Machine Learning in Robotics
Lecture 9-10: Hidden Markov models

Prof. Dongheui Lee

Institute of Automatic Control Engineering
Technische Universität München

dhlee@tum.de

Discrete Markov Processes

Consider a system described by the following process

- At any given time, the system can be in one of N possible states $\{s_1, s_2, ..., s_N\}$
- At regularly spaced times, the system undergoes a transition to a new state
- Transition between states can be described probabilistically

In general, the probability that the system is in state $q_t = s_j$ will be a function of the complete history of the system

- To simplify the analysis, however, it is common to assume that the state of the system at time t depends only on the state at time $t-1$

$$p(q_t = s_j | q_{t-1} = s_i, q_{t-2} = s_k, ...) = p(q_t = s_j | q_{t-1} = s_i)$$

- This is known as a first-order Markov Process
- We assume that the transition probability between any two states is independent of time

$$a_{ij} = p(q_t = s_j | q_{t-1} = s_i)$$

An example

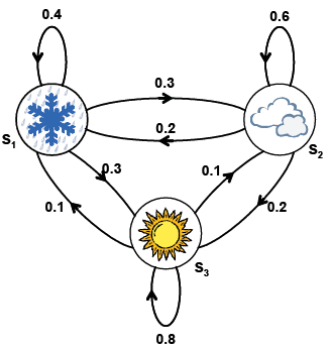
Consider a simple three-state Markov model of the weather.

- Any given day, the weather can be described as being
  - State 1: Rain or Snow
  - State 2: Cloudy
  - State 3: Sunny
- Transitions between states are described by the transition matrix

a_ij: from i to j

$$A = \{a_{ij}\} = \begin{bmatrix} 0.4 & 0.3 & 0.3 \\ 0.2 & 0.6 & 0.2 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}$$

- This model can be described by the following directed graph



An example

- Q1. Given that the weather on day t=1 is sunny, what is the probability that the weather for the next 7 days will be ”sun, sun, rain, rain, sun, clouds, sun”?

$$
\begin{aligned}
&p(q_2 = s, q_3 = s, q_4 = r, q_5 = r, q_6 = s, q_7 = c, q_8 = s | q_1 = s) \\
&= p(q_2 = s|q_1 = s)p(q_3 = s|q_2 = s)p(q_4 = r|q_3 = s)p(q_5 = r|q_4 = r) \\
&p(q_6 = s|q_5 = r)p(q_7 = c|q_6 = s)p(q_8 = s|q_7 = c) \\
&= (0.8)(0.8)(0.1)(0.4)(0.3)(0.1)(0.2)
\end{aligned}
$$

- Q2. What is the probability that the weather stays in the same known state s_i for exactly T consecutive days?

从第T天开始不再满足 $a_{ii}^{T-1}(1 - a_{ii})$

Discrete Markov Processes

The previous model assumes that each state can be uniquely associated with an observable event

检索

- Once an observation is made, the state of the system is then trivially retrieved
- This model, however, is too restrictive to be of practical use for most realistic problems

限制性的

To make the model more flexible, we will assume that the outcomes or observations of the model are a probabilistic function of each state

- Each state can produce a number of outputs according to a unique probability distribution, and each distinct output can potentially be generated at any state
- These are known a Hidden Markov Models (HMM), because the state sequence is not directly observable, it can only be approximated from the sequence of observations produced by the system

An example: Weather

Markov model $\rightarrow$ Hidden Markov model:

- Hide the weather!!
- Assume that one cannot see the weather directly. But he has only limited information to guess the weather, which is observing an umbrella of a person.

