Hidden Markov Models

Hongxin Zhang
zhx@cad.zju.edu.cn

State Key Lab of CAD&CG, ZJU
2010-03-25

Outline

- Background
- Markov Chains
- Hidden Markov Models

Example: Video Textures

- Problem statement

video clip   video texture

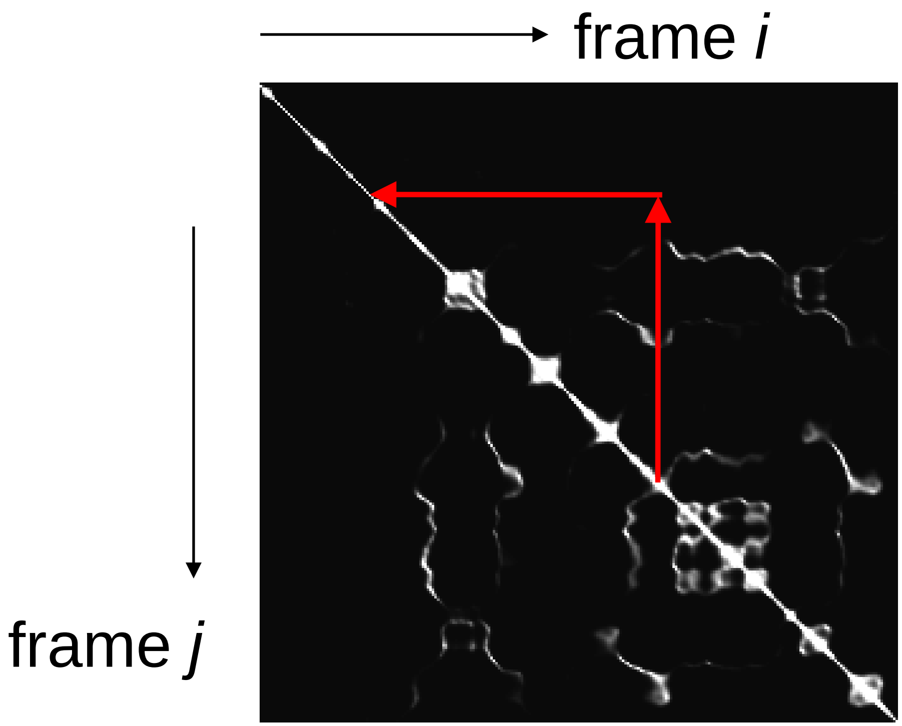
SIGGRAPH 2000. Schoedl et. al.

The approach

How do we find good transitions?

Finding good transitions

Compute L_2 distance $D_{i,j}$ between all frames



Similar frames make good transitions

Demo: Fish Tank

Mathematic model of Video Texture

A sequence of random variables

{ADEABEDADBCAD}

A sequence of random variables

{BDACBDCACDBCADCBADCA}

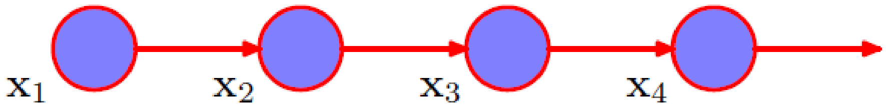
Markov Model

The future is independent of the past and given by the present.

Markov Property

- Formal definition
  - Let $X=\{X_n\}_{n=0\ldots N}$ be a sequence of random variables taking values $s_k \in N$ if and only if $P(X_m=s_m|X_0=s_0,\ldots,X_{m-1}=s_{m-1}) = P(X_m=s_m| X_{m-1}=s_{m-1})$

then the X fulfills Markov property



- Informal definition
  - The future is independent of the past given the present.

History of MC

- Markov chain theory developed around 1900.
- Hidden Markov Models developed in late 1960's.
- Used extensively in speech recognition in 1960-70.
- Introduced to computer science in 1989.

Andrei Andreyevich Markov

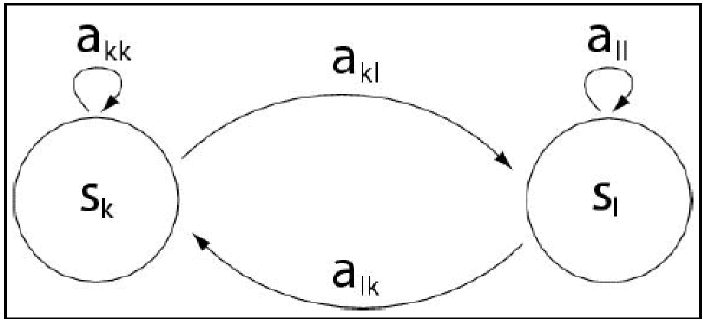
Applications

- ➢ Bioinformatics.
- ➢ Signal Processing
- ➢ Data analysis and Pattern recognition

Markov Chain

- A Markov chain is specified by
  - A **state space** $S = \{ s_1, s_2 ..., s_n \}$
  - An **initial distribution** a_0
  - A **transition matrix** A

  Where $A(n)_{ij} = a_{ij} = P(q_t = s_j | q_{t-1} = s_i)$



- Graphical Representation as a directed graph where
  - Vertices represent states
  - Edges represent transitions with positive probability

Probability Axioms

- Marginal Probability – sum the joint probability

$$P(x = a_i) \equiv \sum_{y \in A_Y} P(x = a_i, y)$$

- Conditional Probability

$$P(x = a_i \mid y = b_j) \equiv \frac{P(x = a_i, y = b_j)}{P(y = b_j)} \text{ if } P(y = b_j) \neq 0.$$

Calculating with Markov chains

- Probability of an observation sequence:
  - Let $X=\{x_t\}^L_{t=0}$ be an observation sequence from the Markov chain $\{S, a_0, A\}$

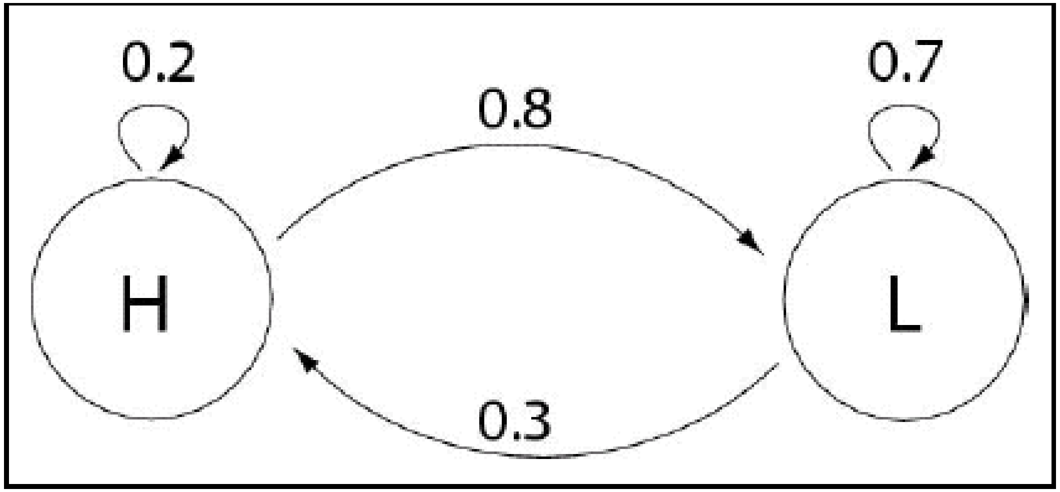
$$
\begin{aligned}
P(x) &= P(x_L, \ldots, x_1, x_0) \\
&= P(x_L \mid x_{L-1}, \ldots, x_0) P(x_{L-1} \mid x_{L-2}, \ldots, x_0) \cdots P(x_0) \\
&= P(x_L \mid x_{L-1}) P(x_{L-1} \mid x_{L-2}) \cdots P(x_0) \\
&= \mathbf{b}_{x_0} \prod_{i=1}^{L} a_{x_{i-1} x_i}
\end{aligned}
$$

Example

Assume we are modeling a time series of high and low pressures during the Danish autumn.

Let $S = \{H, L\}$, $\mathbf{b} = \pi = \left[\frac{3}{11}, \frac{8}{11}\right]$, and $A = \begin{bmatrix} 0.2 & 0.8 \\ 0.3 & 0.7 \end{bmatrix}$.

