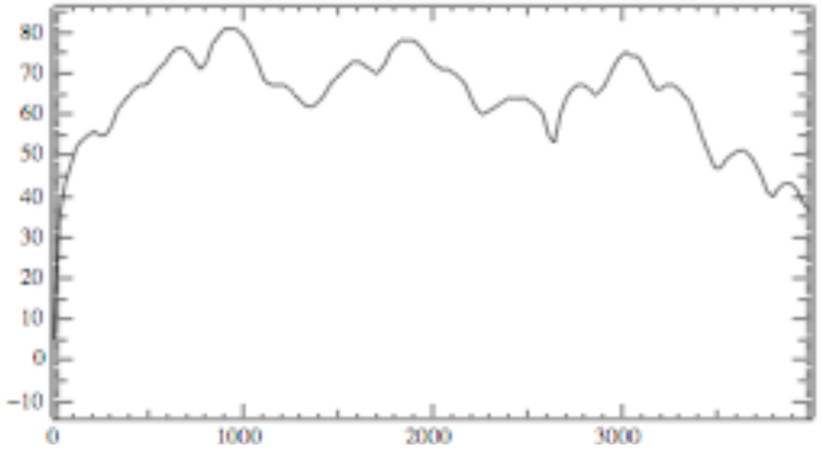


Human ear figure: depion.blogspot.com

Part of [ae] from “lab”

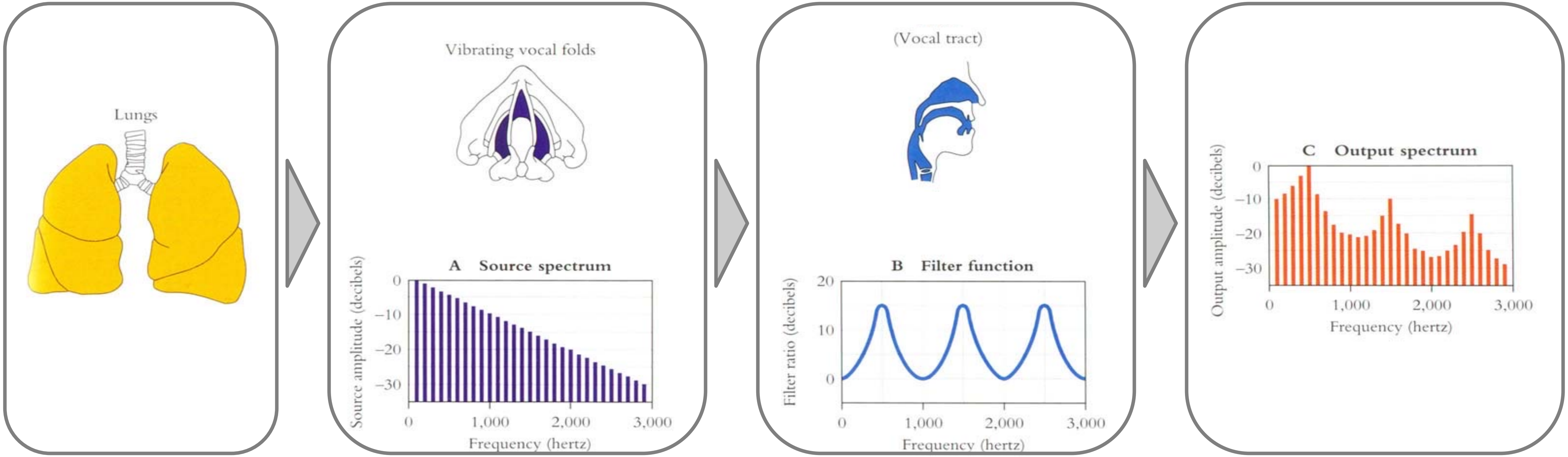


- Complex wave repeating nine times
  - Plus smaller wave that repeats 4x for every large cycle
  - Large wave: freq of 250 Hz (9 times in .036 seconds)
  - Small wave roughly 4 times this, or roughly 1000 Hz