weather weather	Probability to have an umbrella
sun	0.1
rain	0.8
cloud	0.3

In Markov process, direct observation of the weather is possible. Therefore, the probability to observe a sequence of weather can be expressed as

$$p(q_1, q_2, ..., q_T) = p(q_1) \prod_{t=2}^{T} p(q_t|q_{t-1})$$

But, in HMM, the weather is hidden. The probability of the weather becomes

$$p(q_i|o_i) = \frac{p(o_i|q_i)p(q_i)}{p(o_i)}$$

Given a sequence of observations (umbrella), the conditional probability of the weather sequence is

$$p(q_1, ..., q_t|o_1, ..., o_t) \propto \prod_{t=1}^{t} [p(o_i|q_i)] \prod_{i=2}^{t} [p(q_i|q_{i-1})]p(q_1)$$

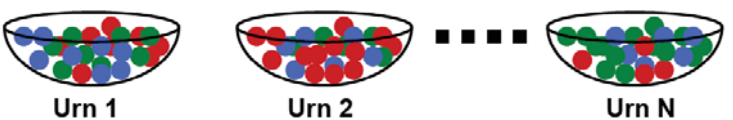
Example: The urn-ball problem

To further illustrate the concept of an HMM, consider this scenario

- You are placed in the same room with a curtain
- Behind the curtain there are N urns, each containing a large number of balls with M different colors
- The person behind the curtain selects an urn according to an internal random process, then randomly grabs a ball from the selected urn
- He shows you the ball, and places it back in the urn
- This process is repeated over and over

Questions?

- How would you represent this experiment with an HMM?
- What are the states? Why are the states hidden? Urn number
- What are the observations? Color of the ball



Elements of a Hidden Markov Model

A Hidden Markov Model is characterized by the following

- N, the number of states in the model $\{s_1, s_2, ..., s_N\}$
- M, The number of discrete observation $\{v_1, v_2, ..., v_M\}$
- $A = \{a_{ij}\}$, the transition probability

$$a_{ij} = p(q_t = s_j | q_{t-1} = s_i) \qquad \sum_j a_{ij} = 1, \forall i$$

- $B = b_j(k)$, the observation probability distribution

$$b_j(k) = p(o_t = v_k | q_t = s_j) \qquad \sum_k b_j(k) = 1, \forall j$$

- π_i, the initial state distribution $\pi_i = p(q_1 = s_i), \quad \sum_i \pi_i = 1$

Therefore, an HMM is specified by two scalars (M, N) and three probability distributions (A, B, π). we will represent an HMM by the compact notation $\lambda = (A, B, \pi)$

HMM generation of observation sequences

Given a completely specified HMM $\lambda = (A, B, \pi)$, how can an observation sequence $\mathcal{O} = \{o_1, o_2 ... o_T\}$ be generated?

1. Choose an initial state according to the initial state distribution: $q_1 = \arg\max_i \pi_i$
2. Set $t = 1$
3. Generate and observation o_t according to the observation distribution $b_j(k)$: $o_t = \arg\max_k b_{q_t}(k)$
4. Move to a new state according to the state transition distribution a_{ij}
5. Set $t = t + 1$ and return to step 3 until $t \geq T$

The three basic HMM problems

- **Probability Evaluation problem**
  Determine the probability $p(\mathcal{O}|\lambda)$ that a particular sequence of visible states $\mathcal{O} = \{o_1, o_2 ... o_T\}$ was generated by the model $\lambda = \{A, B, \pi\}$.
  $\Rightarrow$ The solution is given by the Forward and Backward procedures.
- **Optimal State Sequence**
  Given a model λ and a sequence of observations, determine the most likely sequence of hidden states that led to the observation sequence.
  $\Rightarrow$ The solution is provided by the Viterbi algorithm.
- **Parameter Estimation**
  Given a set of observations (training dataset), determine the parameters of the HMM $\lambda = \{A, B, \pi\}$ to maximize the likelihood $p(\mathcal{O}|\lambda)$.
  $\Rightarrow$ Baum-Welch re-estimation procedure

Problem 1: Probability Evaluation

Our goal is to compute the likelihood of an observation sequence $\mathcal{O} = \{o_1, o_2 ... o_T\}$ given a particular HMM model defined by $\lambda = (A, B, \pi)$