Graphical representation of A

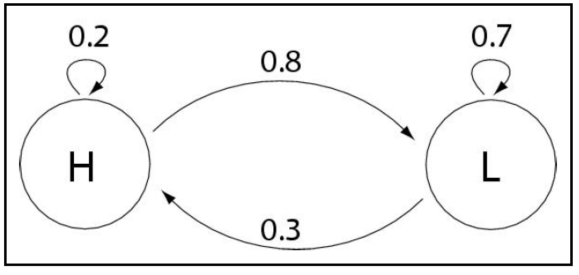


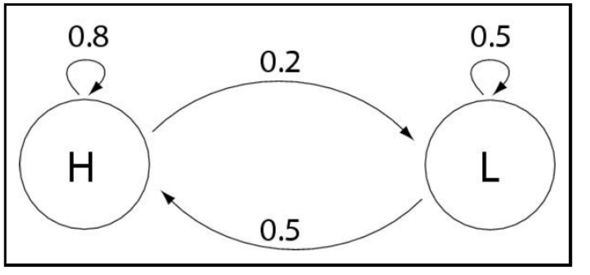
Example

Comparing likelihoods

We want to know the likelihood of one week of high pressure in Denmark (DK) versus California (Cal).

x=HHHHHHH





$$P(x \mid DK) = \mathbf{b}_H a_{HH} a_{HH} a_{HH} a_{HH} a_{HH} a_{HH} a_{HH} = \frac{3}{11}\left(\frac{1}{5}\right)^6 \approx 0.0017\%$$

$$P(x \mid Cal) = \mathbf{b}_H a_{HH} a_{HH} a_{HH} a_{HH} a_{HH} a_{HH} a_{HH} = \frac{5}{7}\left(\frac{4}{5}\right)^6 \approx 0.19\%$$

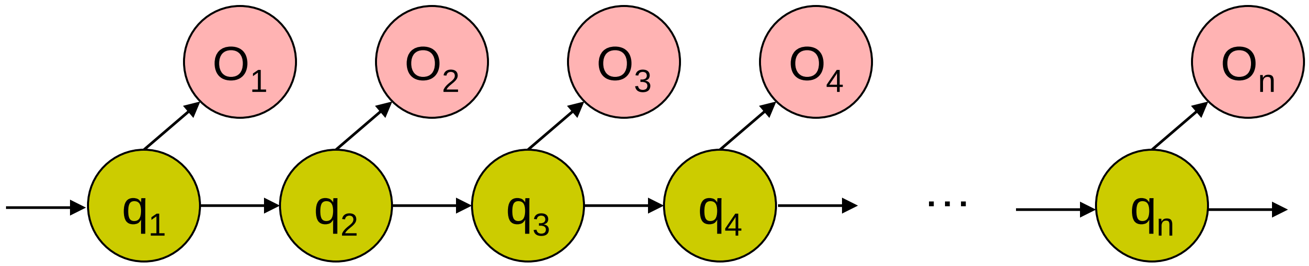
Motivation of Hidden Markov Models

- **Hidden states**
  - The state of the entity we want to model is often not observable:
    - The state is then said to be hidden.
- **Observables**
  - Sometimes we can instead observe the state of entities influenced by the hidden state.
- **A system can be modeled by an HMM if:**
  - The sequence of hidden states is Markov
  - The sequence of observations are independent (or Markov) given the hidden

Hidden Markov Model

- Definition $M=\{S,V,A,B,\pi\}$
  - **Set of states** $S = \{ s_1, s_2, \ldots, s_N\}$
  - **Observation symbols** $V = \{ v_1, v_2, \ldots, v_M\}$
  - **Transition probabilities**
    - A between any two states $a_{ij} = P(q_t=s_j|q_{t-1}=s_i)$
  - **Emission probabilities**
    - B within each state $b_j(O_t) = P(O_t=v_j| q_t = s_j)$
  - **Start probabilities** $\pi = \{a_0\}$

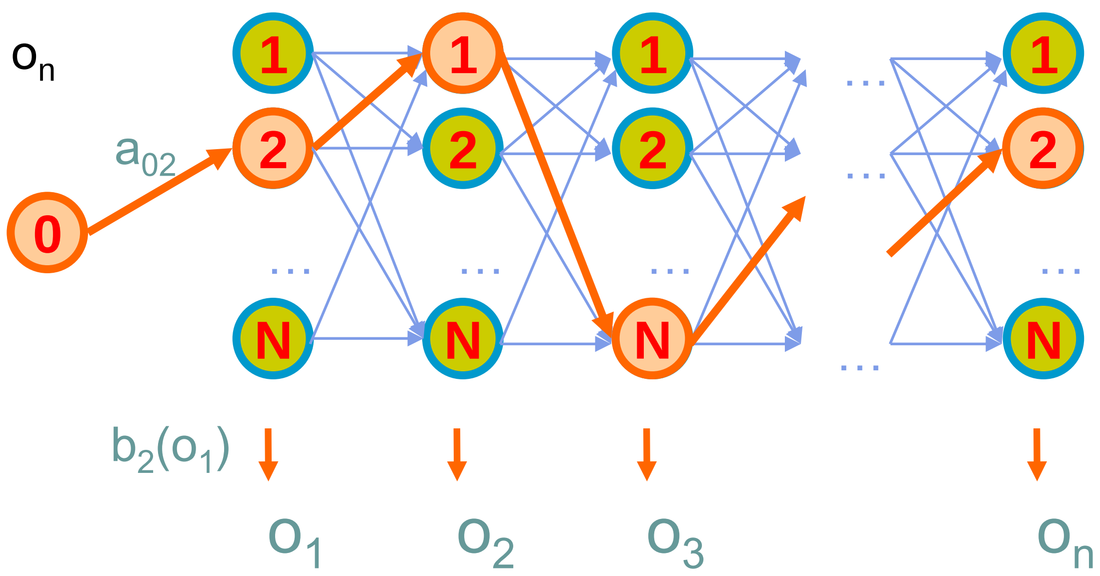
Use $\lambda = (A, B, \pi)$ to indicate the parameter set of the model.



Generating a sequence by the model

Given a HMM, we can generate a sequence of length n as follows:

1. Start at state q_1 according to prob a_{0t1}
2. Emit letter o_1 according to prob $e_{t1}(o_1)$
3. Go to state q_2 according to prob a_{t1t2}
4. … until emitting o_n



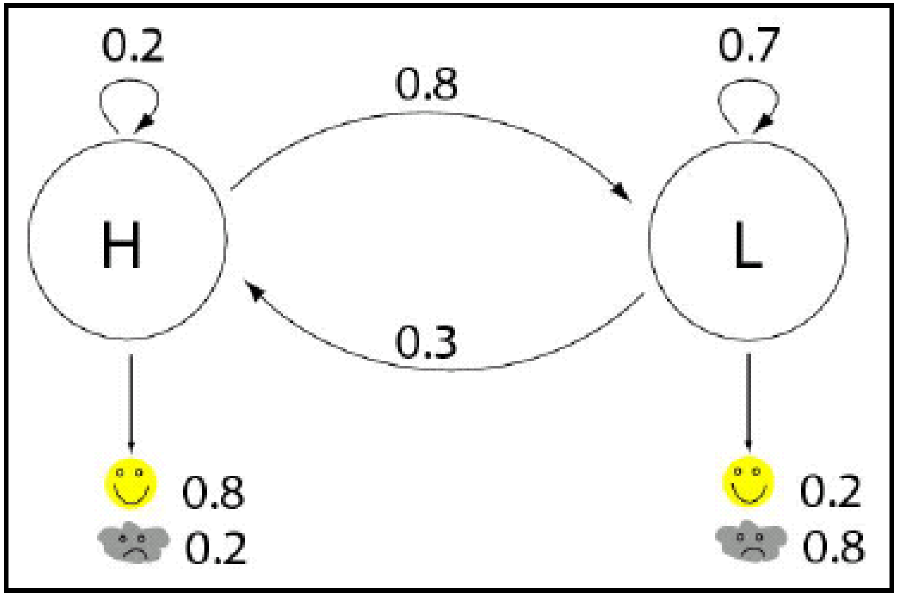
Example

Model of high and low pressures

Assume we can not measure high and low pressures.

The state of the weather is influenced by the air pressure.