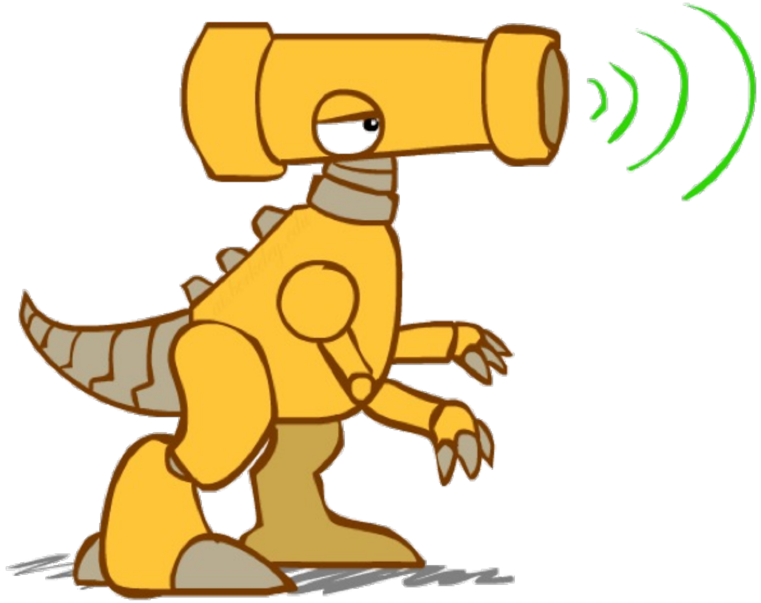
Why These Peaks?

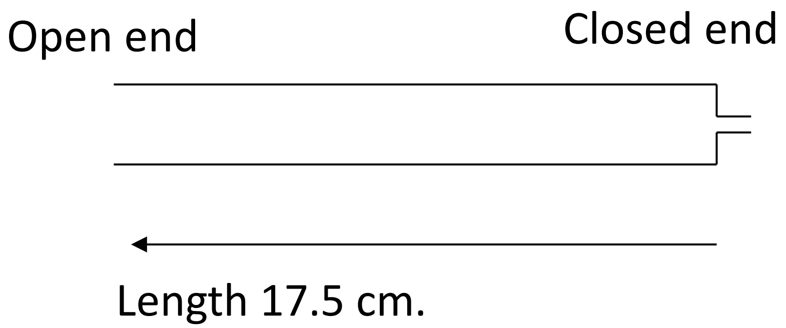
- Articulator process:
  - Vocal cord vibrations create harmonics
  - The mouth is an amplifier
  - Depending on shape of mouth, some harmonics are amplified more than others

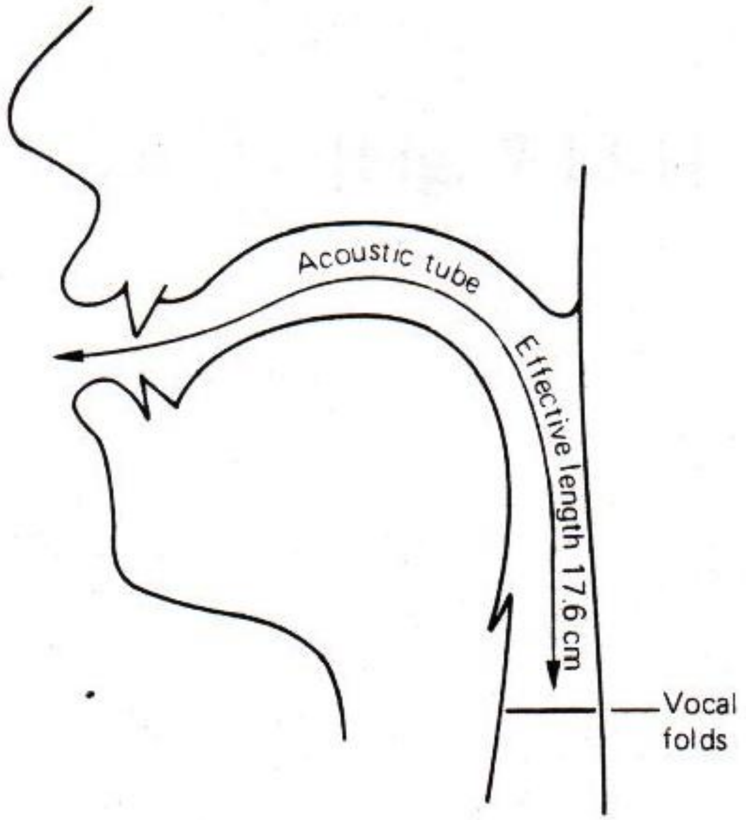


Resonances of the Vocal Tract

- The human vocal tract as an open tube







- Air in a tube of a given length will tend to vibrate at resonance frequency of tube
- Constraint: Pressure differential should be maximal at (closed) glottal end and minimal at (open) lip end