- Computation of this probability involves enumerating every possible state sequence and evaluating the corresponding probability

$$p(\mathcal{O}|\lambda) = \sum_{\forall \mathcal{Q}} p(\mathcal{O}, \mathcal{Q}|\lambda) = \sum_{\forall \mathcal{Q}} p(\mathcal{O}|\mathcal{Q}, \lambda) p(\mathcal{Q}|\lambda)$$

- For a particular state sequence $\mathcal{Q} = \{q_1, q_2 ...\}$ the probability $p(\mathcal{O}|\mathcal{Q}, \lambda)$ is

$$p(\mathcal{O}|\mathcal{Q}, \lambda) = \prod_{t=1}^{T} p(o_t|q_t, \lambda) = \prod_{t=1}^{T} b_{q_t}(o_t)$$

- The probability of the state sequence $\mathcal{Q}$ is

$$p(\mathcal{Q}|\lambda) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} ... a_{q_{T-1} q_T}$$

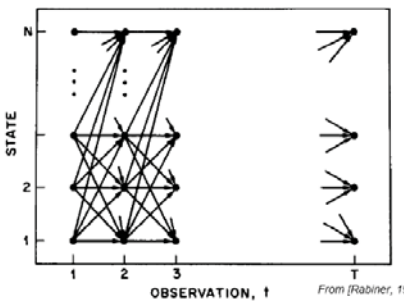
- Merging these results, we obtain

$$P(\mathcal{O}|\lambda) = \sum_{q_1, q_2, \ldots, q_T} \pi_{q_1} b_{q_1}(o_{q_1}) a_{q_1 q_2} b_{q_2}(o_{q_2}) ... a_{q_{T-1} q_T} b_{q_T}(o_{q_T})$$

Problem 1: Probability Evaluation

Computational complexity

- With N^T possible state sequences, this approach becomes unfeasible even for small problems
  - For $N = 5$ and $T = 100$, the method would require the order of 10^{72} computations !!
- Fortunately, the computation of $p(\mathcal{O}|\lambda)$ has the lattice (格子) (or trellis (格子结构)) structure shown below, which lends itself to a very efficient implementation known as the *Forward procedure*



Problem 1: The Forward procedure

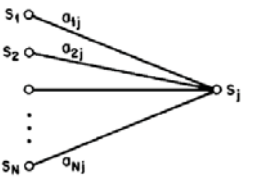
Consider the following variable $\alpha_t(i)$ defined as *Forward Variable*
$\alpha_t(i) = p(o_1, o_2, ..., o_t, q_t = s_i|\lambda)$ which represents the probability of the observation sequence up to time t and the state at time t to be s_i, given the model λ

Computation of this variable can be efficiently performed by induction 归纳法

- Initialization: $\alpha_1(i) = \pi_i b_i(o_1)$
- Induction:

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(o_{t+1}) \quad \begin{cases} 1 \le t \le T-1 \\ 1 \le j \le N \end{cases}$$

- Termination: $p(\mathcal{O}|\lambda) = \sum_{i=1}^{N} \alpha_T(i)$



From [Rabiner, 1989]

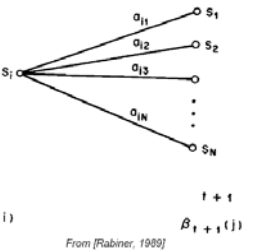
As a result, computation of $p(\mathcal{O}|\lambda)$ can be reduced from $2T \times N^T$ down to $N^2 \times T$ operations (from 10^{72} to 3000 for $N = 5$, $T = 100$)

Problem 1: The Backward procedure

In analogy to the forward procedure, consider the *backward variable* $\beta_t(i)$ defined as $\beta_t(i) = p(o_{t+1}, o_{t+2}, ..., o_T | q_t = s_i, \lambda)$.
$\beta_t(i)$ represents the probability of the partial observation sequence from $t+1$ to the end, given state s_i at time t and the model λ
Computation of $\beta_t(i)$ can be done through induction