We make an HMM with hidden states representing high and low pressure and observations representing the weather:



Calculating with Hidden Markov Model

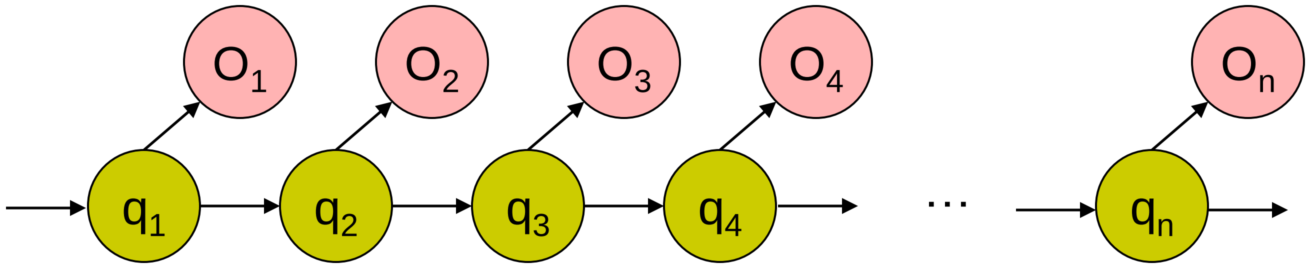
Consider one such fixed state sequence

$$Q = q_1 q_2 \cdots q_T$$

The observation sequence ***O*** for the ***Q*** is

$$P(O \mid Q, \lambda) = \prod_{t=1}^{T} P(O_t \mid q_t, \lambda)$$

$$= b_{q_1}(O_1) \cdot b_{q_2}(O_2) \cdots b_{q_T}(O_T)$$



Calculating with Hidden Markov Model (cont.)

The probability of such a state sequence Q

$$P(Q \mid \lambda) = a_{0q_1} a_{q_1 q_2} \cdot a_{q_2 q_3} \cdots a_{q_{T-1} q_T}$$

The probability that O and Q occur simultaneously, is simply the product of the above two terms, i.e.,

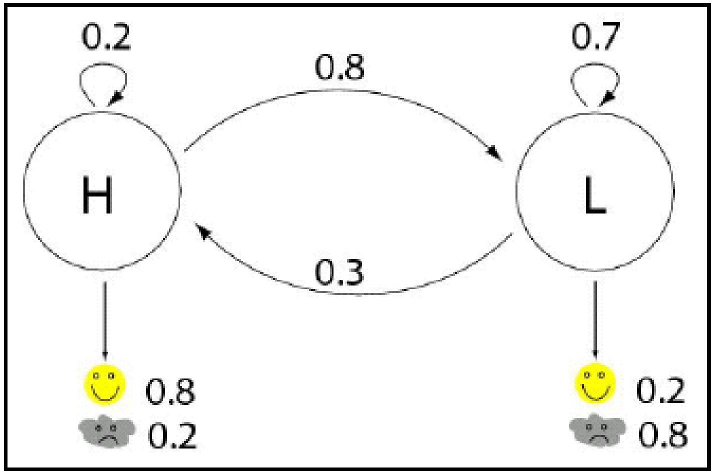
$$P(O, Q \mid \lambda) = P(O \mid Q, \lambda) P(Q \mid \lambda)$$

$$P(O, Q \mid \lambda) = a_{0q_1} b_{q_1}(O_1) a_{q_1 q_2} b_{q_2}(O_2) a_{q_2 q_3} \cdots a_{q_{T-1} q_T} b_{q_T}(O_T)$$

Example

$$
\begin{aligned}
&P(x,\pi)\\
&=\left(a_{0L}e_L(R)\right)\left(a_{LL}e_L(R)\right)\left(a_{LL}e_L(S)\right)\left(a_{LL}e_L(R)\right)\left(a_{LH}e_H(S)\right)\left(a_{HH}e_H(S)\right)\left(a_{HL}e_L(R)\right)\\
&=\left(\frac{8}{11}\frac{8}{10}\right)\left(\frac{7}{10}\frac{8}{10}\right)\left(\frac{7}{10}\frac{2}{10}\right)\left(\frac{7}{10}\frac{8}{10}\right)\left(\frac{3}{10}\frac{8}{10}\right)\left(\frac{2}{10}\frac{8}{10}\right)\left(\frac{8}{10}\frac{8}{10}\right)\\
&=0.0006278
\end{aligned}
$$

Hidden states: L L L L H H L
Observations:



The three main questions on HMMs

1. **Evaluation**

GIVEN a HMM $M=(S, V, A, B, \pi)$, and a sequence O,
FIND $P[O|M]$

2. **Decoding**

GIVEN a HMM $M=(S, V, A, B, \pi)$, and a sequence O,
FIND the sequence Q of states that maximizes $P(O, Q \mid \lambda)$

3. **Learning**

GIVEN a HMM $M=(S, V, A, B, \pi)$, with unspecified transition/emission probabilities and a sequence Q,
FIND parameters $\theta = (e_i(.), a_{ij})$ that maximize $P[x|\theta]$

Evaluation

- Find the likelihood a sequence is generated by the model

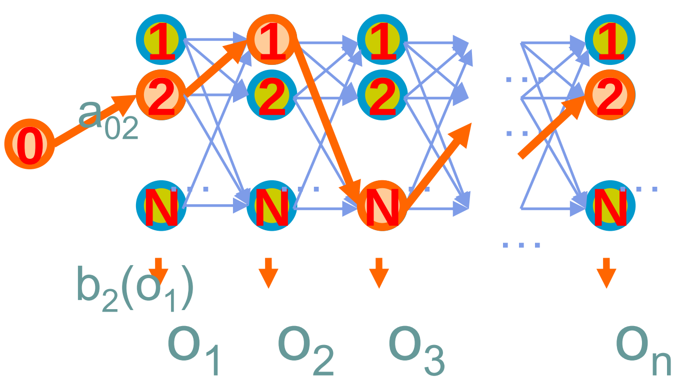
- A straightforward way （穷举法）
  - The probability of *O* is obtained by summing all possible state sequences *q* giving

$$P(O|\lambda) = \sum_{all\,Q} P(O|Q,\lambda)P(Q|\lambda)$$

$$= \sum_{q_1,q_2,\ldots q_T} \pi_{q_1} b_{q_1}(O_1) a_{q_1q_2} b_{q_2}(O_2) a_{q_2q_3} \cdots a_{q_{T-1}q_T} b_{q_T}(O_T)$$

Complexity is *O(N^T)*

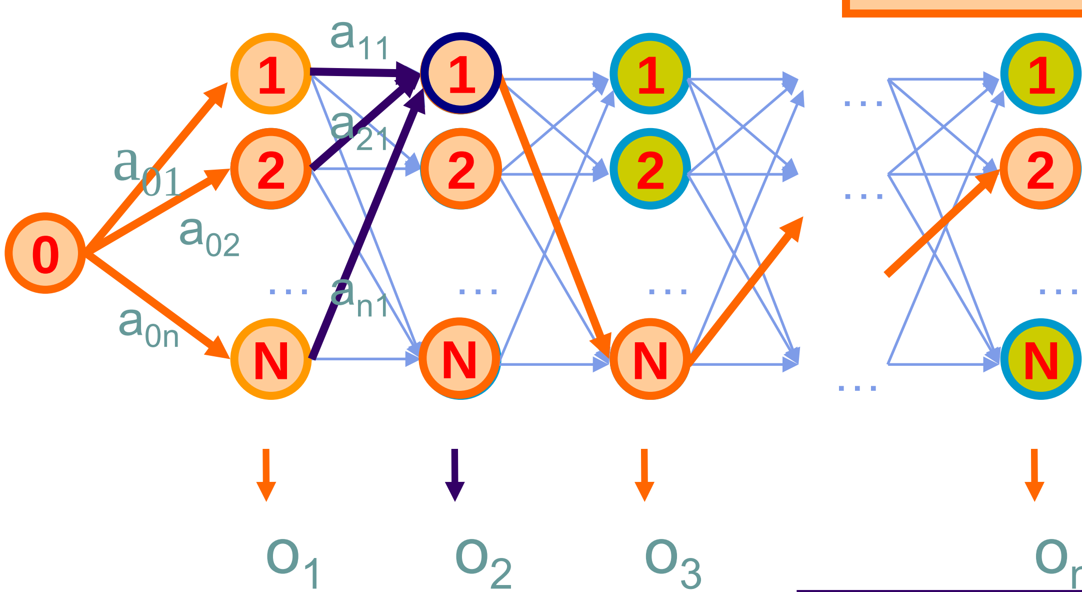
Calculations is unfeasible



The Forward Algorithm

- A more elaborate algorithm
  - The Forward Algorithm

$$P(O \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i)$$



$$\alpha_2(1) = [\sum_{i=1}^{N} \alpha_1(i) a_{i1}] b_1(O_2)$$

$$P(O_1 O_2 \mid \lambda) = \sum_{i=1}^{N} \alpha_2(i)$$

The Forward Algorithm

The Forward variable

$$\alpha_t(i) = P(O_1 O_2 \cdots O_t, q_t = S_i \mid \lambda)$$

We can compute $\alpha(i)$ for all N, i,

Initialization:

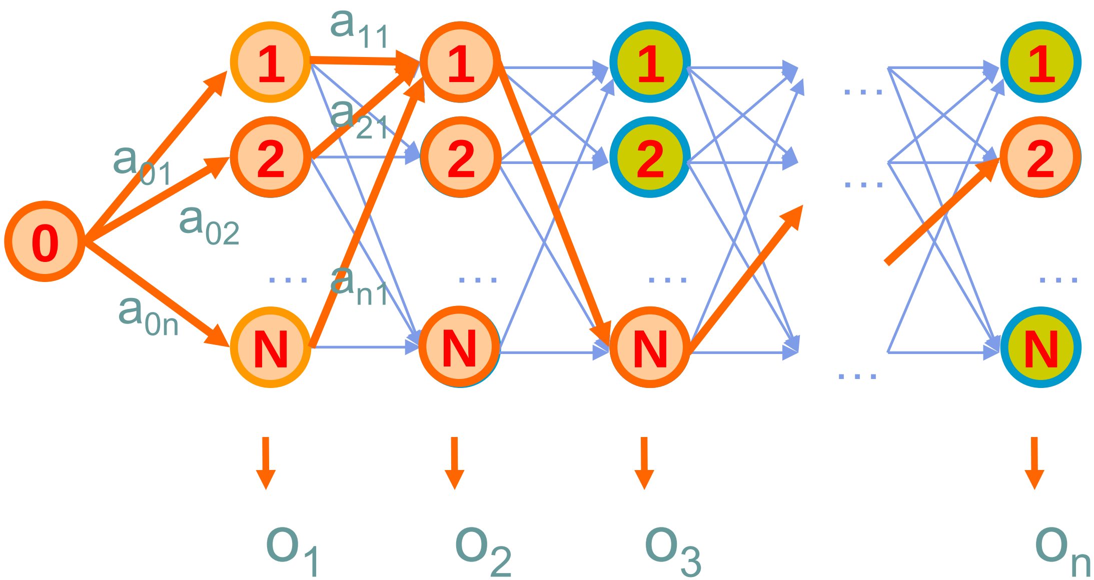
$$\alpha_1(i) = a_{0i} b_{0i}(O_1) \qquad i = 1 \ldots N$$

Iteration:

$$\alpha_{t+1}(i) = [\sum_{i=1}^{N} \alpha_t(i) a_{ij}] b_j(O_{t+1}) \qquad t = 1 \ldots T-1$$

Termination:

$$P(O \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i)$$



The Backward Algorithm

The backward variable

$$\beta_t(i) = P(O_{t+1}O_{t+2}\cdots O_T \mid q_t = S_i, \lambda)$$

Similar, we can compute backward variable for all *N*, *i*,

Initialization:

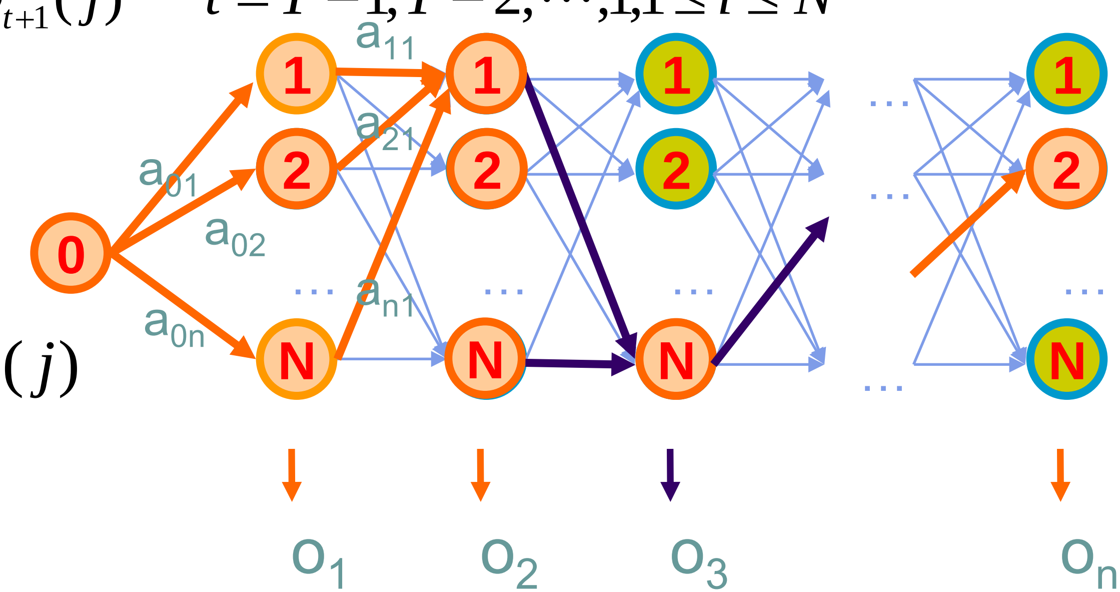
$$\beta_T(i) = 1,\ \ i = 1, \dots, N$$

Iteration:

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1})\beta_{t+1}(j) \qquad t = T-1, T-2, \cdots, 1, 1 \le i \le N$$

Termination:

$$P(O \mid \lambda) = \sum_{j=1}^{N} a_{0j} b_1(O_1)\beta_1(j)$$



Consider $\alpha_T(i) = P(O_1 O_2 \ldots O_T, q_T = S_i | \lambda)$

Thus $P(q_T = S_i | O) = \dfrac{P(O, q_T = S_i)}{P(O)} = \dfrac{\alpha_T(i_T)}{\sum_i \alpha_T(i_T)}$

Also $P(q_t = S_i | O) = \dfrac{P(\mathrm{O}, q_t = S_i)}{P(\mathrm{O})}$

$$= \frac{P(O_1 O_2 \cdots O_t, q_t = S_{i_t}, O_{t+1} O_{t+2} \cdots O_T)}{P(\mathrm{O})}$$

$$= \frac{P(O_1 O_2 \cdots O_t, q_t = S_i) \mathrm{P}(O_{t+1} O_{t+2} \cdots O_T | O_1 O_2 \cdots O_t, q_t = S_i)}{P(\mathrm{O})}$$

Forward, $\alpha_k(i)$

Backward, $\beta_k(i)$

$$= \frac{\boxed{P(O_1 O_2 \cdots O_t, q_t = S_i)} \boxed{\mathrm{P}(O_{t+1} O_{t+2} \cdots O_T | q_t = S_i)}}{P(\mathrm{O})}$$

$$= \frac{\alpha_t(i)\beta_t(i)}{\sum_i \alpha_T(i)} = \gamma(i)$$

Decoding

GIVEN a HMM, and a sequence O.

Suppose that we know the parameters of the Hidden Markov Model and the observed sequence of observations $O_1, \mathrm{O}_2, \ldots, O_T$.

FIND the sequence Q of states that maximizes

$P(Q|O,\lambda)$

Determining the sequence of States $q_1, q_2, \ldots, q_T$, which is optimal in some meaningful sense. (i.e. best “explain” the observations)

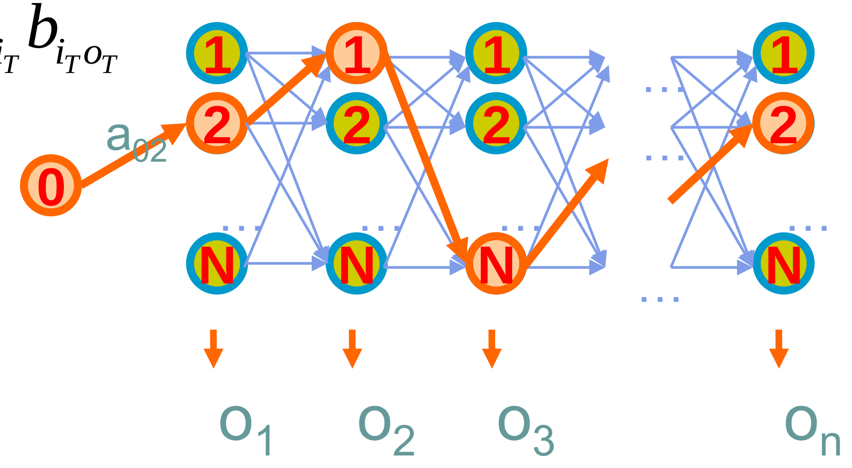
Decoding

Consider $P(Q|O,\lambda) = \dfrac{P(\mathrm{O},Q|\lambda)}{P(\mathrm{O}|\lambda)}$