Figure: W. Barry Speech Science slides

Spectrum Shapes

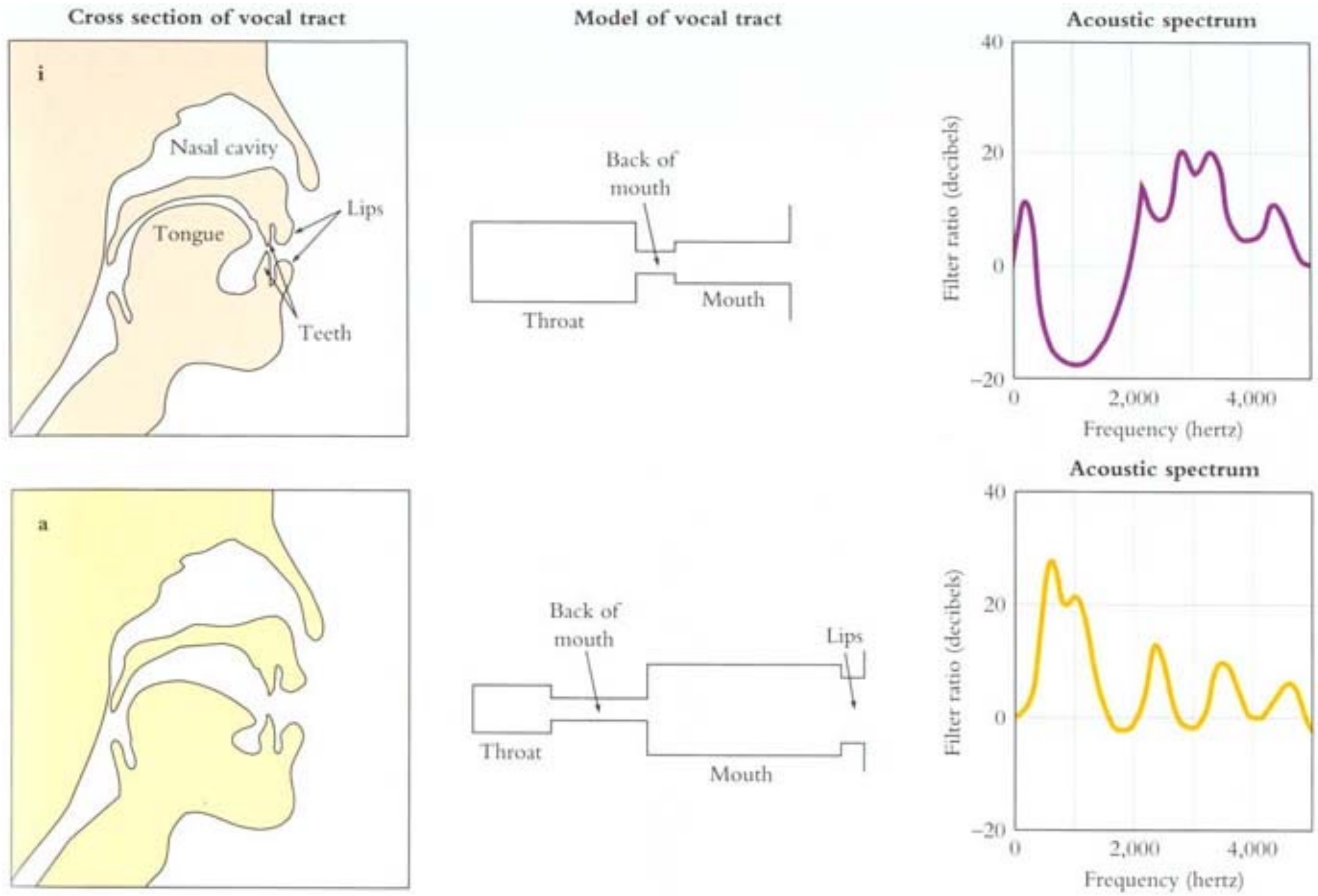
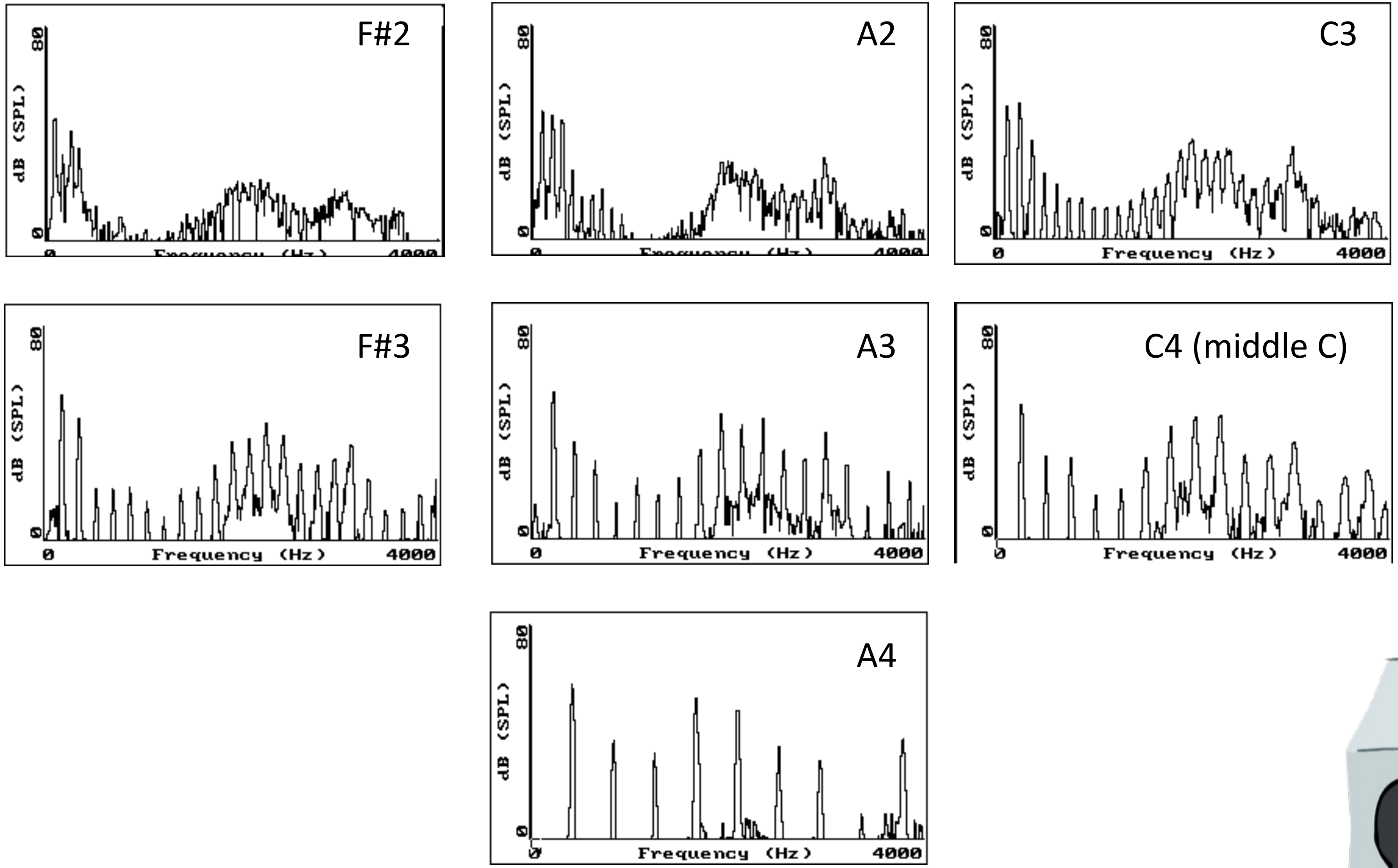