- Initialization $\beta_T(i) = 1$ (arbitrary)
- Induction

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j) \begin{cases} t = T-1 \cdots 1 \\ 1 \le i \le N \end{cases}$$



- Termination: $p(\mathcal{O}|\lambda) = \sum_{i=1}^{N} b_i(o_1)\beta_1(i)\pi_i$

As a result, computation can be effectively performed in the order of $N^2 \times T$ operations

Problem 2: Optimal State Sequence

Two approaches to find the optimal state sequence

1. finding the states q_t that are individually more likely at each time t
2. finding the single best state sequence path (i.e., maximize the posterior $P(\mathcal{Q}|\mathcal{O}, \lambda)$)

We define *Forward-Backward Variable* $\gamma_t(i)$.

$\gamma_t(i) = p(q_t = s_i|\mathcal{O}, \lambda)$

$\gamma_t(i)$ represents the probability of being in state s_i at time t, given the observation sequence $\mathcal{O}$ and the model λ .

$$\gamma_t(i) = p(q_t = s_i|\mathcal{O}, \lambda) = \frac{p(q_t = s_i, \mathcal{O}|\lambda)}{p(\mathcal{O}|\lambda)} = \frac{p(q_t = s_i, \mathcal{O}|\lambda)}{\sum_{i=1}^{N} p(q_t = s_i, \mathcal{O}|\lambda)}$$

Problem 2: Optimal State Sequence 1

The numerator of $\gamma_t(i)$ is equal to the product of $\alpha_t(i)$ and $\beta_t(i)$

$$\gamma_t(i) = \frac{p(q_t = s_i, \mathcal{O}|\lambda)}{\sum_{i=1}^{N} p(q_t = s_i, \mathcal{O}|\lambda)} = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)}$$

The individually most likely state q_t^* at each time is then

$$q_t^* = \arg \max_{1 \leq i \leq N} [\gamma_t(i)] \qquad \forall t = 1, ..., T$$

The problem with choosing the individually most likely states is that the overall state sequence may not be valid

- Consider a situation where the individually most likely states are $q_t = s_i$ and $q_{t+1} = s_j$ but the transition probability $a_{ij} = 0$

To avoid this and other problems, it is common to look for the single best state sequence, at the expense of having sub-optimal individual states

- This is accomplished with the Viterbi algorithm

Problem 2: The Viterbi algorithm

To find the single best state sequence, we define another variable

$$\delta_t(i) = \max_{q_1,\ldots,q_{t-1}} p[q_1, \ldots, q_{t-1}, q_t = s_i, o_1, o_2 \ldots o_t | \lambda]$$

which represents the highest probability along a single path that accounts for the first t observations and ends at state s_i

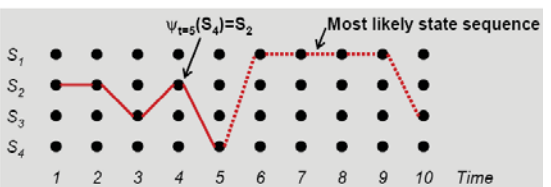
- By induction, $\delta_{t+1}(j)$ can be computed as

$$\delta_{t+1}(j) = \max_i [\delta_t(i) a_{ij}] b_j(o_{t+1})$$

- To retrieve the state sequence, we also need to keep track of the state that maximizes $\delta_t(i)$ at each time t, which is done by constructing an array

$$\psi_{t+1}(j) = \arg \max_{1 \le i \le N} [\delta_t(i) a_{ij}]$$

- $\psi_{t+1}(j)$, which is the state at time t that maximizes the probability $\delta_{t+1}(j)$ when transiting to state s_j



Problem 2: The Viterbi algorithm

The Viterbi algorithm for finding the optimal state sequence

- Initialization

  $\delta_1(i) = \pi_i b_i(o_1),\ 1 \le i \le N$

  $\psi_1(i) = 0$ (no previous states)