To maximize the above probability is equivalent to maximizing

$$P(\mathrm{O},Q|\lambda)$$

$$= a_{i_1} b_{i_1 o_1} a_{i_1 i_2} b_{i_2 o_2} a_{i_2 i_3} b_{i_3 o_3} \ldots a_{i_{T-1} i_T} b_{i_T o_T}$$



A best path finding problem

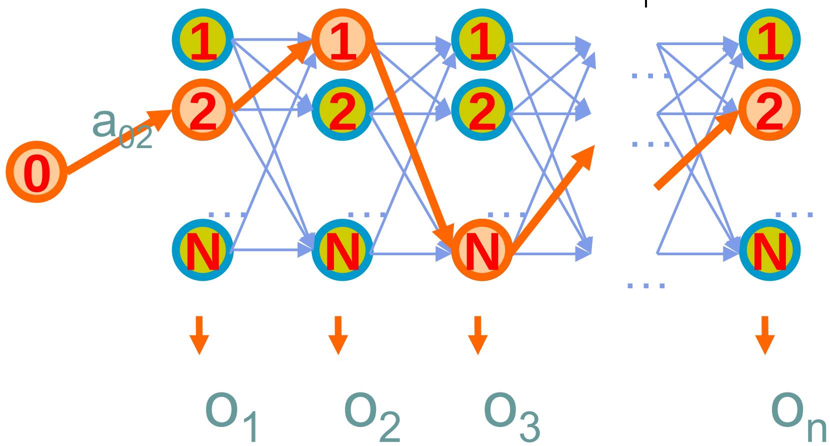
$$\max P(\mathrm{O},Q|\lambda)$$

$$= \max \ln(P(\mathrm{O},Q|\lambda))$$

$$= \max\left(\ln\left(a_{i_1} b_{i_1 o_1}\right) + \ln\left(a_{i_1 i_2} b_{i_2 o_2}\right) \ldots + \ln\left(a_{i_{T-1} i_T} b_{i_T o_T}\right)\right)$$

Viterbi Algorithm

[Dynamic programming]



Initialization:

$\delta_1(i) = a_{0i} b_i(O_1)$, $\quad i = 1 \ldots N$

$\psi_1(i) = 0.$

Recursion:

$\delta_t(j) = \max_i [\delta_{t-1}(i)\, a_{ij}] b_j(O_t)$ $\quad t=2\ldots T \quad j=1\ldots N$

$\psi_1(j) = \text{argmax}_i [\delta_{t-1}(i)\, a_{ij}]$ $\quad t=2\ldots T \quad j=1\ldots N$

Termination:

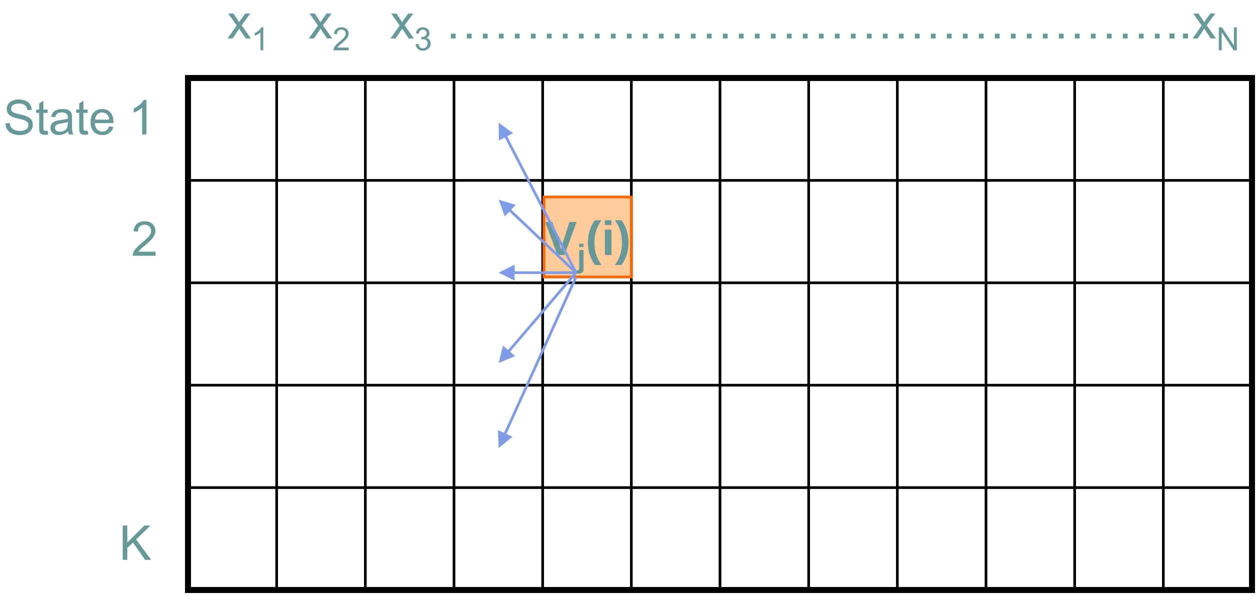
$P^* = \max_i \delta_T(i)$

$q_T^* = \text{argmax}_i [\delta_T(i)]$

Traceback:

$q_t^* = \psi_1(q^*_{t+1})$ $\quad t=T-1, T-2, \ldots, 1.$

The Viterbi Algorithm



Similar to "aligning" a set of states to a sequence

<u>Time:</u> $O(K^2N)$

<u>Space:</u> $O(KN)$

Learning

- Estimation of Parameters of a Hidden Markov Model
  1. Both the sequence of observations O and the sequence of states Q is observed

     learning $\lambda = (A, B, \pi)$
  2. Only the sequence of observations O are observed

     learning Q and $\lambda = (A, B, \pi)$

Maximal Likelihood Estimation

- Given O and Q, the Likelihood is given by:

$$L(A,B,\pi) = a_{i_1} b_{i_1 o_1} a_{i_1 i_2} b_{i_2 o_2} a_{i_2 i_3} b_{i_3 o_3} \ldots a_{i_{T-1} i_T} b_{i_T o_T}$$

Maximal Likelihood Estimation

- the log-Likelihood is:

$$l(A,B,\pi) = \ln L(A,B,\pi) = \ln\left(a_{i_1}\right) + \ln\left(b_{i_1 o_1}\right) + \ln\left(a_{i_1 i_2}\right)$$

$$+\ln\left(a_{i_2 i_3}\right) + \ln\left(b_{i_3 o_3}\right) \ldots + \ln\left(a_{i_{T-1} i_T}\right) + \ln\left(b_{i_T o_T}\right)$$

$$= \sum_{i=1}^{M} f_{i0} \ln\left(a_i\right) + \sum_{i=1}^{M}\sum_{j=1}^{M} f_{ij} \ln\left(a_{ij}\right) + \sum_{i=1}^{M}\sum_{o(i)} \ln\left(b_{io}\right)$$

where $f_{i0} =$ the number of times state i occurs in the first state

$f_{ij} =$ the number of times state i changes to state j.

$\beta_{iy} = f\left(y \middle| \theta_i\right)$ (or $p\left(y \middle| \theta_i\right)$ in the discrete case)

$\sum_{o(i)} \square =$ the sum of all observations o_t where $q_t = S_i$

Maximal Likelihood Estimation

In such case these parameters computed by *Maximum Likelihood Estimation* are:

$$\hat{a}_i = \frac{f_{i0}}{1} \qquad \hat{a}_{ij} = \frac{f_{ij}}{\sum_{j=1}^{M} f_{ij}}, \text{ and}$$

$\hat{b}_i$ = the MLE of b_i computed from the observations o_t where $q_t = S_i$.

Maximal Likelihood Estimation

- Only the sequence of observations O are observed

$$L(A,B,\pi) = \sum_{i_1, i_2 \ldots i_T} a_{i_1} b_{i_1 o_1} a_{i_1 i_2} b_{i_2 o_2} a_{i_2 i_3} b_{i_3 o_3} \ldots a_{i_{T-1} i_T} b_{i_T o_T}$$

  - It is difficult to find the Maximum Likelihood Estimates directly from the Likelihood function.
  - The Techniques that are used are

    1. ***The Segmental K-means Algorith***
    2. ***The Baum-Welch (E-M) Algorithm***

The Baum-Welch Algorithm

- The E-M algorithm was designed originally to handle "Missing observations".
- In this case the missing observations are the states $\{q_1, q_2, \ldots, q_T\}$.
- Assuming a model, the states are estimated by finding their expected values under this model. (The E part of the E-M algorithm).

