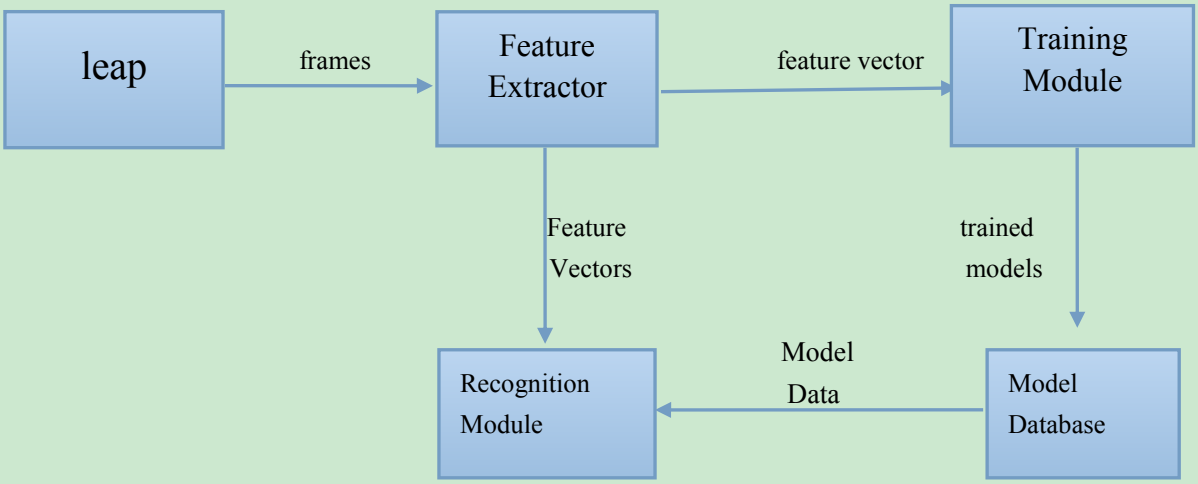
Overview

Goal: this application is designed to recognize gestures from German fingerspelling alphabet.
Solution: In order to implement continuous letter recognition with high accuracy, letter to letter transition must be considered. Each transition should be modeled as a hidden Markov Model. To begin with, we only consider the transitions of letter A to other letters, and then extend our system to cope with the whole alphabet.

Architecture

- Main functions:



- Details:

Leap:

Input:signer performing signs from German fingerspelling alphabet
Output:a sequence of frames
The leap motion controller can be used to track hands, fingers and finger-like tolls with high precision and tracking frame rate and reports discrete positions, gestures, and motion. There are two ways of getting the tracking data of frames,first is to use the well-formed API provided by the company , we can easily get access to a frame of data whenever we want by invoking the methods in the classes. A frame of data contains a list of basic tracking data,such as the characteristics of hands, fingers, bones of fingers and so on. Another way of data tracking is using the Web Socket Interface, data is sent in the form of JSON messages.

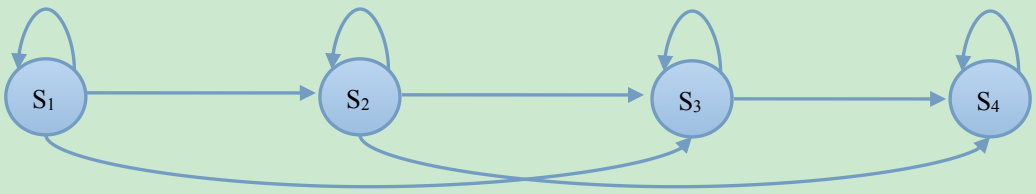
Feature extractor:

Input: a sequence of frames
Output:a a sequence of feature vectors
The LMC returns frames of data which contains basic tracking information. Robust features can be extracted and used to identify the signs. Some of the basic tracking data returned by the device include the following:
Frame ID
Numbers of hands
Finger length
Translation probability of the hand
Finger width
Numbers of fingers detected
Hand rotational matrix with respect to (w.r.t) x, y and z-axis
Hand scale factor
Hand sphere radius
Palm position w.r.t x, y, and z-axis
Hand yaw
Hand roll and
Hand pitch

Some of these parameters can be combined together to form a feature vector. These vectors are then sent to the training module to train our HMMs or can be used to do the recognition.

Training module:

Input: sequence(s) of feature vectors
Output: a set of parameters of a HMM corresponding a sign
This module is used to train our models using the feature vectors from feature extractor. Due to the fact that the performing speed of the signer can vary, Bakis-HMM is used to model our letter to letter transitions. A typical Bakis HMM looks like this:



To avoid quantization errors and to achieve a simple representation consisting of few parameters, continuous emission probability can be used. For example, the M-component Laplacian mixture density

$$b_j(X) = \sum_{m=1}^{M} b_{jm}(X) = \sum_{m=1}^{M} c_{jm} g_{jm}(X, \mu_{jm}, \sigma_{jm})$$

Where c_{jm} is the mixture coefficient, μ_{jm} the mean vector and σ_{jm} the vector of the absolute deviation of the m-th mixture component g_{jm} of state s_j .