- Recursion ($t = 2...T$) for $j = 1...N$

  递归

  $$\delta_t(j) = \max_{1 \le i \le N} [\delta_{t-1}(i) a_{ij}] b_j(o_t)$$

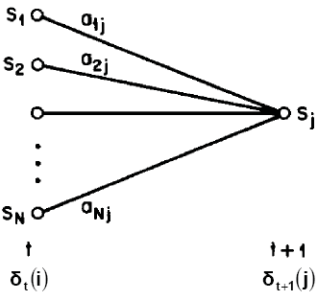
  $$\psi_t(j) = \arg \max_{1 \le i \le N} [\delta_{t-1}(i) a_{ij}]$$

  结束

- Termination

  $$p^* = \max_{1 \le i \le N} [\delta_T(i)]$$

  $$q_T^* = \arg \max_{1 \le i \le N} [\delta_T(i)]$$



From [Rabiner, 1989]

And the optimal state sequence can be retrieved by backtracking

$$q_t^* = \psi_{t+1}(q_{t+1}^*) \qquad t = T-1...1$$

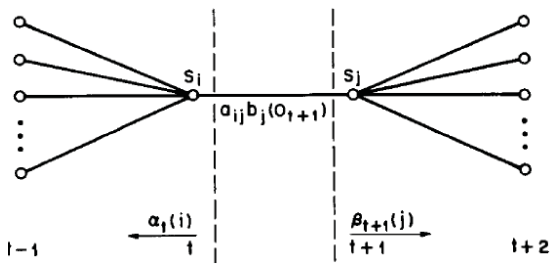
Notice that the Viterbi algorithm is similar to the Forward procedure, except that it uses a maximization over previous states instead of a summation

Problem 3: Parameter estimation

The most important and difficult problem in HMMs is to find the model parameters $\lambda = (A, B, \pi)$ from data

- HMMs are trained with the Maximum Likelihood criterion: seek model parameters that best explain the observations, as measured by $p(\mathcal{O}|\lambda)$
- This problem is solved with an iterative procedure known as Baum-Welch, which is an implementation of the EM algorithm

We define a variable $\xi_t(i,j) = p(q_t = s_i, q_{t+1} = s_j|\mathcal{O}, \lambda)$, the probability of being at state s_i at time t and state s_j at time $t+1$.



Problem 3: Parameter estimation

From the definition of $\alpha_t(i)$, and $\beta_t(j)$ and conditional probability we can rewrite the equation as follows:

$$\xi_t(i,j) = \frac{p(q_t = s_i, q_{t+1} = s_j, \mathcal{O}|\lambda)}{p(\mathcal{O}|\lambda)} = \frac{\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{p(\mathcal{O}|\lambda)} = \frac{\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}$$

Intuitive interpretation of $\gamma_t(i)$ and $\xi_t(i,j)$

- $\gamma_t(i)$ is the probability of being at state s_i at time t given the observation sequence $\mathcal{O}$ and the model λ. It can be related to $\xi_t(i,j)$ by $\gamma_t(i) = \sum_{j=1}^{N}\xi_t(i,j)$
- The summation of $\gamma_t(i)$ over time is the expected number of times that state s_i is visited or the number of transitions from s_i.

$$\sum_{t=1}^{T-1}\gamma_t(i) = \textit{expected number of transitions from state } s_i \textit{ in } \mathcal{O}$$

- Similarly, the sum of $\xi_t(i,j)$ from $t = 1$ to $t = T-1$ is the expected number of transitions from state s_i to state s_j.

$$\sum_{t=1}^{T-1}\xi_t(i,j) = \textit{expected number of transitions from state } s_i \textit{ to state } s_j$$

Problem 3: Parameter estimation

Using this line of reasoning, we can produce a method to iteratively update the parameters of an HMM by simply "counting events"

$$\hat{\pi}_i = \gamma_1(i), \textit{ expected frequency (number of times) in state } s_i \textit{ at time } (t=1)$$