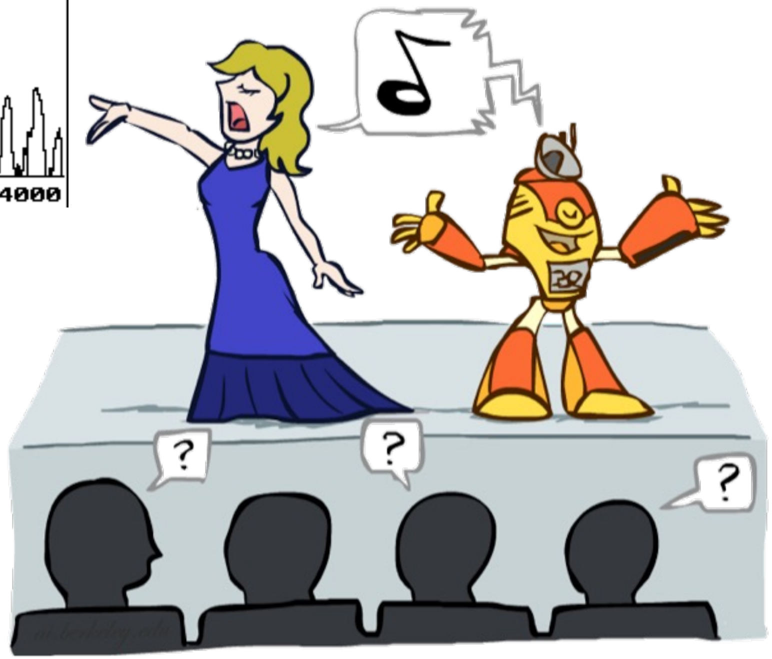
Figure: Mark Liberman

[Demo: speech synthesis]

Video of Demo Speech Synthesis

Vowel [i] sung at successively higher pitches

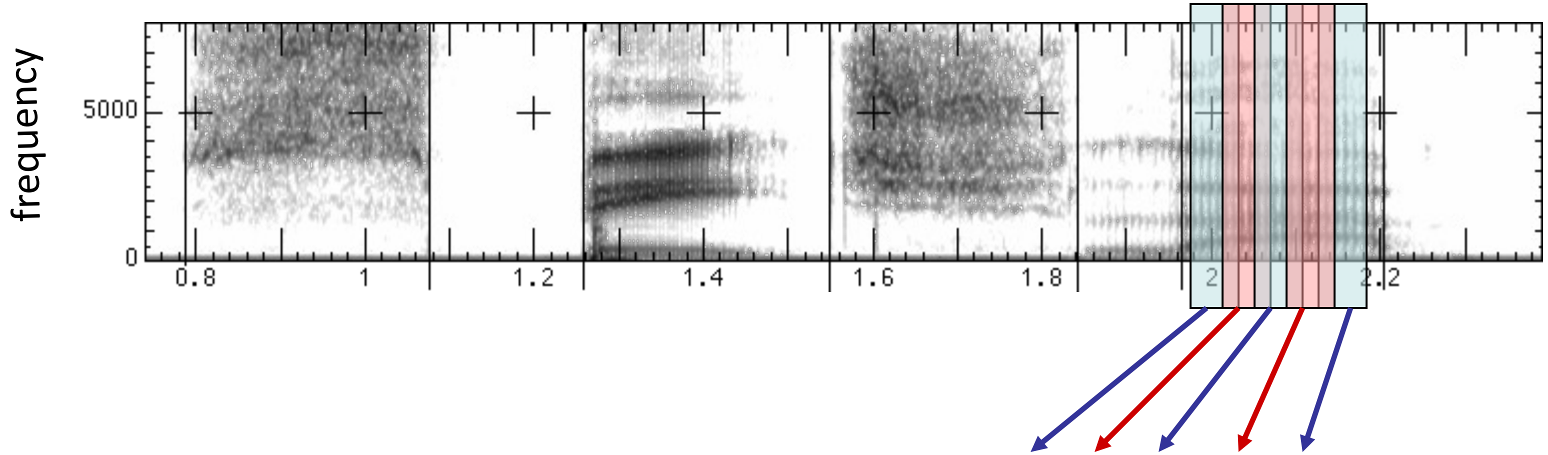




Graphs: Ratree Wayland

Acoustic Feature Sequence

- Time slices are translated into acoustic feature vectors (~39 real numbers per slice)