Now the problem is to estimate the parameter set $\lambda=(\pi, A, B)$ of a HMM given one or more observation sequences O. The very first thing we should do is to choose the initial number of states for the HMM for the sign. Next, the system assigns the vectors of each sequence evenly to the states and initialize the transition matrix A. The mean and deviation of the emission distributions of each state then can be calculated according to the initial assignment. The initial HMM $\lambda^{(0)}$ is now complete with $\pi_1=1$ for Bakis-HMM.

Given $\lambda^{(0)}$ the Viterbi algorithm determines a new assignment $q^{(u)*}$ of each training sequence $O^{(u)}$ to the states of $\lambda^{(0)}$. With the transition probability matrix A and in the next iteration the mean and deviation are updated. Then the algorithm checks each mixture component of the emission distributions, if it is preferable to split it in two.The criterion is denoted by:

$$\theta_{jm} > \frac{\sum_{i=1}^{N}\sum_{m=1}^{M(i)} \theta_{im}}{\sum_{i=1}^{N} M(i)},$$

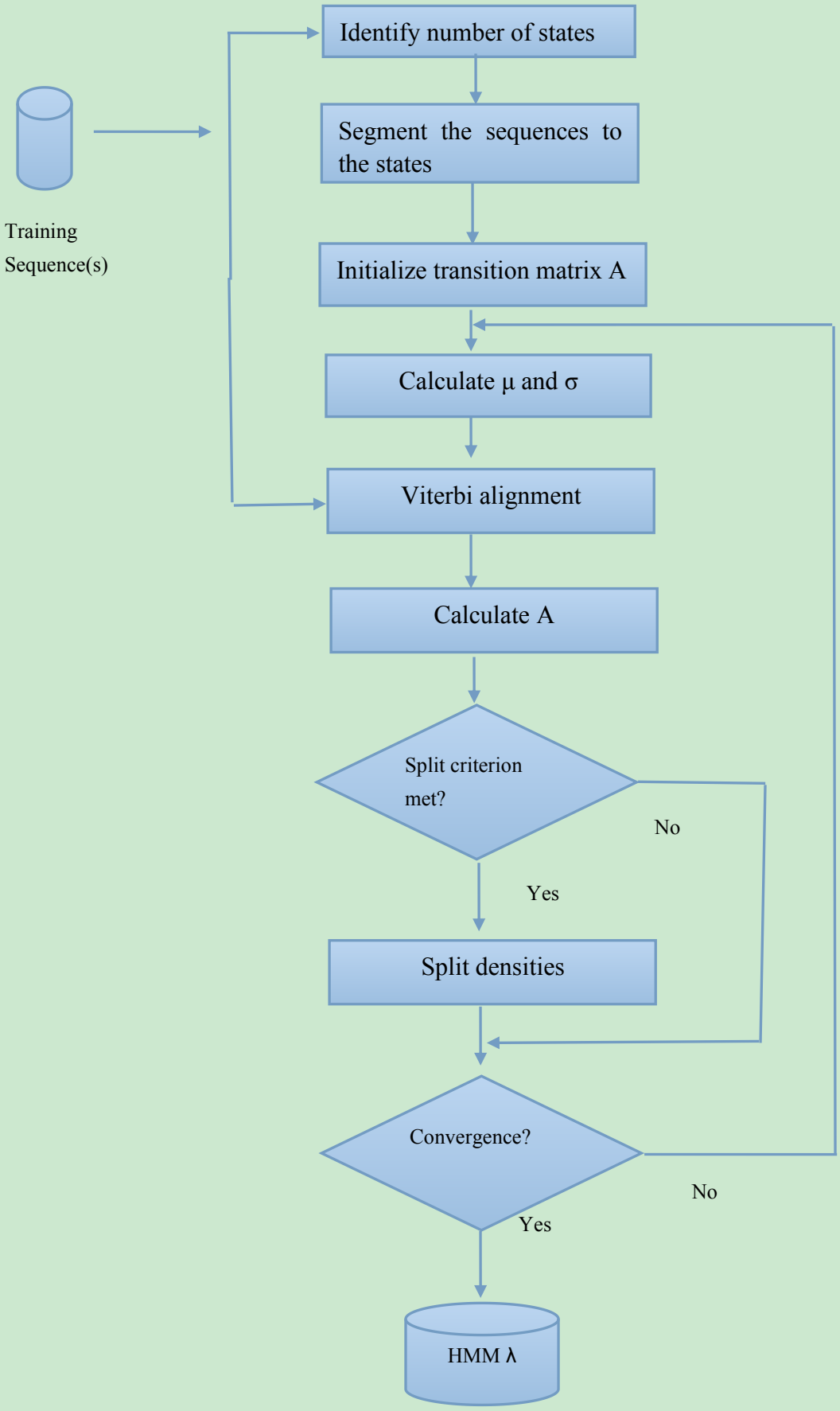
$$\text{with} \quad \theta_{jm} = -\sum_{u=1}^{U}\sum_{t=1}^{T} \ln(b_{jm}(X_t^{(u)}))\chi\left[q_t^{(u)^*} = j, b_{jm}(X_t^{(u)}) = \max_l \{b_{jl}(X_t^{(u)})\}\right]$$