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

$$\hat{b}_j(k) = \frac{\sum_{t=1, s.t\ o_t=v_k}^{T-1} \gamma_t(j)}{\sum_{t=1}^{T-1} \gamma_t(j)}$$

- where the right-hand side of the equations is computed from the "old" parameter values, and the left-hand side are the re-estimated (new) parameters
- It can be shown that each iteration of this procedure increases the likelihood of the data until a local minimum is found $p(\mathcal{O}|\lambda^{new}) \geq P(\mathcal{O}|\lambda^{old})$
- This property is due to the fact that Baum-Welch is just an implementation of the Expectation-Maximization algorithm

Problem 3: Parameter estimation

Baum-Welch is an implementation of the EM algorithm where

- The observation sequence $\mathcal{O} = \{o_1, o_2, o_3, ...\}$ is the observed data.
- The underlying state sequence $\mathcal{Q} = \{q_1, q_2, q_3, ...\}$ is the missing or hidden data
- The incomplete data likelihood is $p(\mathcal{O}|\lambda)$
- The complete data likelihood is $p(\mathcal{O}, \mathcal{Q}|\lambda)$

Therefore, the auxiliary Q function from EM becomes

$$Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{i-1}) = \mathbb{E}_{\boldsymbol{Z}}[\ln p(\boldsymbol{X}, \boldsymbol{Z}|\boldsymbol{\theta})|\boldsymbol{X}, \boldsymbol{\theta}^{i-1}]$$

from which the expected value $\mathbb{E}_{\mathcal{Q}}$ is computed by averaging over all the state sequences.

$$Q(\lambda|\lambda^{i-1}) = \mathbb{E}_{\mathcal{Q}}[\ln p(\mathcal{O}, \mathcal{Q}|\lambda)|\mathcal{O}, \lambda^{i-1}] = \sum_{\forall \mathcal{Q}} \ln p(\mathcal{O}, \mathcal{Q}|\lambda) p(\mathcal{Q}|\mathcal{O}, \lambda)$$

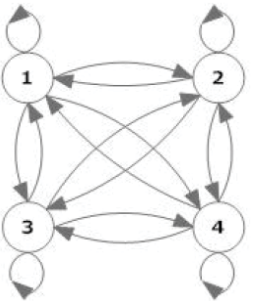
In re-estimation procedure the E-step (expectation) is calculation of the auxiliary function Q, and the M step is the maximization over λ^{i-1}.

Details on this derivation can be found in [Rabiner and Juang, 1993; Bilmes, 1998]

Types of HMM structure

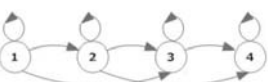
An ergodic HMM is a fully connected model, where each state can be reached in one step from every other state

- This is the most general type of HMM, and the one that has been implicitly assumed in the previous derivations



A left-right model is one where no transitions are allowed to states whose indices are lower than the current state: $a_{ij} = 0 \quad \forall j < i$

- Left-right models are best suited to model signals whose properties change over time, such as speech



Implementation issues for HMMs

Scaling

- Since $\alpha_t(i)$ involves the product of a large number of terms that are less than one, the machine precision is likely to be exceeded at some point in the computation
- To solve this problem, the $\alpha_t(i)$ are re-scaled periodically (e.g., every iteration t) to avoid underflow. A similar scaling is done to the $\beta_t(i)$

Multiple observation sequences

- The HMM derivation in these lectures is based on a single observation sequence. This becomes a problem in left-right models, since the transient nature of the states only allows a few observations to be used for each state
- For this reason, one has to use multiple observation sequences. Re-estimation formulas for multiple sequences can be found in [Rabiner and Juang, 1993]

Initial parameter estimates

- How are the initial HMM parameters chosen so that the local maximum to which Baum-Welch converges to is actually the global maximum?
- Random or uniform initial values for p and A have experimentally been found to work well in most cases
- Careful selection of initial values for B, however, has been found to be helpful in the discrete case and essential in the continuous case. These initial estimates may be found by segmenting the sequences with k-means clustering