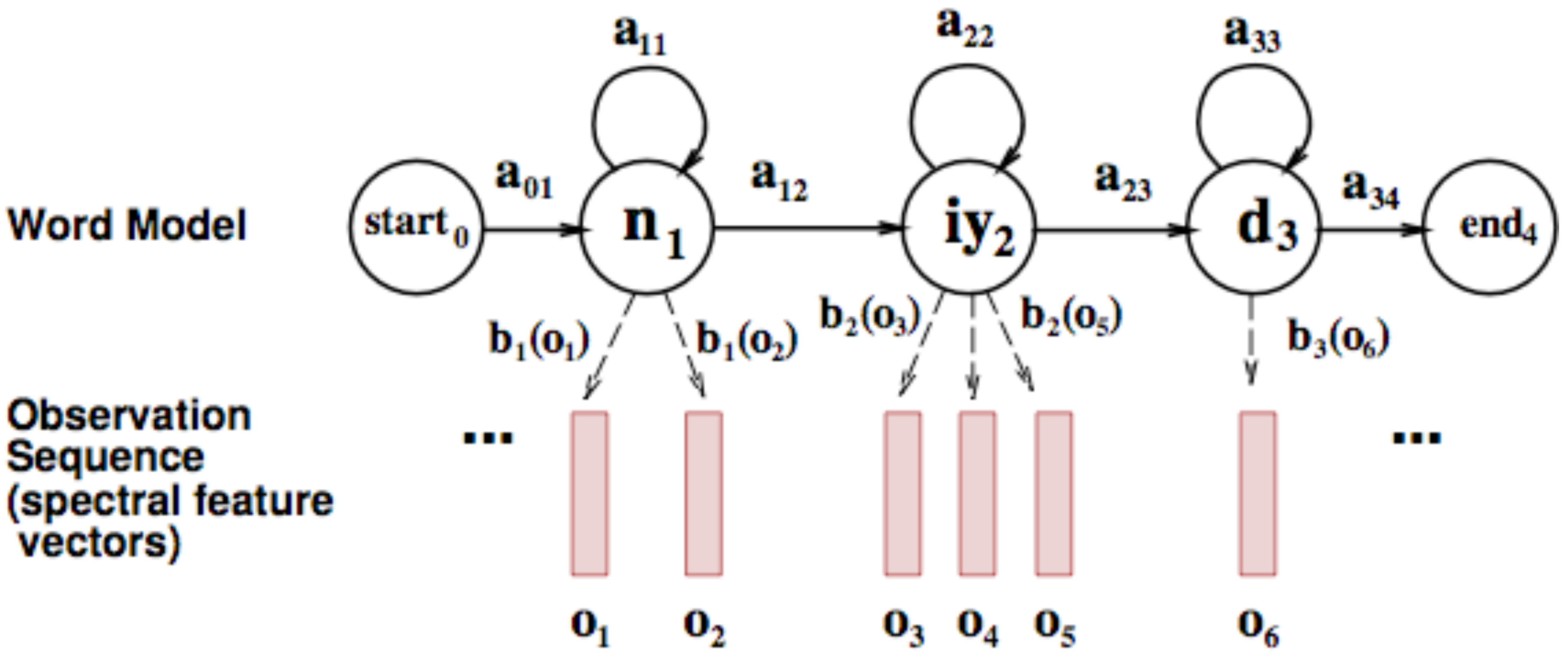
……………………………………………… $e_{12}e_{13}e_{14}e_{15}e_{16}$ ……….