Where M(i) is the number of mixture components of state i and U is the number of training sequences. χ is an abbreviation for χ[true]=1 and χ[false]=0.

Next, we decide if the HMM produces the observation sequences $O^{(u)}$ with a sufficient probability by judging if the condition below is satisfied.

$$\sum_{u=1}^{U} \ln P^{*(n)}(O^{(u)} \mid \lambda) - \sum_{u=1}^{U} \ln P^{*(n-1)}(O^{(u)} \mid \lambda) < \varepsilon$$

The above procedure can be presented by a flow diagram:


Identify number of states
Segment the sequences to the states
Initialize transition matrix A
Calculate μ and σ
Viterbi alignment
Calculate A
Split criterion met?
No
Yes
Split densities
Convergence?
No
Yes
HMM λ
Training Sequence(s)

Use-case 1

Use case name: learning the hand shape transitions.
Primary actor: a person who can perform the sign from German fingerspelling alphabet.
Goal in context: the signer perform a sign and the system capture the feature vectors from a sequence of frames, using this frames to train the HMM, deliver a set of parameters of the model to database
Preconditions:the leap motion controller is set up and connected to the system correctly.
Trigger: the person want to build models for the signs he performed

Scenario:
1.The user choose the characters from a check box to form the feature vector.
2.The user click the button “start to collect data” .
3.The user perform a sign in the effective range of the leap motion controller.
4.The user click the button”stop recording”.
5.The user click “delete example” to delete an example when he thinks that he did not perform well.
6.The user enter the name of the sign just performed.
7.The user can choose continue recording or go to the training step by clicking corresponding buttons.
8.The user hits the button “train now” to train the models of the signs using the data collected from the last step.

Exceptions:
1.If no character is chosen before collecting data, system displays appropriate error message.
2.When “start to collect data” button is clicked, if the LMC don’t recognize anything in its effective range, an error message should be displayed.
3.When “train now ” button is clicked, if no data was collected or the model has already built using the existing data, a message should be displayed.

Model database

Model database should store the parameter sets of our trained HMMs. For each HMM,the transition matrix A, the parameters of the M-component Laplacian distribution.

Recognition module

Input: a sequence of feature vectors and parameter sets from model database
Output:the name of a sign
Once all the models are trained, signs can be recognized. The leap motion controller returns a sequence of frames(feature vectors), the Forward procedure is performed to sequentially compute the probability $P(O|\lambda_i)$. The model who gives the highest probability is the sign model we want.

The Forward-Backward Procedure(we are only using the forward part):

Consider the forward variable $a_t(i)$ defined as:

$$a_t(i) = p(O_1 O_2 \cdots O_t, q_t = s_i \mid \lambda)$$

i.e., the partial observation sequence, $O_1\ O_2 \cdots\ O_t$,and state S_i at time t.
We can solve for $a_t(i)$ inductively:

1) Initialization

$a_1(i) = \pi_i\, b_i\,(O_1), \qquad 1 \le i \le N.$

2) Induction:

$$a_{t+1}(j) = \left[\sum_{i=1}^{N} a_t(i) a_{ij}\right] b_j(O_{t+1}) \qquad 1 \le t \le T-1 \qquad 1 \le j \le N$$

3) Termination:

$$p(O \mid \lambda) = \sum_{i=1}^{N} a_T(i)$$

Use-case 2

Use case name: performing the recognition of single hand shapes.
Primary actor: a person who want the sign performed by him can be recognized by the system.
Goal in context: the person performs the sign in the effective range of the LMC and the system displays the name of the sign.
Preconditions: the LMC is set up and connected to the system correctly.
Trigger: the person wants the signs that performed by him to be recognized.

Scenario:
1.The user clicks “start recognizing” button.
2.The user performs the hand shape transition.
3.The system displays the name of the sign.
4.The user clicks the “evaluation” button to see how well the performed sign matches

the models.

Exceptions:

1.when clicking “start recognizing” button, if no model has already been built, an error message should be displayed.

2.If the LMC can not recognize anything from its effective range, an error message should be displayed.