The Baum-Welch Algorithm

- With these values the model is estimated by Maximum Likelihood Estimation (The M part of the E-M algorithm).

- The process is repeated until the estimated model converges.

The Baum-Welch Algorithm

Initialization:

Pick the best-guess for model parameters (or arbitrary)

Iteration:

Forward

Backward

Calculate A_{kl}, $E_k(b)$

Calculate new model parameters a_{kl}, $e_k(b)$

Calculate new log-likelihood $P(x \mid \theta)$

GUARANTEED TO BE HIGHER BY EXPECTATION-MAXIMIZATION

Until $P(x \mid \theta)$ does not change much

The Baum-Welch Algorithm

Let $f(O,Q|\lambda) = L(O,Q,\lambda)$ denote the joint distribution of Q,O. Consider the function:

$$Q(\lambda,\lambda') = E_{\mathbf{X}}\left(\ln L(O,Q,\lambda)\middle|Q,\lambda'\right)$$

Starting with an initial estimate of λ $\left(\lambda^{(1)}\right)$.

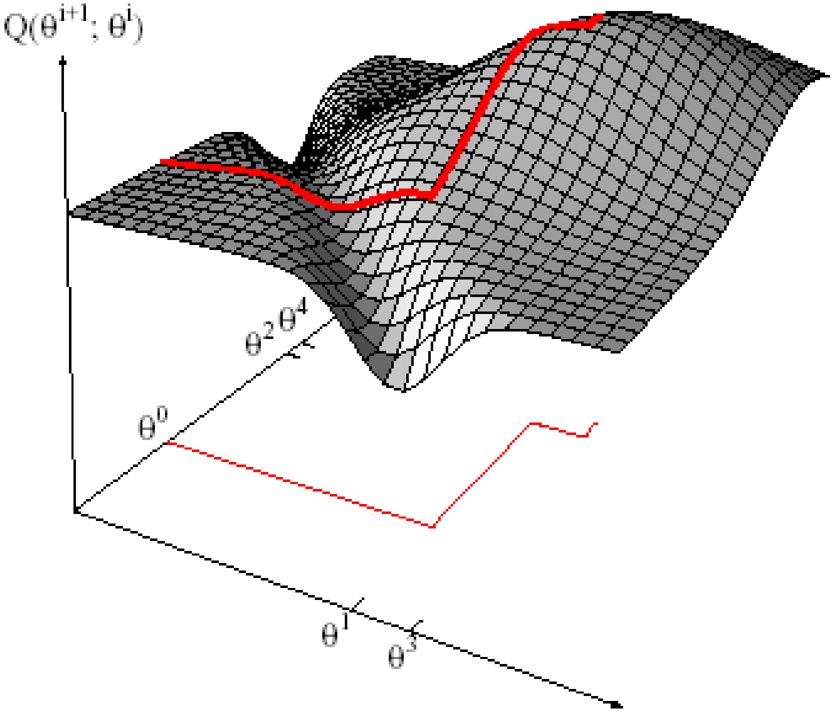
A sequence of estimates $\left\{\lambda^{(m)}\right\}$ are formed by finding $\lambda = \lambda^{(m+1)}$ to maximize $Q\left(\lambda,\lambda^{(m)}\right)$ with respect to λ.

The Baum-Welch Algorithm

The sequence of estimates $\left\{\lambda^{(m)}\right\}$

converge to a local maximum of the likelihood

$$L(Q,\lambda) = f(Q|\lambda)$$



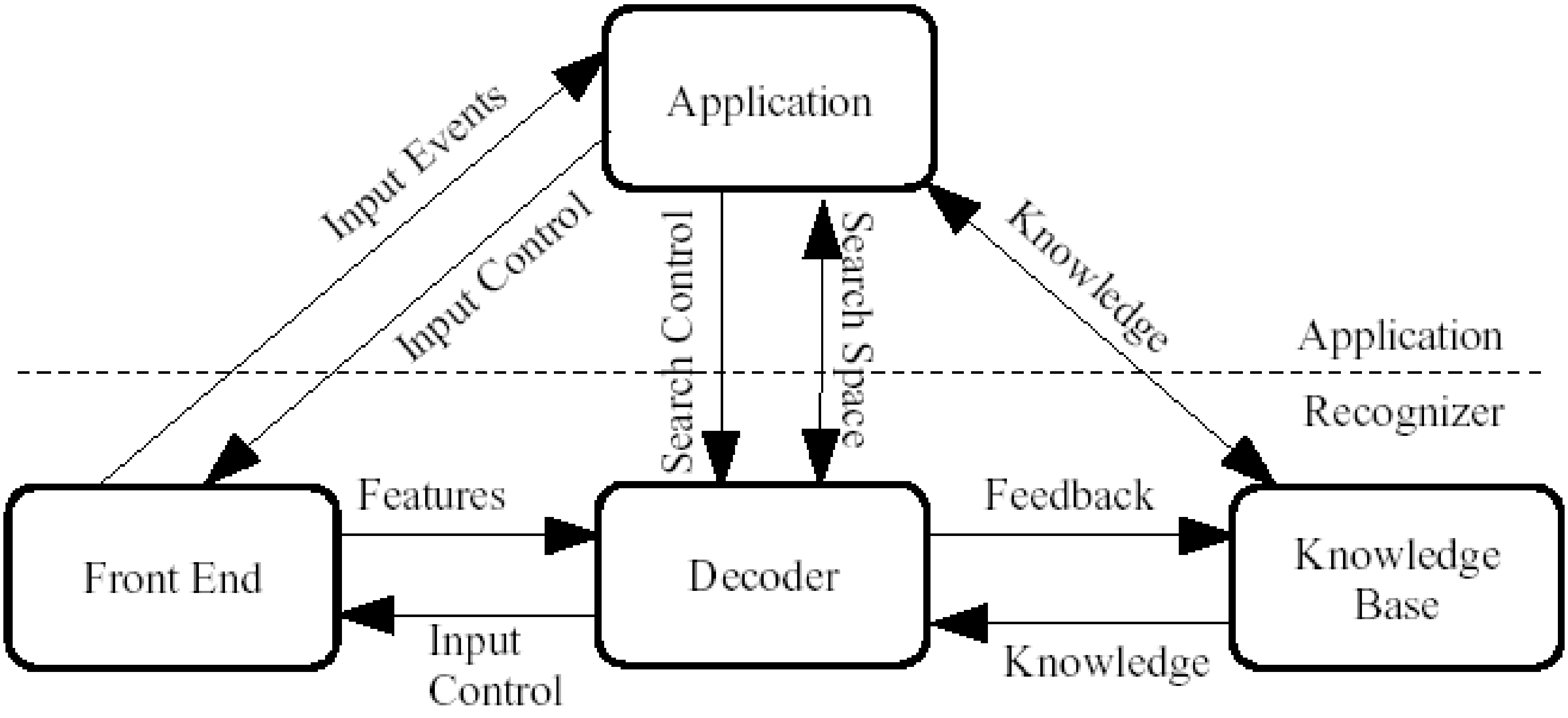
Speech Recognition

- On-line documents of Java™ Speech API
  - http://java.sun.com/products/java-media/speech/
- On-line documents of Free TTS
  - http://freetts.sourceforge.net/docs/
- On-line documents of Sphinx-II
  - http://www.speech.cs.cmu.edu/sphinx/

Brief History of CMU Sphinx

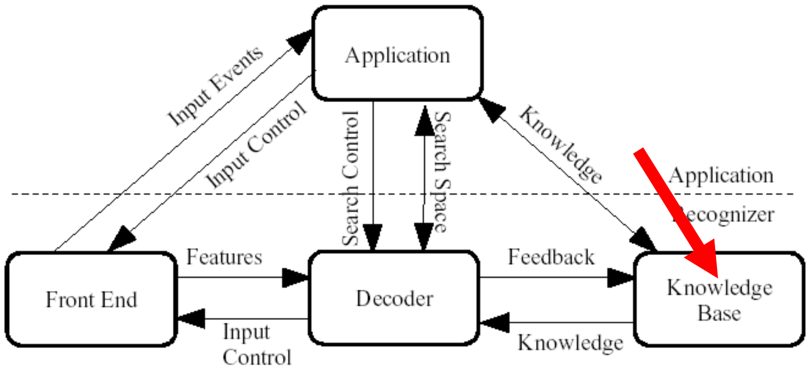
- Sphinx-I (1987)
  - The first user independent, high performance ASR of the world.
  - Written in C by Kai-Fu Lee (李開復博士，現任Google副總裁).
- Sphinx-II (1992)
  - Written by Xuedong Huang in C. (黃學東博士，現為Microsoft Speech.NET團隊領導人)
  - 5-state HMM / N-gram LM.
- Sphinx-III (1996)
  - Built by Eric Thayer and Mosur Ravishankar.
  - Slower than Sphinx-II but the design is more flexible.
- Sphinx-4 (Originally Sphinx 3j)
  - Refactored from Sphinx 3.
  - Fully implemented in Java. (Not finished yet …)

Components of CMU Sphinx



Knowledge Base

- The data that drives the decoder.
- Three sets of data
  - Acoustic Model.
  - Language Model.
  - Lexicon (Dictionary).