- These are the observations E, now we need the hidden states X

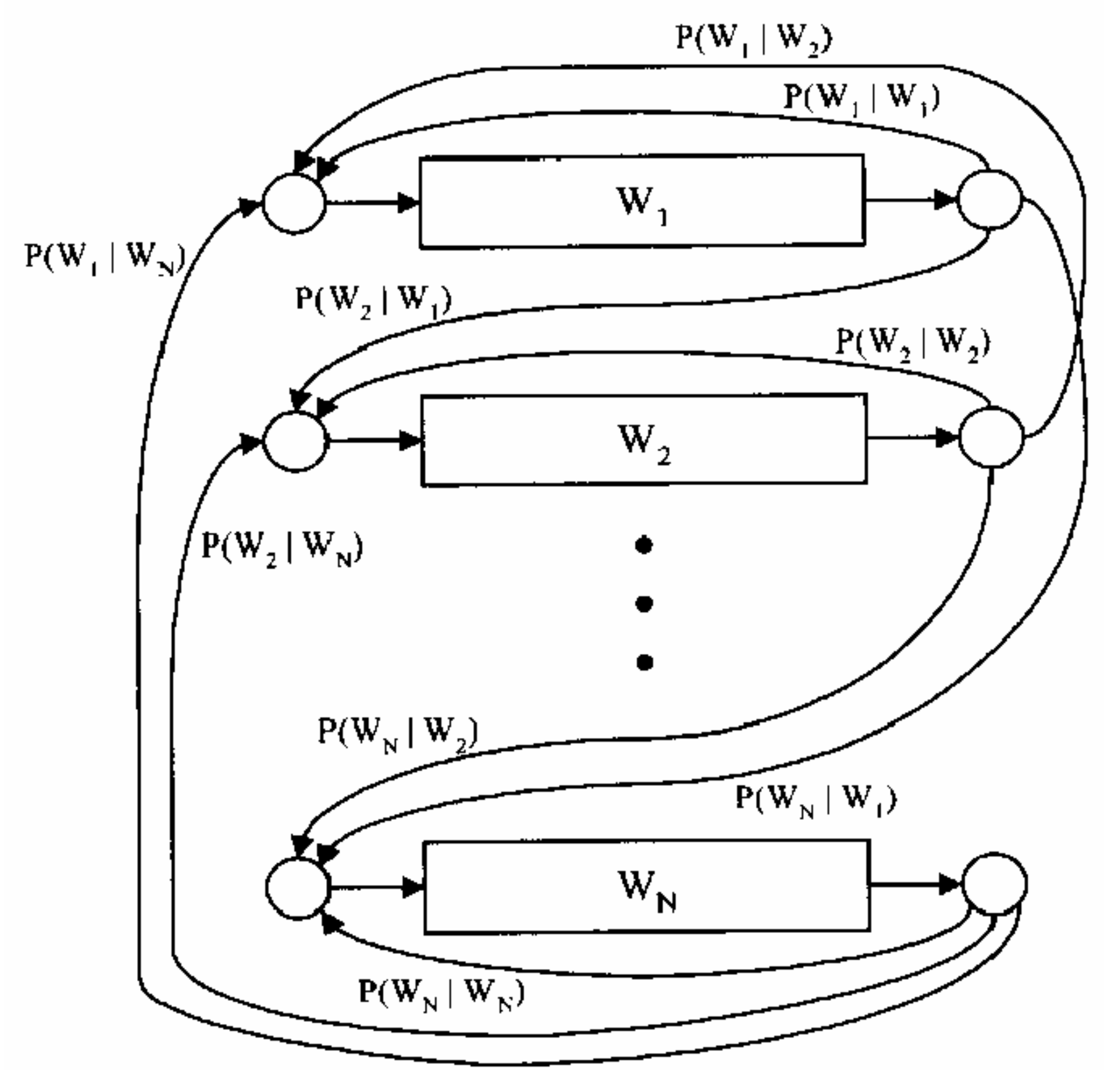
Speech State Space

- HMM Specification
  - P(E|X) encodes which acoustic vectors are appropriate for each phoneme (each kind of sound)
  - P(X|X’) encodes how sounds can be strung together
- State Space
  - We will have one state for each sound in each word
  - Mostly, states advance sound by sound
  - Build a little state graph for each word and chain them together to form the state space X

States in a Word



Transitions with a Bigram Model



Training Counts

198015222 the first
194623024 the same
168504105 the following
158562063 the world
…
14112454 the door

23135851162 the *

$$\hat{P}(\text{door}|\text{the}) = \frac{14112454}{23135851162}$$

$$= 0.0006$$

Figure: Huang et al, p. 618

Decoding

- Finding the words given the acoustics is an HMM inference problem
- Which state sequence $x_{1:T}$ is most likely given the evidence $e_{1:T}$?

$$x^*_{1:T} = \arg\max_{x_{1:T}} P(x_{1:T}|e_{1:T}) = \arg\max_{x_{1:T}} P(x_{1:T}, e_{1:T})$$

- From the sequence x, we can simply read off the words

Next Time: Bayes' Nets!