Continuous HMMs

The discussion thus far has focused on discrete HMMs

- Discrete HMMs assume that the observations are defined by a set of discrete symbols from a finite alphabet
- In most pattern recognition applications, however, observations are inherently multidimensional and having continuous features

Two alternatives to handle continuous vectors with HMMs

- Convert the continuous multivariate observations into discrete univariate observations via a codebook (e.g., cluster the observations with k-means). This approach, however, may lead to degraded performance as a result of the discretization of the continuous signals
- Employ HMM states that have continuous observation densities $b_j(.)$

Continuous HMMs

Continuous HMMs model the observation probabilities with a continuous density function, as opposed to a multinomial

- The most common form is the Gaussian mixture model

$$b_j(\boldsymbol{o}) = \sum_{k=1}^{M} c_{jk} \mathcal{N}(\boldsymbol{o}, \boldsymbol{\mu}_{jk}, \boldsymbol{\Sigma}_{jk})$$

where $\boldsymbol{o}$ is the observation vector, c_{jk} is the mixture coefficient, $\boldsymbol{\mu}_{jk}$ is the mean vector, $\boldsymbol{\Sigma}_{jk}$ is the covariance matrix for the k^{th} Gaussian component in state s_j, and M is the number of Gaussians

The re-estimation formulas for the continuous case generalize very gracefully from the discrete HMM

- $\gamma_t(j)$ generalizes to $\gamma_t(j,k)$ which is the probability of being at state j at time t with the k^{th} mixture component accounting for the observation $\boldsymbol{o}_t$

$$\gamma_t(j,k) = \left[\frac{\alpha_t(i)\beta_t(j)}{\sum_{j=1}^{N} \alpha_t(i)\beta_t(j)} \right] \left[\frac{c_{jk}\mathcal{N}(\boldsymbol{o}_t, \boldsymbol{\mu}_{jk}, \boldsymbol{\Sigma}_{jk})}{\sum_{m=1}^{M} c_{jm}\mathcal{N}(\boldsymbol{o}_t, \boldsymbol{\mu}_{jm}, \boldsymbol{\Sigma}_{jm})} \right]$$

Continuous HMMs

The re-estimation formulas for the continuous HMM become

$$\hat{c}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)}{\sum_{t=1}^{T} \sum_{k=1}^{M} \gamma_t(j,k)} \quad \hat{\boldsymbol{\mu}}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)\boldsymbol{o}_t}{\sum_{t=1}^{T} \gamma_t(j,k)} \quad \hat{\boldsymbol{\Sigma}}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)(\boldsymbol{o}_t - \boldsymbol{\mu}_{jk})(\boldsymbol{o}_t - \boldsymbol{\mu}_{jk})^T}{\sum_{t=1}^{T} \gamma_t(j,k)}$$

- The re-estimation formula c_{jk} is the ratio between the expected number of times the system is in state j using the k^{th} mixture component, and the expected number of times the system, is in state j.
- The mean vector $\boldsymbol{\mu}_{jk}$ re-estimation formula weights the numerator in the equation c_{jk} by the observation, to produce the portion of the observation that can be accounted by mixture component.

Continuous HMMs

The re-estimation formulas for the continuous HMM become

$$\hat{c}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)}{\sum_{t=1}^{T} \sum_{k=1}^{M} \gamma_t(j,k)} \quad \hat{\boldsymbol{\mu}}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)\boldsymbol{o}_t}{\sum_{t=1}^{T} \gamma_t(j,k)} \quad \hat{\boldsymbol{\Sigma}}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)(\boldsymbol{o}_t - \boldsymbol{\mu}_{jk})(\boldsymbol{o}_t - \boldsymbol{\mu}_{jk})^T}{\sum_{t=1}^{T} \gamma_t(j,k)}$$

- The re-estimation formula c_{jk} is the ratio between the expected number of times the system is in state j using the k^{th} mixture component, and the expected number of times the system, is in state j.
- The mean vector $\boldsymbol{\mu}_{jk}$ re-estimation formula weights the numerator in the equation c_{jk} by the observation, to produce the portion of the observation that can be accounted by mixture component.
- The re-estimation formula for the transition probabilities is the same as in the discrete HMM