Speech Recognition Architecture

- Observations : $O = o_1, o_2, o_3, \cdots, o_t$
- Word Sequences : $W = w_1, w_2, w_3, \cdots, w_n$
- Probabilistic implementation can be expressed :

$$\hat{W} = \arg\max_{W \in L} P(W \mid O)$$

- Then we can use Bayes' rule to break it down :

$$\hat{W} = \arg\max_{W \in L} P(W \mid O) = \arg\max_{W \in L} \frac{P(O \mid W)P(W)}{P(O)}$$

$$\left(\begin{array}{l} \because P(W \mid O) = \dfrac{P(WO)}{P(O)} \quad \mathit{and} \quad P(O \mid W) = \dfrac{P(WO)}{P(W)} \\ \therefore P(W \mid O) \cdot P(O) = P(WO) = P(O \mid W) \cdot P(W) \end{array} \right)$$

Speech Recognition Architecture

- For each potential sentence we are still examining the same observations O, which must have the same probability $P(O)$.

$$\hat{W} = \arg\max_{W \in L} P(W \mid O)$$

Posterior probability

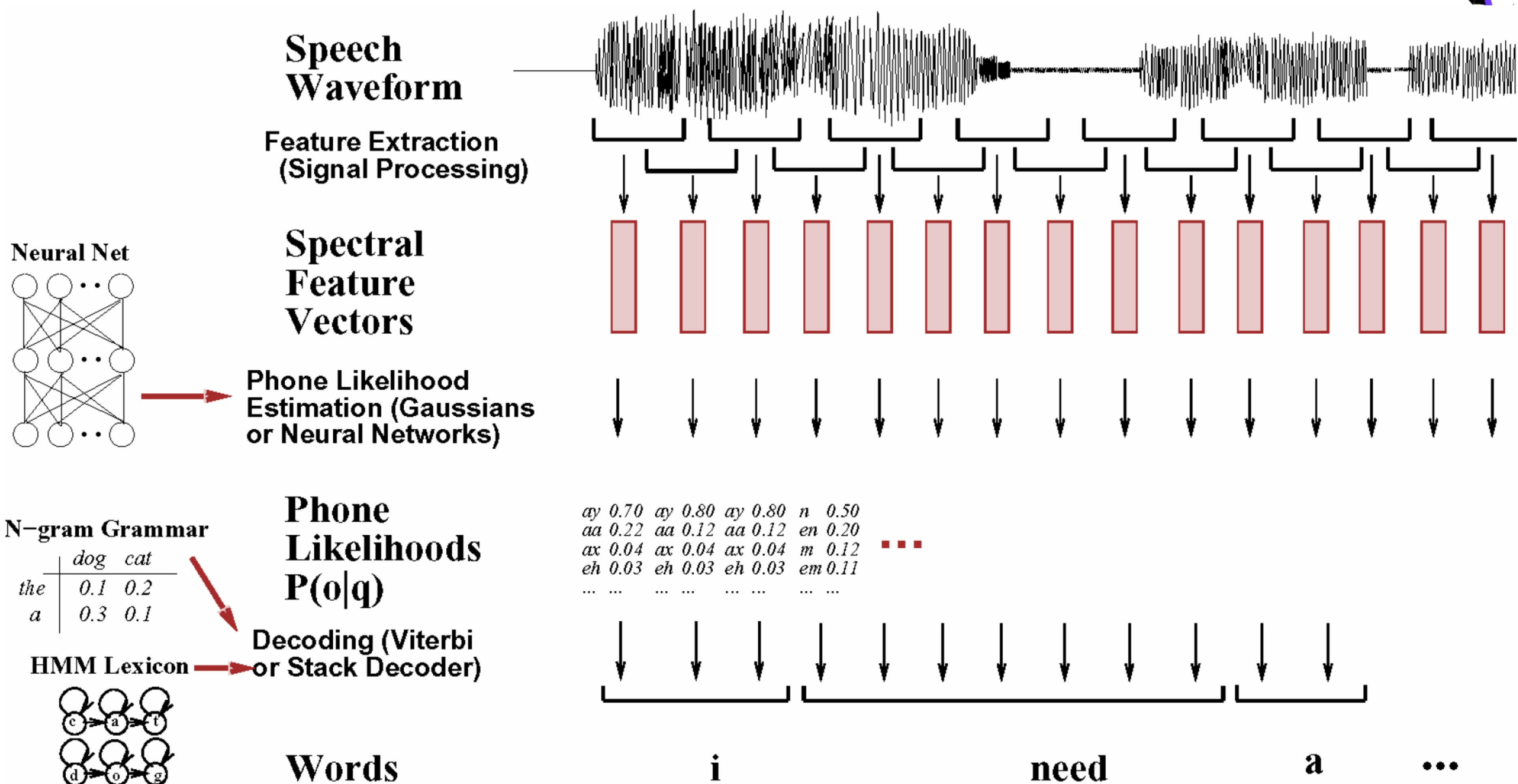
$$= \arg\max_{W \in L} \frac{P(O \mid W)P(W)}{P(O)} = \arg\max_{W \in L} P(O \mid W)P(W)$$

Observation likelihood
Acoustic model

Prior probability
Language model

Speech Recognition Architecture

↓ Figure 7.2 Schematic architecture for a speech recognition



Acoustic Model

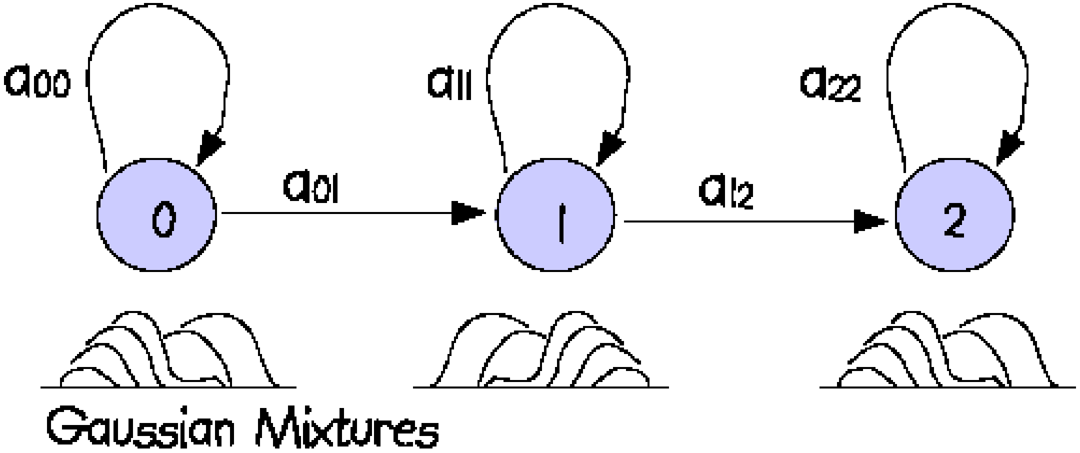
- /model/hmm/6k
- Database of statistical model.
- Each statistical model represents a phoneme.
- Acoustic Models are trained by analyzing large amount of speech data.

HMM in Acoustic Model

- HMM represent each unit of speech in the Acoustic Model.
- Typical HMM use 3-5 states to model a phoneme.
- Each state of HMM is represented by a set of *Gaussian mixture density functions*.
- Sphinx2 default phone set.

Mixture of Gaussians

- Represent each state in HMM.
- Each set of Gaussian Mixtures are called “senones”.
- HMM can share “senones”.



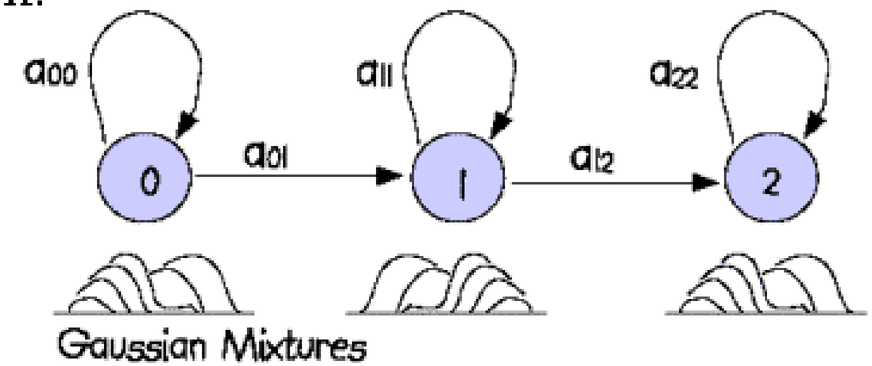
Mixture of Gaussians

$$N(x;\mu,\Sigma)=\frac{1}{(2\pi)^{n/2}\,|\Sigma|^{1/2}}\exp\left[-\frac{1}{2}(x-\mu)^t\Sigma^{-1}(x-\mu)\right]$$

$$f(x)=\sum_{k=1}^{K}c_k N_k(x;\mu_k,\Sigma_k)$$ 其中

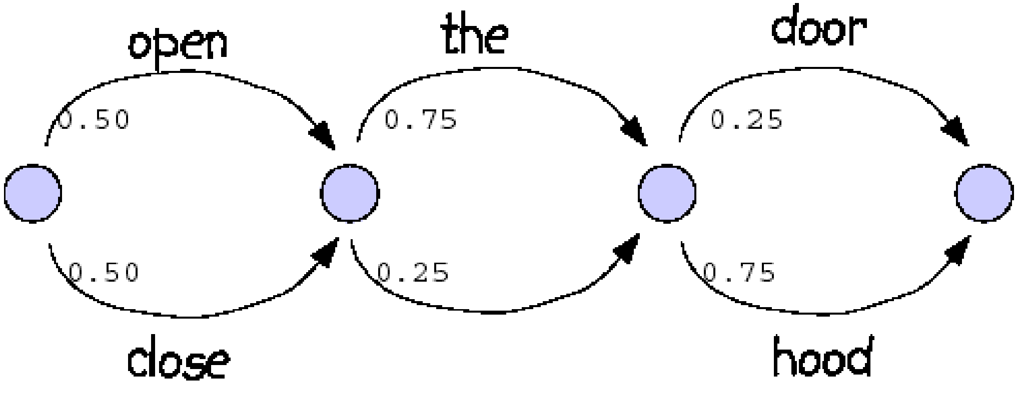
$$c_k\geq 0 \quad 且 \quad \sum_{k=1}^{K}c_k=1$$

Gaussian mixtures with enough mixture components can approximate any distribution.



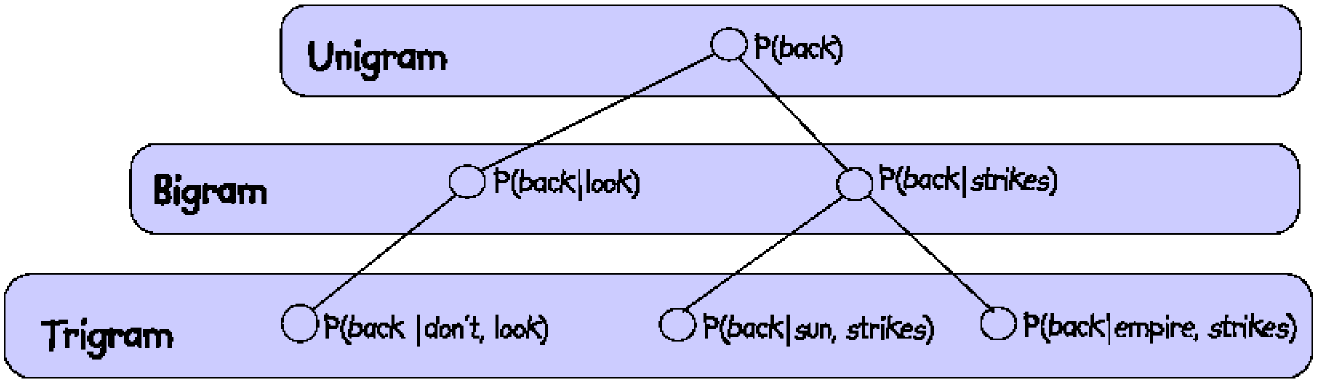
Language Model

- Describes what is likely to be spoken in a particular context
- Word transitions are defined in terms of transition probabilities
- Helps to constrain the search space



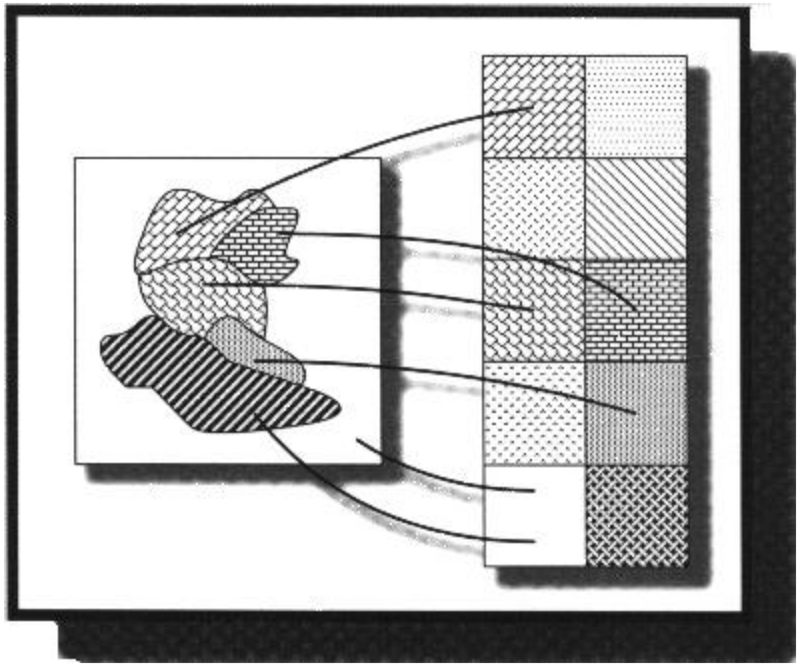
N-gram Language Model

- Probability of word N dependent on word N-1, N-2, ...
- Bigrams and trigrams most commonly used
- Used for large vocabulary applications such as dictation
- Typically trained by very large (millions of words) corpus



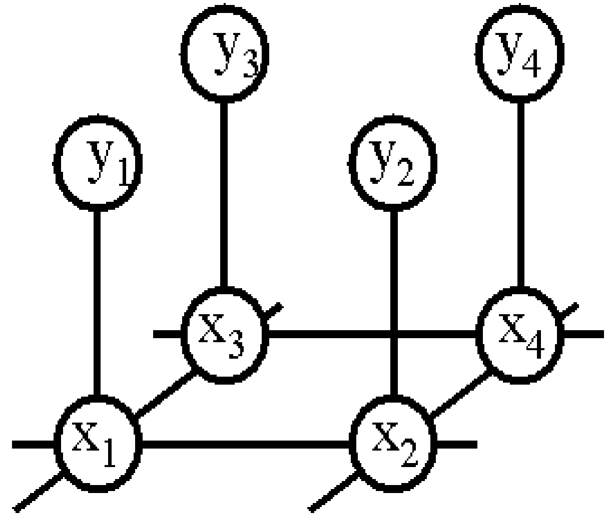
Markov Random field

- See webpage
- http://www.nlpr.ia.ac.cn/users/szli/MRF_Book/MRF_Book.html



Belief Network (Propagation)

Y. Weiss and W. T. Freeman
Correctness of Belief Propagation in Gaussian Graphical Models of Arbitrary Topology. in: Advances in Neural Information Processing Systems 12, edited by S. A. Solla, T. K. Leen, and K-R Muller, 2000. [MERL-TR99-38.]



Homework

- Read the motion texture siggraph paper.