Motion learning and recognition

- Coordinate-free representations of a rigid body motion.
- A person perform 20 demonstrations of 9 different class of motions.
- Invariant movements are learned in the form of HMMs.
- HMM parameters are trained by Expectation-Maximization.
- The likelihood for all the HMMs is computed by Forward-Backward procedure.
- The motion is assigned to the class with higher likelihood if this likelihood is over a certain threshold.

De Schutter et al., *Recognition of 6 DOF Rigid Body Trajectories using a Coordinate-Free Representation*, ICRA, 2011.

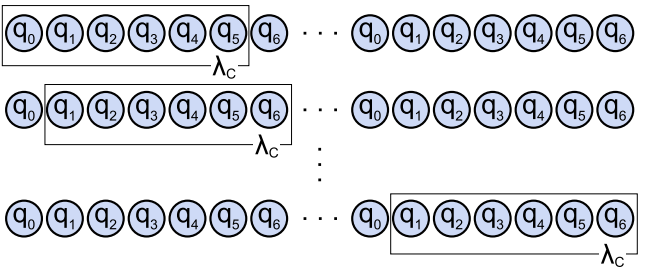
Continuous motion recognition

- Each gesture consists of a time series of hand positions (features).
- Gestures are quantized using K-means to generate K observable symbols.
- *Set Median String (SMS)* as gesture prototype.
  - SMS: the string, belonging to the set, that minimizes the sum of the distances above all the strings of a given motion class.
  - Levenshtein distance as metric

S. Iengo, S. Rossi, M. Staffa and A. Finzi, *Continuous Gesture Recognition for Flexible Human-Robot Interaction*, International Conference on Robotics and Automation, 2014.

Continuous motion recognition

- Prototype of each class learned using a DHMM.
- Temporal sliding method for continuous gesture recognition.



S. Iengo, S. Rossi, M. Staffa and A. Finzi, *Continuous Gesture Recognition for Flexible Human-Robot Interaction*, International Conference on Robotics and Automation, 2014.

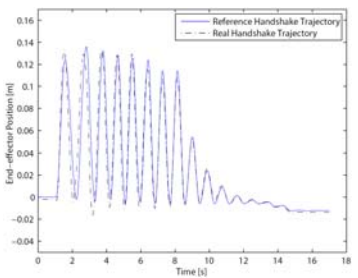
Human robot physical interaction

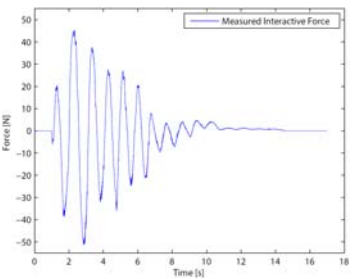
- Task: ensure the correct behaviour (stiff or compliant) of the robot during the handshake.
- An impedance control with time varying parameters to ensure a compliant behaviour when needed.
- Human impedance parameters estimated by Recursive Least Square.
- Human intentions estimation using discrete HMM.

Z. Wang, A. Peer and M. Buss, *An HMM approach to realistic haptic human-robot interaction*, World Haptics Conference, 2009.

Understand human intentions from physical interaction

- Symbols Abstractions: Human impedance parameters (Mass-Damping-Stiffness) are abstracted in 8 binary (low - high) observable symbols.
- A discrete HMM with 1 hidden state, representing the Active/Passive behaviour during the handshake is learned.





Z. Wang, A. Peer and M. Buss, *An HMM approach to realistic haptic human-robot interaction*, World Haptics Conference, 2009.

Announcements

- Reading
  - Duda, Chapter 3.10
  - Bishop, Chapter 13.1-13.2
- Exam
  - 23.07.15, 11:00 - 12:30
  - 90 min
  - 2750 (Not confirmed yet)